aws

Guide du développeur

Amazon Keyspaces (pour Apache Cassandra)

Amazon Keyspaces (pour Apache Cassandra): Guide du développeur

Table of Contents

Qu'est-ce qu'Amazon Keyspaces (pour Apache Cassandra) ?

Amazon Keyspaces (pour Apache Cassandra) est un service de base de données évolutif, hautement disponible et géré compatible avec Apache Cassandra. Avec Amazon Keyspaces, vous n'avez pas besoin de provisionner, de patcher ou de gérer des serveurs, ni d'installer, de maintenir ou d'exploiter des logiciels.

Amazon Keyspaces fonctionne sans serveur. Vous ne payez donc que pour les ressources que vous utilisez, et le service fait automatiquement évoluer les tables vers le haut ou vers le bas en fonction du trafic des applications. Vous pouvez créer des applications au service de milliers de demandes par seconde avec un débit et un stockage pratiquement illimités.

> **Note**
>
> Apache Cassandra est un magasin de données open source, à colonne large, conçue pour gérer de grandes quantités de données. Pour de plus amples informations, veuillez consulter Apache Cassandra.

Amazon Keyspaces facilite la migration, l'exécution et le dimensionnement des charges de travail Cassandra dans le. AWS Cloud En quelques clics sur la console de AWS gestion ou en quelques lignes de code, vous pouvez créer des espaces clés et des tables dans Amazon Keyspaces, sans déployer d'infrastructure ni installer de logiciel.

Avec Amazon Keyspaces, vous pouvez exécuter vos charges de travail Cassandra existantes en AWS utilisant le même code d'application Cassandra et les mêmes outils de développement que ceux que vous utilisez aujourd'hui.

Pour obtenir la liste des points de terminaison Régions AWS et des points de terminaison disponibles, consultez la section Points de terminaison de service pour Amazon Keyspaces.

Nous vous recommandons de commencer par lire les sections suivantes :

Rubriques

- Amazon Keyspaces : comment ça marche
- Cas d'utilisation d'Amazon Keyspaces

- Qu'est-ce que le langage de requête Cassandra (CQL) ?

Amazon Keyspaces : comment ça marche

Amazon Keyspaces élimine les frais administratifs liés à la gestion de Cassandra. Pour comprendre pourquoi, il est utile de commencer par l'architecture de Cassandra, puis de la comparer à celle d'Amazon Keyspaces.

Rubriques

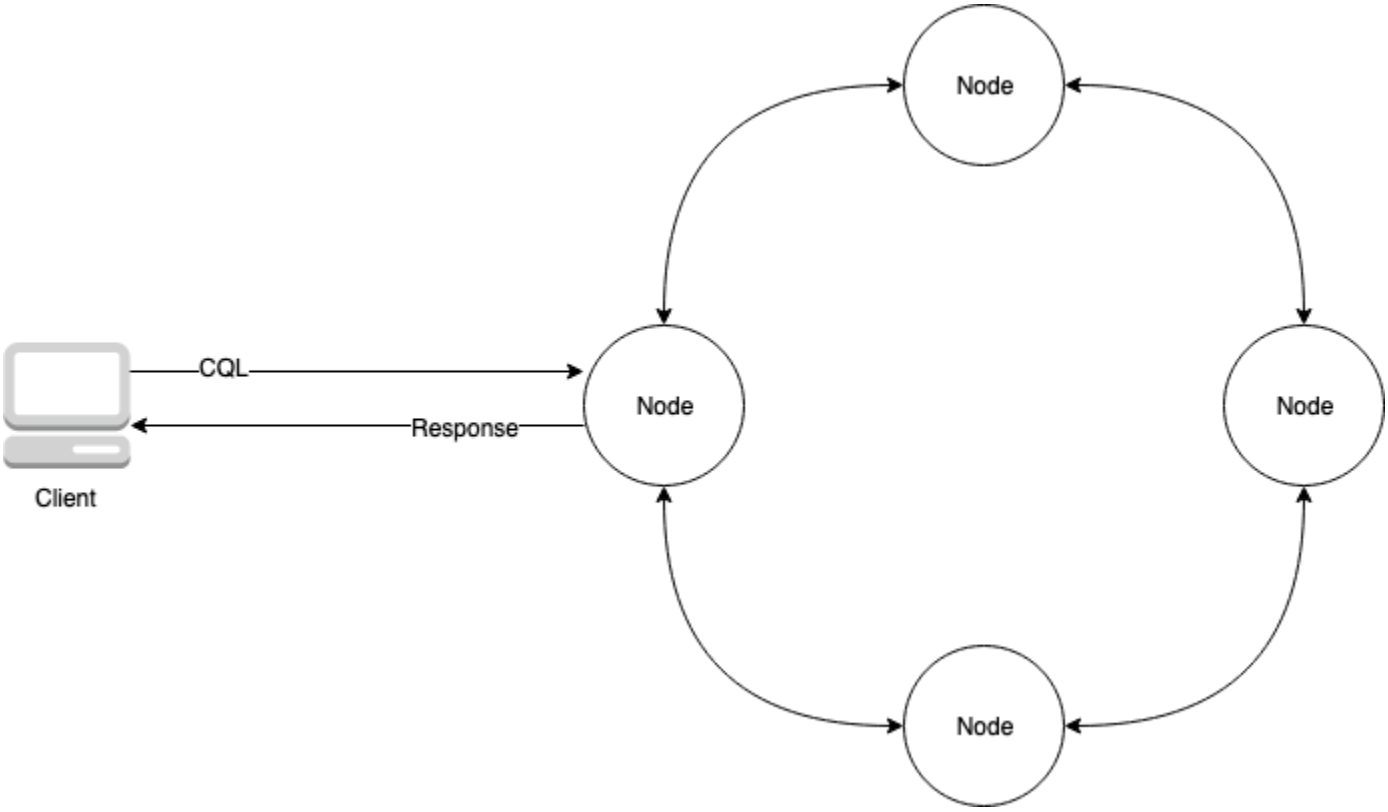
- Architecture de haut niveau : Apache Cassandra contre Amazon Keyspaces
- Modèle de données Cassandra
- Accès à Amazon Keyspaces depuis une application

Architecture de haut niveau : Apache Cassandra contre Amazon Keyspaces

Apache Cassandra traditionnelle est déployée dans un cluster composé d'un ou plusieurs nœuds. Vous êtes responsable de la gestion de chaque nœud et de l'ajout et de la suppression de nœuds à mesure que votre cluster évolue.

Un programme client accède à Cassandra en se connectant à l'un des nœuds et en émettant des instructions Cassandra Query Language (CQL). CQL est similaire à SQL, le langage populaire utilisé dans les bases de données relationnelles. Même si Cassandra n'est pas une base de données relationnelle, CQL fournit une interface familière pour interroger et manipuler des données dans Cassandra.

Le diagramme suivant montre un cluster Apache Cassandra simple, composé de quatre nœuds.



Un déploiement Cassandra de production peut être composé de centaines de nœuds, s'exécutant sur des centaines d'ordinateurs physiques dans un ou plusieurs centres de données physiques. Cela peut entraîner un fardeau opérationnel pour les développeurs d'applications qui doivent allouer, corriger et gérer des serveurs en plus de l'installation, de la maintenance et de l'exploitation des logiciels.

Avec Amazon Keyspaces (pour Apache Cassandra), vous n'avez pas besoin de provisionner, de patcher ou de gérer des serveurs. Vous pouvez donc vous concentrer sur le développement de meilleures applications. Amazon Keyspaces propose deux modes de capacité de débit pour les lectures et les écritures : à la demande et provisionné. Vous pouvez choisir le mode de capacité de débit de votre table pour optimiser le prix des lectures et des écritures en fonction de la prévisibilité et de la variabilité de votre charge de travail.

Avec le mode à la demande, vous ne payez que les lectures et les écritures que votre application effectue. Il n'est pas nécessaire de spécifier à l'avance la capacité de débit de votre table. Amazon Keyspaces gère le trafic de vos applications presque instantanément, qu'il augmente ou diminue, ce qui en fait une bonne option pour les applications dont le trafic est imprévisible.

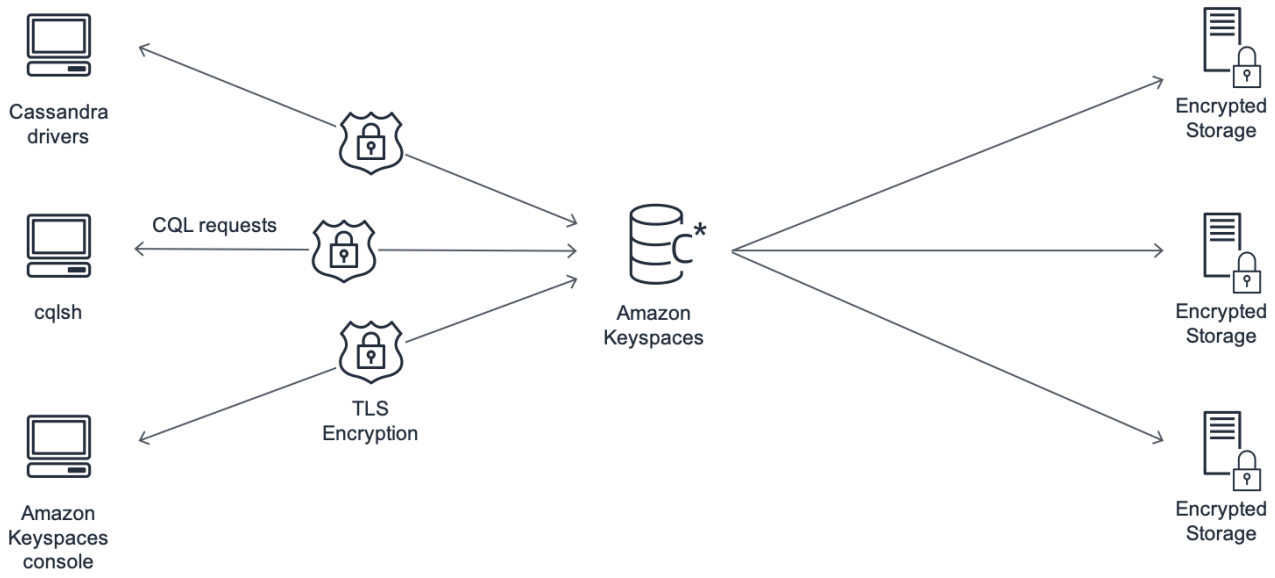
Le mode de capacité allouée permet d’optimiser le débit si vous disposez d’un trafic d’application prévisible et pouvez prévoir les besoins en capacité de votre table. Avec le mode de capacité

allouée, vous spécifiez le nombre de lectures et d'écritures par seconde dont vous pensez que votre application aura besoin. Vous pouvez augmenter et diminuer automatiquement la capacité allouée de votre table en activant la mise à l'échelle automatique.

Vous pouvez modifier le mode de capacité de votre table une fois par jour à mesure que vous en apprendrez plus sur les schémas de trafic de votre charge de travail, ou si vous prévoyez avoir une forte explosion de trafic, par exemple en raison d'un événement majeur qui, selon vous, entraînera beaucoup de trafic de table. Pour de plus amples informations sur l'allocation de capacité en lecture et en écriture, reportez-vous à la section the section called “Modes de capacité de lecture/écriture”.

Amazon Keyspaces (pour Apache Cassandra) stocke trois copies de vos données dans plusieurs zones de disponibilité pour des raisons de durabilité et de haute disponibilité. En outre, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité. Le chiffrement au repos est automatiquement activé lorsque vous créez une nouvelle table Amazon Keyspaces et toutes les connexions client nécessitent le protocole TLS (Transport Layer Security). Les fonctionnalités AWS de sécurité supplémentaires incluent la surveillance et AWS Identity and Access Managementles points de terminaison du cloud privé virtuel (VPC). Pour obtenir une vue d'ensemble de toutes les fonctionnalités de sécurité disponibles, reportez-vous à la section Sécurité.

Le schéma suivant montre l'architecture d'Amazon Keyspaces.



Un programme client accède à Amazon Keyspaces en se connectant à un point de terminaison prédéterminé (nom d'hôte et numéro de port) et en émettant des instructions CQL. Pour obtenir la

liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Modèle de données Cassandra

La façon dont vous modélisez vos données pour votre analyse de rentabilisation est essentielle pour optimiser les performances d'Amazon Keyspaces. Un modèle de données de médiocre qualité peut considérablement dégrader les performances.

Même si CQL ressemble à SQL, les backends de Cassandra et des bases de données relationnelles sont très différents et doivent être abordés différemment. Voici quelques-unes des questions les plus importantes à considérer :

Stockage

Vous pouvez visualiser vos données Cassandra dans des tables, chaque ligne représentant un enregistrement et chaque colonne un champ dans cet enregistrement.

Conception du tableau : première requête

Il n'y a pas d'instruction `JOIN` dans CQL. Par conséquent, vous devez concevoir vos tables en fonction de la forme de vos données et de la manière dont vous devez y accéder pour vos cas d'utilisation professionnels. Cela peut entraîner une dénormalisation avec des données dupliquées. Vous devez concevoir chacune de vos tables spécifiquement pour un modèle d'accès particulier.

Partitions

Vos données sont stockées dans des partitions sur le disque. Le nombre de partitions dans lesquelles vos données sont stockées et la façon dont elles sont distribuées entre les partitions sont déterminées par votre clé de partition. La façon dont vous définissez votre clé de partition peut avoir un impact significatif sur les performances de vos requêtes. Pour connaître les bonnes pratiques, consultez the section called “Création de clés de partition”.

Clé primaire

Dans Cassandra, les données sont stockées sous la forme d'une paire clé-valeur. À cette fin, chaque table Cassandra doit avoir une clé primaire, qui est la clé de chaque ligne de la table. La clé primaire est le composite d'une clé de partition requise et de colonnes de clustering facultatives. Les données qui composent la clé primaire doivent être uniques pour tous les enregistrements d'une table.

- Clé de partition : la partie clé de partition de la clé primaire est requise et détermine dans quelle partition de votre cluster les données sont stockées. La clé de partition peut être une seule colonne ; il peut aussi s'agir d'une valeur composée de deux colonnes ou plus. Vous utiliserez une clé de partition composée si une clé de partition à colonne unique entraîne une partition unique ou quelques partitions ayant la plupart des données et supportant ainsi la majorité des opérations d'E/S disque.
- Colonne de clustering : la partie optionnelle de la colonne de clustering de votre clé primaire détermine la manière dont les données sont regroupées et triées au sein de chaque partition. Si vous incluez une colonne de clustering dans votre clé primaire, la colonne de clustering peut comporter une ou plusieurs colonnes. S'il y a plusieurs colonnes dans la colonne de regroupement, l'ordre de tri est déterminé par l'ordre dans lequel les colonnes sont répertoriées dans la colonne de regroupement, de gauche à droite.

Pour plus d'informations sur le design NoSQL et Amazon Keyspaces, consultez. the section called “Conception NoSQL” Pour plus d'informations sur Amazon Keyspaces et la modélisation des données, consultez. the section called “Modélisation des données”

Accès à Amazon Keyspaces depuis une application

Amazon Keyspaces (pour Apache Cassandra) implémente l'API Apache Cassandra Query Language (CQL), afin que vous puissiez utiliser les pilotes CQL et Cassandra que vous utilisez déjà. La mise à jour de votre application est aussi simple que de mettre à jour votre pilote ou votre `cqlsh` configuration Cassandra pour qu'elle pointe vers le point de terminaison du service Amazon Keyspaces. Pour plus d'informations sur les informations d'identification requises, consultezthe section called “Informations d'identification IAM pour l' AWS authentification”.

> **Note**
>
> Pour vous aider à démarrer, vous trouverez des exemples de end-to-end code de connexion à Amazon Keyspaces à l'aide de différents pilotes clients Cassandra dans le référentiel d'exemples de code Amazon Keyspaces sur. GitHub

Considérez le programme Python suivant, qui se connecte à un cluster Cassandra et interroge une table.

```
from cassandra.cluster import Cluster
```

```
#TLS/SSL configuration goes here

ksp = 'MyKeyspace'
tbl = 'WeatherData'

cluster = Cluster(['NNN.NNN.NNN.NNN'], port=NNNN)
session = cluster.connect(ksp)

session.execute('USE ' + ksp)

rows = session.execute('SELECT * FROM ' +  tbl)
for row in rows:
    print(row)
```

Pour exécuter le même programme sur Amazon Keyspaces, vous devez :

- Ajouter le point de terminaison et le port du cluster : Par exemple, l'hôte peut être remplacé par un point de terminaison de service, tel que `cassandra.us-east-2.amazonaws.com`, et le numéro de port par : `9142`.
- Ajouter la configuration TLS/SSL : pour plus d'informations sur l'ajout de la configuration TLS/SSL pour se connecter à Amazon Keyspaces à l'aide d'un pilote Python du client Cassandra, consultez. Utilisation d'un pilote client Cassandra Python pour accéder à Amazon Keyspaces par programmation

Cas d'utilisation d'Amazon Keyspaces

Voici quelques-unes des manières dont vous pouvez utiliser Amazon Keyspaces :

- Créez des applications nécessitant une faible latence : traitez les données à haute vitesse pour les applications nécessitant une single-digit-millisecond latence, telles que la maintenance des équipements industriels, la surveillance des échanges, la gestion de flotte et l'optimisation des itinéraires.
- Créez des applications à l'aide de technologies open source : créez des applications à AWS l'aide d'API et de pilotes Cassandra open source disponibles pour un large éventail de langages de programmation, tels que Java, Python, Ruby, Microsoft .NET, Node.js, PHP, C++, Perl et Go. Pour des exemples de code, consultez Bibliothèques et outils.
- Déplacez vos charges de travail Cassandra vers le cloud : gérer vous-même les tables Cassandra prend du temps et coûte cher. Avec Amazon Keyspaces, vous pouvez configurer, sécuriser et

dimensionner les tables Cassandra AWS Cloud sans avoir à gérer l'infrastructure. Pour plus d'informations, consultez Gestion des ressources sans serveur.

Qu'est-ce que le langage de requête Cassandra (CQL) ?

Le langage de requête Cassandra (CQL) est le langage principal pour communiquer avec Apache Cassandra. Amazon Keyspaces (pour Apache Cassandra) est compatible avec l'API CQL 3.x (rétrocompatible avec la version 2.x).

Pour exécuter les requêtes CQL, vous pouvez effectuer l'une des opérations suivantes :

- Utilisez l'éditeur CQL sur le AWS Management Console.
- Use AWS CloudShell et extension cqlsh.
- Exécutez-les sur le `cqlsh` client.
- Exécutez-les par programmation à l'aide d'un pilote client Cassandra sous licence Apache 2.0.

En outre, vous pouvez accéder à Amazon Keyspaces à l'aide du AWS SDK et du. AWS Command Line Interface

Pour plus d'informations sur l'utilisation de ces méthodes pour accéder à Amazon Keyspaces, consultez. Accès à Amazon Keyspaces (pour Apache Cassandra)

Pour de plus amples informations sur CQL, veuillez consulter Référence du langage CQL pour Amazon Keyspaces (pour Apache Cassandra).

Quelle est la différence entre Amazon Keyspaces (pour Apache Cassandra) et Apache Cassandra ?

Pour établir une connexion à Amazon Keyspaces, vous pouvez utiliser un point de terminaison de AWS service public ou un point de terminaison privé à l'aide des points de terminaison Interface VPC (AWS PrivateLink) dans Amazon Virtual Private Cloud. Selon le point de terminaison utilisé, Amazon Keyspaces peut apparaître au client de l'une des manières suivantes.

AWS connexion aux terminaux de service

Il s'agit d'une connexion établie sur n'importe quel point de terminaison public. Dans ce cas, Amazon Keyspaces apparaît sous la forme d'un cluster Apache Cassandra 3.11.2 à neuf nœuds pour le client.

Interface : connexion au point de terminaison VPC

Il s'agit d'une connexion privée établie à l'aide d'un point de terminaison VPC d'interface. Dans ce cas, Amazon Keyspaces apparaît sous la forme d'un cluster Apache Cassandra 3.11.2 à trois nœuds pour le client.

Indépendamment du type de connexion et du nombre de nœuds visibles par le client, Amazon Keyspaces fournit un débit et un stockage pratiquement illimités. Pour ce faire, Amazon Keyspaces mappe les nœuds aux équilibreurs de charge qui acheminent vos requêtes vers l'une des nombreuses partitions de stockage sous-jacentes. Pour plus d’informations sur les connexions, consultez the section called “Comment ils fonctionnent”.

Amazon Keyspaces stocke les données dans des partitions. Une partition est une allocation de stockage pour une table, soutenue par des disques SSD (Solid State Drive). Amazon Keyspaces réplique automatiquement vos données sur plusieurs zones de disponibilité dans un souci de durabilité et de Région AWS haute disponibilité. À mesure que vos besoins en débit ou en stockage augmentent, Amazon Keyspaces gère la gestion des partitions pour vous et provisionne automatiquement les partitions supplémentaires requises. Pour plus d’informations, consultez the section called “Stockage”.

Amazon Keyspaces prend en charge toutes les opérations courantes du plan de données Cassandra, telles que la création d'espaces de touches et de tables, la lecture de données et l'écriture de données. Amazon Keyspaces fonctionne sans serveur, vous n'avez donc pas besoin de provisionner, de patcher ou de gérer des serveurs. Vous n'avez pas non plus besoin d'installer,

de maintenir ou d'utiliser un logiciel. Par conséquent, dans Amazon Keyspaces, vous n'avez pas besoin d'utiliser les opérations de l'API du plan de contrôle Cassandra pour gérer les paramètres des clusters et des nœuds.

Amazon Keyspaces configure automatiquement les paramètres tels que le facteur de réplication et le niveau de cohérence pour vous garantir une disponibilité, une durabilité et des performances élevées. single-digit-millisecond Pour encore plus de résilience et des lectures locales à faible latence, Amazon Keyspaces propose une réplication multirégionale.

Rubriques

Différences fonctionnelles : Amazon Keyspaces et Apache Cassandra

Voici les différences fonctionnelles entre Amazon Keyspaces et Apache Cassandra.

Rubriques

- Déclarations préparées
- Supprimer une plage
- Tables système
- Horodatages

API, opérations et types de données Apache Cassandra

Amazon Keyspaces prend en charge toutes les opérations courantes du plan de données Cassandra, telles que la création d'espaces de touches et de tables, la lecture de données et l'écriture de données. Pour voir ce qui est actuellement pris en charge, veuillez consulter API, opérations, fonctions et types de données Cassandra pris en charge dans Amazon Keyspaces.

Création et suppression asynchrones d'espaces clés et de tables

Amazon Keyspaces exécute des opérations de langage de définition de données (DDL), telles que la création et la suppression d'espaces clés et de tables, de manière asynchrone. Pour savoir comment surveiller l'état de création des ressources, consultez the section called “Création d'espaces clés” etthe section called “Création de tables”. Pour obtenir la liste des instructions DDL dans la référence du langage CQL, consultez. the section called “Instructions DDL”

Authentification et autorisation

Amazon Keyspaces (pour Apache Cassandra) utilise AWS Identity and Access Management (IAM) pour l'authentification et l'autorisation des utilisateurs, et prend en charge les politiques d'autorisation équivalentes à celles d'Apache Cassandra. Amazon Keyspaces ne prend donc pas en charge les commandes de configuration de sécurité d'Apache Cassandra.

Par lots

Amazon Keyspaces prend en charge les commandes par lots non enregistrées contenant jusqu'à 30 commandes par lot. Seules les commandes inconditionnelles INSERT, UPDATE ou DELETE sont autorisées dans un lot. Les lots consignés ne sont pas pris en charge.

Configuration de cluster

Amazon Keyspaces fonctionne sans serveur, il n'y a donc pas de clusters, d'hôtes ou de machines virtuelles Java (JVM) à configurer. Les paramètres de Cassandra relatifs au compactage, à la

compression, à la mise en cache, à la collecte des déchets et au filtrage des floraisons ne sont pas applicables à Amazon Keyspaces et sont ignorés s'ils sont spécifiés.

Connexions

Vous pouvez utiliser les pilotes Cassandra existants pour communiquer avec Amazon Keyspaces, mais vous devez les configurer différemment. Amazon Keyspaces prend en charge jusqu'à 3 000 requêtes CQL par connexion TCP et par seconde, mais le nombre de connexions qu'un pilote peut établir est illimité.

La plupart des pilotes Cassandra open source établissent un pool de connexions avec Cassandra et équilibrent la charge des requêtes sur ce pool de connexions. Amazon Keyspaces expose 9 adresses IP homologues aux pilotes, et le comportement par défaut de la plupart des conducteurs est d'établir une connexion unique avec chaque adresse IP homologue. Par conséquent, le débit de requêtes CQL maximal d'un pilote utilisant les paramètres par défaut est de 27 000 requêtes CQL par seconde.

Pour augmenter ce nombre, nous vous recommandons d'augmenter le nombre de connexions par adresse IP que votre pilote maintient dans son groupe de connexions. Par exemple, si vous définissez le nombre maximum de connexions par adresse IP sur 2, le débit maximal de votre pilote est doublé pour atteindre 54 000 requêtes CQL par seconde.

Il est recommandé de configurer les pilotes pour qu'ils utilisent 500 requêtes CQL par seconde et par connexion afin de réduire les coûts et d'améliorer la distribution. Dans ce scénario, la planification de 18 000 requêtes CQL par seconde nécessite 36 connexions. La configuration du pilote pour 4 connexions sur 9 points de terminaison permet d'obtenir 36 connexions effectuant 500 demandes par seconde. Pour plus d'informations sur les meilleures pratiques en matière de connexions, consultezthe section called “Connexions”.

Lorsque vous vous connectez à des points de terminaison VPC, il se peut qu'il y ait moins de points de terminaison disponibles. Cela signifie que vous devez augmenter le nombre de connexions dans la configuration du pilote. Pour plus d'informations sur les meilleures pratiques relatives aux connexions VPC, consultez. the section called “Connexions aux points de terminaison VPC”

`IN`mot clé

Amazon Keyspaces prend en charge le `IN` mot clé dans la `SELECT` déclaration. `IN`n'est pas compatible avec `UPDATE` et`DELETE`. Lorsque vous utilisez le `IN` mot clé dans l'`SELECT`instruction,

les résultats de la requête sont renvoyés dans l'ordre dans lequel les clés sont présentées dans l'`SELECT`instruction. Dans Cassandra, les résultats sont classés lexicographiquement.

Lors de l'utilisation `ORDER BY`, la réorganisation complète avec pagination désactivée n'est pas prise en charge et les résultats sont classés au sein d'une page. Les requêtes Slice ne sont pas prises en charge avec le `IN` mot clé. `TOKEN`Sne sont pas pris en charge par le `IN` mot clé. Amazon Keyspaces traite les requêtes avec le `IN` mot clé en créant des sous-requêtes. Chaque sous-requête est considérée comme une connexion dans le cadre de la limite de 3 000 requêtes CQL par connexion TCP et par seconde. Pour plus d'informations, consultez the section called "`INSELECT`Déclaration".

Réglage du débit des requêtes CQL

Amazon Keyspaces prend en charge jusqu'à 3 000 requêtes CQL par connexion TCP et par seconde, mais le nombre de connexions qu'un pilote peut établir est illimité.

La plupart des pilotes Cassandra open source établissent un pool de connexions avec Cassandra et équilibrent la charge des requêtes sur ce pool de connexions. Amazon Keyspaces expose 9 adresses IP homologues aux pilotes, et le comportement par défaut de la plupart des conducteurs est d'établir une connexion unique avec chaque adresse IP homologue. Par conséquent, le débit maximal de requêtes CQL d'un pilote utilisant les paramètres par défaut sera de 27 000 requêtes CQL par seconde.

Pour augmenter ce nombre, nous vous recommandons d'augmenter le nombre de connexions par adresse IP que votre pilote maintient dans son groupe de connexions. Par exemple, si vous définissez le nombre maximum de connexions par adresse IP sur 2, vous doublerez le débit maximal de votre pilote pour le porter à 54 000 requêtes CQL par seconde.

Pour plus d'informations sur les meilleures pratiques en matière de connexions, consultezthe section called "Connexions".

Lorsque vous vous connectez à des points de terminaison VPC, moins de points de terminaison sont disponibles. Cela signifie que vous devez augmenter le nombre de connexions dans la configuration du pilote. Pour plus d'informations sur les meilleures pratiques relatives aux connexions des points de terminaison VPC, consultez. the section called "Connexions aux points de terminaison VPC"

`FROZEN`collections

Le `FROZEN` mot clé de Cassandra sérialise plusieurs composants d'un type de données de collection en une seule valeur immuable traitée comme un. `BLOB` `INSERT`et les `UPDATE` instructions remplacent l'ensemble de la collection.

Amazon Keyspaces prend en charge par défaut jusqu'à cinq niveaux d'imbrication pour les collections figées. Pour plus d'informations, consultez the section called “Quotas de service Amazon Keyspaces”.

Amazon Keyspaces ne prend pas en charge les comparaisons d'inégalités qui utilisent l'intégralité de la collection figée dans un conditionnel `UPDATE` ou `SELECT` une instruction. Le comportement des collections et des collections gelées est le même dans Amazon Keyspaces.

Lorsque vous utilisez des collections figées avec des horodatages côté client, dans le cas où l'horodatage d'une opération d'écriture est le même que celui d'une colonne existante qui n'est ni expirée ni gravée, Amazon Keyspaces n'effectue aucune comparaison. Au lieu de cela, il permet au serveur de déterminer le dernier rédacteur, et c'est le dernier rédacteur qui gagne.

Pour plus d'informations sur les collections congelées, consultezthe section called “Types de collections”.

Transactions légères

Amazon Keyspaces (pour Apache Cassandra) prend entièrement en charge les fonctionnalités de comparaison et de définition surINSERT, et DELETE les commandesUPDATE, connues sous le nom de transactions légères (LWT) dans Apache Cassandra. En tant qu'offre sans serveur, Amazon Keyspaces (pour Apache Cassandra) fournit des performances constantes à toutes les échelles, y compris pour les transactions légères. Avec Amazon Keyspaces, l'utilisation de transactions légères n'entraîne aucune baisse de performance.

Equilibrage de charge

Les entrées du `system.peers` tableau correspondent aux équilibreurs de charge Amazon Keyspaces. Pour de meilleurs résultats, nous vous recommandons d'utiliser une politique d'équilibrage de charge circulaire et d'ajuster le nombre de connexions par IP en fonction des besoins de votre application.

Pagination

Amazon Keyspaces pagine les résultats en fonction du nombre de lignes lues pour traiter une demande, et non du nombre de lignes renvoyées dans le jeu de résultats. Par conséquent, certaines pages peuvent contenir moins de lignes que ce que vous spécifiez dans TAILLE DE PAGE pour les requêtes filtrées. En outre, Amazon Keyspaces pagine automatiquement les résultats après avoir lu

1 Mo de données afin de fournir aux clients des performances de lecture cohérentes à un chiffre en millisecondes. Pour plus d'informations, consultez the section called "Pagination des résultats".

Cloisons

Le partitionneur par défaut d'Amazon Keyspaces est compatible avec Cassandra. `Murmur3Partitioner` De plus, vous avez le choix d'utiliser soit les Amazon Keyspaces, soit les appareils compatibles avec `DefaultPartitioner` Cassandra. `RandomPartitioner`

Avec Amazon Keyspaces, vous pouvez modifier le partitionneur de votre compte en toute sécurité sans avoir à recharger vos données Amazon Keyspaces. Une fois la modification de configuration terminée, qui prend environ 10 minutes, les clients verront le nouveau partitionneur se configurer automatiquement lors de leur prochaine connexion. Pour plus d'informations, consultez the section called "Utilisation de partitionneurs".

Déclarations préparées

Amazon Keyspaces prend en charge l'utilisation d'instructions préparées pour les opérations en langage de manipulation de données (DML), telles que la lecture et l'écriture de données. Amazon Keyspaces ne prend actuellement pas en charge l'utilisation d'instructions préparées pour les opérations en langage de définition de données (DDL), telles que la création de tables et d'espaces de touches. Les opérations DDL doivent être exécutées en dehors des instructions préparées.

Supprimer une plage

Amazon Keyspaces prend en charge la suppression de lignes dans une plage. Une plage est un ensemble contigu de lignes au sein d'une partition. Vous spécifiez une plage dans une opération DELETE à l'aide d'une clause WHERE. Vous pouvez spécifier que la plage correspond à une partition complète.

En outre, vous pouvez définir une plage comme un sous-ensemble de lignes contiguës au sein d'une partition en utilisant des opérateurs relationnels (par exemple, « **>** », « **<** ») ou en incluant la clé de partition et en omettant une ou plusieurs colonnes de clustering. Avec Amazon Keyspaces, vous pouvez supprimer jusqu'à 1 000 lignes au sein d'une plage en une seule opération. De plus, les suppressions de plages sont atomiques, mais ne sont pas isolées.

Tables système

Amazon Keyspaces remplit les tables système requises par les pilotes open source Cassandra d'Apache 2.0. Les tables système visibles par un client contiennent des informations propres à

l'utilisateur authentifié. Les tables système sont entièrement contrôlées par Amazon Keyspaces et sont en lecture seule.

L'accès en lecture seule aux tables système est requis, et vous pouvez le contrôler à l'aide des politiques d'accès IAM. Pour plus d'informations, consultez the section called "Gestion des accès à l'aide de politiques". Vous devez définir des politiques de contrôle d'accès basées sur des balises pour les tables système différemment selon que vous utilisez le AWS SDK ou des appels d'API Cassandra Query Language (CQL) via les pilotes Cassandra et les outils de développement. Pour en savoir plus sur le contrôle d'accès basé sur des balises pour les tables système, consultezthe section called " Accès aux ressources Amazon Keyspaces basé sur des balises".

Si vous accédez à Amazon Keyspaces via des points de terminaison Amazon VPC, des entrées apparaissent dans le tableau pour `system.peers` chaque point de terminaison Amazon VPC qu'Amazon Keyspaces est autorisé à consulter. Par conséquent, votre pilote Cassandra peut émettre un message d'avertissement concernant le nœud de contrôle lui-même dans le `system.peers` tableau. Vous pouvez ignorer cet avertissement en toute sécurité.

Horodatages

Dans Amazon Keyspaces, les horodatages au niveau des cellules compatibles avec les horodatages par défaut d'Apache Cassandra sont une fonctionnalité optionnelle.

La `USING TIMESTAMP` clause et la `WRITETIME` fonction ne sont disponibles que lorsque les horodatages côté client sont activés pour une table. Pour en savoir plus sur les horodatages côté client dans Amazon Keyspaces, consultez. Horodatages côté client

API, opérations, fonctions et types de données Cassandra pris en charge dans Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) est compatible avec l'API Cassandra Query Language (CQL) 3.11 (rétrocompatible avec la version 2.x).

Amazon Keyspaces prend en charge toutes les opérations courantes du plan de données Cassandra, telles que la création d'espaces de touches et de tables, la lecture et l'écriture de données.

Les sections suivantes répertorient les fonctionnalités prises en charge.

Rubriques

Support de l'API Cassandra

Opération API	Pris en charge
`CREATE KEYSPACE`	Oui
`ALTER KEYSPACE`	Oui
`DROP KEYSPACE`	Oui
`CREATE TABLE`	Oui
`ALTER TABLE`	Oui
`DROP TABLE`	Oui
`CREATE INDEX`	Non
`DROP INDEX`	Non
`UNLOGGED BATCH`	Oui
`LOGGED BATCH`	Non
`SELECT`	Oui
`INSERT`	Oui
`DELETE`	Oui
`UPDATE`	Oui
`USE`	Oui

Opération API	Pris en charge
`CREATE TYPE`	Non
`ALTER TYPE`	Non
`DROP TYPE`	Non
`CREATE TRIGGER`	Non
`DROP TRIGGER`	Non
`CREATE FUNCTION`	Non
`DROP FUNCTION`	Non
`CREATE AGGREGATE`	Non
`DROP AGGREGATE`	Non
`CREATE MATERIALIZED VIEW`	Non
`ALTER MATERIALIZED VIEW`	Non
`DROP MATERIALIZED VIEW`	Non
`TRUNCATE`	Non

Support de l'API du plan de contrôle Cassandra

Amazon Keyspaces étant géré, les opérations de l'API du plan de contrôle Cassandra pour gérer les paramètres des clusters et des nœuds ne sont pas requises. Par conséquent, les fonctionnalités Cassandra suivantes ne sont pas applicables.

Fonctionnalité	Raison
Basculement des écritures durables	Toutes les écritures sont durables
Lire les paramètres de réparation	Ne s’applique pas

Fonctionnalité	Raison
Secondes de grâce GC	Ne s’applique pas
Paramètres du filtre de Bloom	Ne s’applique pas
Paramètres de compactage	Ne s’applique pas
Compression settings (Paramètres de compression)	Ne s’applique pas
Paramètres de mise en cache	Ne s’applique pas
Paramètres de sécurité	Remplacé par IAM

Support de l'API du plan de données Cassandra

Fonctionnalité	Pris en charge
Support JSON pour les instructions SELECT et INSERT	Oui
Colonnes statiques	Oui
Time to Live (TTL)	Oui

Support de la fonction Cassandra

Pour plus d'informations sur les fonctions prises en charge, consultezthe section called “Fonctions intégrées”.

Fonction	Pris en charge
Fonctions `Aggregate`	Non
`Blobconversion`	Oui
`Cast`	Oui

Fonction	Pris en charge
Fonctions `Datetime`	Oui
Fonctions de conversion temporelle	Oui
Fonctions `TimeUuid`	Oui
`Token`	Oui
`User defined functions (UDF)`	Non
`Uuid`	Oui

Support des types de données Cassandra

Type de données	Pris en charge	Remarque
`ascii`	Oui	
`bigint`	Oui	
`blob`	Oui	
`boolean`	Oui	
`counter`	Oui	
`date`	Oui	
`decimal`	Oui	
`double`	Oui	
`float`	Oui	
`frozen`	Oui	
`inet`	Oui	

Type de données	Pris en charge	Remarque
`int`	Oui	
`list`	Oui	
`map`	Oui	
`set`	Oui	
`smallint`	Oui	
`text`	Oui	
`time`	Oui	
`timestamp`	Oui	
`timeuuid`	Oui	
`tinyint`	Oui	
`tuple`	Oui	
`user-defined types` (UDT)	Non	Pour refactoriser les UDT avec des tampons de protocole, consultez Amazon Keyspaces Protocol Buffers.
`uuid`	Oui	
`varchar`	Oui	
`varint`	Oui	

Niveaux de cohérence d'Apache Cassandra pris en charge dans Amazon Keyspaces

Les rubriques de cette section décrivent les niveaux de cohérence d'Apache Cassandra pris en charge pour les opérations de lecture et d'écriture dans Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Niveaux de cohérence en écriture

Amazon Keyspaces reproduit toutes les opérations d'écriture trois fois sur plusieurs zones de disponibilité pour garantir la durabilité et la haute disponibilité. Les écritures sont stockées durablement avant d'être reconnues à l'aide du niveau de cohérence `LOCAL_QUORUM`. Pour chaque écriture de 1 Ko, vous êtes facturé 1 unité de capacité d'écriture (WCU) pour les tables utilisant le mode de capacité allouée ou 1 unité de demande d'écriture (WRU) pour les tables utilisant le mode à la demande.

Vous pouvez utiliser`cqlsh`pour définir la cohérence de toutes les requêtes de la session en cours sur`LOCAL_QUORUM`en utilisant le code suivant.

```
CONSISTENCY LOCAL_QUORUM;
```

Pour configurer le niveau de cohérence par programmation, vous pouvez définir la cohérence à l'aide des pilotes client Cassandra appropriés. Par exemple, les pilotes Java de la version 4.x vous permettent de définir le niveau de cohérence dans`app config`fichier comme indiqué ci-dessous.

```
basic.request.consistency = LOCAL_QUORUM
```

Si vous utilisez un pilote Java Cassandra version 3.x, vous pouvez spécifier le niveau de cohérence de la session en ajoutant`.withQueryOptions(new QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)`comme indiqué dans l'exemple de code suivant.

```
Session session = Cluster.builder()
                      .addContactPoint(endPoint)
                      .withPort(portNumber)
                      .withAuthProvider(new SigV4AuthProvider("us-east-2"))
                      .withSSL()
                      .withQueryOptions(new
 QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
                      .build()
                      .connect();
```

Pour configurer le niveau de cohérence pour des opérations d'écriture spécifiques, vous pouvez définir la cohérence lorsque vous appelez`QueryBuilder.insertInto`avec un`setConsistencyLevel`argument lorsque vous utilisez le pilote Java.

Lire les niveaux de cohérence

Amazon Keyspaces prend en charge trois niveaux de cohérence de lecture :`ONE`,`LOCAL_ONE`, et`LOCAL_QUORUM`. Au cours d'une`LOCAL_QUORUM`lu, Amazon Keyspaces renvoie une réponse reflétant les mises à jour les plus récentes issues de toutes les opérations d'écriture précédentes réussies. L'utilisation du niveau de cohérence `ONE` ou `LOCAL_ONE` peut améliorer les performances et la disponibilité de vos demandes de lecture, mais la réponse peut ne pas refléter les résultats d'une écriture récemment terminée.

Pour chaque lecture de 4 Ko à l'aide de la cohérence `ONE` ou `LOCAL_ONE`, vous êtes facturé 0,5 unités de capacité de lecture (RCU) pour les tables utilisant le mode de capacité allouée ou 0,5 unités de demande de lecture (RRU) pour les tables utilisant le mode à la demande. Pour chaque lecture de 4 Ko avec la cohérence `LOCAL_QUORUM`, vous êtes facturé 1 unité de capacité de lecture (RCU) pour les tables utilisant le mode de capacité allouée ou 1 unité de demande de lecture (RRU) pour les tables utilisant le mode à la demande.

Facturation basée sur la cohérence de lecture et le mode de débit de capacité de lecture par table pour chaque 4 Ko de lecture

Niveau de cohérence	Alloué	A la demande
`ONE`	0,5 RCU (Read Capacity Unit, unité de capacité de lecture)	0,5 RRU
`LOCAL_ONE`	0,5 RCU (Read Capacity Unit, unité de capacité de lecture)	0,5 RRU

Niveau de cohérence	Alloué	A la demande
LOCAL_QUORUM	1 RCU	1 RRU

Pour spécifier une cohérence différente pour les opérations de lecture, appelez`QueryBuilder.select`avec `unsetConsistencyLevel`argument lorsque vous utilisez le pilote Java.

Niveaux de cohérence non pris en charge

Les niveaux de cohérence suivants ne sont pas pris en charge par Amazon Keyspaces et peuvent entraîner des exceptions.

Niveaux de cohérence non pris en charge

Apache Cassandra	Amazon Keyspaces
EACH_QUORUM	Non pris en charge
QUORUM	Non pris en charge
ALL	Non pris en charge
TWO	Non pris en charge
THREE	Non pris en charge
ANY	Non pris en charge
SERIAL	Non pris en charge
LOCAL_SERIAL	Non pris en charge

Accès à Amazon Keyspaces (pour Apache Cassandra)

Vous pouvez accéder à Amazon Keyspaces à l'aide de la console AWS CloudShell, par programmation en exécutant un `cqlsh` client, le AWS SDK, ou en utilisant un pilote Cassandra sous licence Apache 2.0. Amazon Keyspaces prend en charge les pilotes et les clients compatibles avec Apache Cassandra 3.11.2. Avant d'accéder à Amazon Keyspaces, vous devez terminer la configuration, AWS Identity and Access Management puis accorder à une identité IAM les autorisations d'accès à Amazon Keyspaces.

Con AWS Identity and Access Management figuration

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d‘inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l‘utilisateur racine a accès à l‘ensemble des Services AWS et des ressources de ce compte. Pour des raisons de sécurité, attribuez un accès administratif à un utilisateur et utilisez uniquement l'utilisateur root pour effectuer les tâches nécessitant un accès utilisateur root.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l‘activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, accordez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connectez-vous en tant qu'utilisateur disposant d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribuer l'accès à des utilisateurs supplémentaires

1. Dans IAM Identity Center, créez un ensemble d'autorisations conforme aux meilleures pratiques en matière d'application des autorisations du moindre privilège.

Pour obtenir des instructions, voir Création d'un ensemble d'autorisations dans le guide de AWS IAM Identity Center l'utilisateur.

2. Affectez des utilisateurs à un groupe, puis attribuez un accès d'authentification unique au groupe.

   Pour obtenir des instructions, voir Ajouter des groupes dans le guide de AWS IAM Identity Center l'utilisateur.

Configuration d'Amazon Keyspaces

L'accès aux ressources Amazon Keyspaces est géré à l'aide d'IAM. À l'aide d'IAM, vous pouvez associer des politiques aux utilisateurs, aux rôles et aux identités fédérées IAM qui accordent des autorisations de lecture et d'écriture à des ressources spécifiques dans Amazon Keyspaces.

Pour commencer à accorder des autorisations à une identité IAM, vous pouvez utiliser l'une des politiques AWS gérées pour Amazon Keyspaces :

- AmazonKeyspacesFullAccess— cette politique accorde l'autorisation d'accéder à toutes les ressources d'Amazon Keyspaces avec un accès complet à toutes les fonctionnalités.
- AmazonKeyspacesReadOnlyAccess_v2 : cette politique accorde des autorisations de lecture seule à Amazon Keyspaces.

Pour une explication détaillée des actions définies dans les politiques gérées, consultezthe section called “Politiques gérées par AWS”.

Pour limiter l'étendue des actions qu'une identité IAM peut effectuer ou limiter les ressources auxquelles l'identité peut accéder, vous pouvez créer une politique personnalisée qui utilise la stratégie `AmazonKeyspacesFullAccess` gérée comme modèle et supprimer toutes les autorisations dont vous n'avez pas besoin. Vous pouvez également limiter l'accès à des espaces clés ou à des tables spécifiques. Pour plus d'informations sur la façon de restreindre les actions ou de limiter l'accès à des ressources spécifiques dans Amazon Keyspaces, consultez. the section called “Comment Amazon Keyspaces fonctionne avec IAM”

Pour accéder à Amazon Keyspaces après avoir créé Compte AWS et créé une politique qui accorde à une identité IAM l'accès à Amazon Keyspaces, passez à l'une des sections suivantes :

- Utilisation de la console

- En utilisant AWS CloudShell
- Connexion par programme

Accès à Amazon Keyspaces à l'aide de la console

Vous pouvez accéder à la console d'Amazon Keyspaces à l'adresse. https://console.aws.amazon.com/keyspaces/home Pour plus d'informations sur AWS Management Console l'accès, consultez la section Contrôle de l'accès des utilisateurs IAM au AWS Management Console dans le guide de l'utilisateur IAM.

Vous pouvez utiliser la console pour effectuer les opérations suivantes dans Amazon Keyspaces :

- Créez, supprimez et gérez les espaces clés et les tables.
- Surveillez les statistiques importantes d'un tableau dans l'onglet Surveiller d'un tableau :
  - Taille du tableau facturable (octets)
  - Indicateurs de capacité
- Exécutez des requêtes à l'aide de l'éditeur CQL, par exemple pour insérer, mettre à jour et supprimer des données.
- Modifiez la configuration du partitionneur du compte.
- Consultez les indicateurs de performance et d'erreur du compte sur le tableau de bord.

Pour savoir comment créer un espace de touches et une table Amazon Keyspaces et comment les configurer à l'aide d'exemples de données d'application, consultez. Commencer à utiliser Amazon Keyspaces (pour Apache Cassandra)

Utilisation AWS CloudShell pour accéder à Amazon Keyspaces

AWS CloudShell est un shell pré-authentifié basé sur un navigateur que vous pouvez lancer directement depuis le. AWS Management Console Vous pouvez exécuter AWS CLI des commandes sur AWS des services à l'aide de votre shell préféré (Bash PowerShell ou Z shell). Pour utiliser Amazon Keyspaces à l'aide de`cqlsh`, vous devez installer le. `cqlsh-expansion` Pour les instructions d'`cqlsh-expansion`installation, voirthe section called “Utilisation de l’`cqlsh-expansion`”.

Vous lancez AWS CloudShell à partir de AWS Management Console, et les AWS informations d'identification que vous avez utilisées pour vous connecter à la console sont automatiquement

disponibles dans une nouvelle session shell. Cette pré-authentification des AWS CloudShell utilisateurs vous permet d'ignorer la configuration des informations d'identification lorsque vous interagissez avec AWS des services tels qu'Amazon Keyspaces `cqlsh` en utilisant AWS CLI la version 2 (préinstallée sur l'environnement informatique du shell).

Obtention des autorisations IAM pour AWS CloudShell

À l'aide des ressources de gestion des accès fournies par AWS Identity and Access Management, les administrateurs peuvent accorder des autorisations aux utilisateurs IAM afin qu'ils puissent accéder aux fonctionnalités de l'environnement AWS CloudShell et les utiliser.

Le moyen le plus rapide pour un administrateur d'accorder l'accès aux utilisateurs est d'utiliser une politique AWS gérée. Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. La politique AWS gérée suivante pour CloudShell peut être attachée aux identités IAM :

- `AWSCloudShellFullAccess`: accorde l'autorisation d'utilisation AWS CloudShell avec un accès complet à toutes les fonctionnalités.

Si vous souhaitez limiter l'étendue des actions qu'un utilisateur IAM peut effectuer AWS CloudShell, vous pouvez créer une politique personnalisée qui utilise la stratégie `AWSCloudShellFullAccess` gérée comme modèle. Pour plus d'informations sur la limitation des actions disponibles pour les utilisateurs dans CloudShell, consultez la section Gestion de l' AWS CloudShell accès et de l'utilisation avec les politiques IAM dans le Guide de l'AWS CloudShell utilisateur.

> **Note**
>
> Votre identité IAM nécessite également une politique autorisant les appels vers Amazon Keyspaces.

Vous pouvez utiliser une politique AWS gérée pour donner à votre identité IAM l'accès à Amazon Keyspaces, ou commencer par utiliser la politique gérée comme modèle et supprimer les autorisations dont vous n'avez pas besoin. Vous pouvez également limiter l'accès à des espaces clés et à des tables spécifiques afin de créer une politique personnalisée. La politique gérée suivante pour Amazon Keyspaces peut être associée aux identités IAM :

- AmazonKeyspacesFullAccess— Cette politique autorise l'utilisation d'Amazon Keyspaces avec un accès complet à toutes les fonctionnalités.

Pour une explication détaillée des actions définies dans la politique gérée, consultezthe section called “Politiques gérées par AWS”.

Pour plus d'informations sur la façon de restreindre les actions ou de limiter l'accès à des ressources spécifiques dans Amazon Keyspaces, consultez. the section called “Comment Amazon Keyspaces fonctionne avec IAM”

Interagir avec Amazon Keyspaces à l'aide de AWS CloudShell

Après le lancement AWS CloudShell depuis le AWS Management Console, vous pouvez immédiatement commencer à interagir avec Amazon Keyspaces à l'aide `cqlsh` de l'interface de ligne de commande. Si vous ne l'avez pas encore installé`cqlsh-expansion`, consultez the section called “Utilisation de l’`cqlsh-expansion`” les étapes détaillées.

> **Note**
>
> Lorsque vous utilisez l'`cqlsh-expansion`entrée AWS CloudShell, vous n'avez pas besoin de configurer les informations d'identification avant de passer des appels, car vous êtes déjà authentifié dans le shell.

Connectez-vous à Amazon Keyspaces et créez un nouvel espace de touches. Lisez ensuite une table système pour confirmer que le keyspace a été créé à l'aide de AWS CloudShell

1. À partir de AWS Management Console, vous pouvez lancer CloudShell en choisissant les options suivantes disponibles dans la barre de navigation :
   - Choisissez l' CloudShell icône.
   - Commencez à taper « cloudshell » dans le champ de recherche, puis choisissez l' CloudShelloption.
2. Vous pouvez établir une connexion à Amazon Keyspaces à l'aide de la commande suivante. Assurez-vous de remplacer *`cassandra.us-east-1.amazonaws.com`* par le point de terminaison correspondant à votre région.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

   Si la connexion est établie, vous devriez obtenir un résultat similaire à celui de l'exemple suivant.

   ```
   Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
   ```

```
[cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh current consistency level is ONE.
cqlsh>
```

3. Créez un nouvel espace de touches portant le nom`mykeyspace`. Pour ce faire, vous pouvez utiliser la commande suivante.

```
CREATE KEYSPACE mykeyspace WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

4. Pour confirmer que le keyspace a été créé, vous pouvez lire une table système à l'aide de la commande suivante.

```
SELECT * FROM system_schema_mcs.keyspaces WHERE keyspace_name = 'mykeyspace';
```

Si l'appel aboutit, la ligne de commande affiche une réponse du service similaire au résultat suivant :

```
 keyspace_name  | durable_writes | replication
----------------+----------------
+-------------------------------------------------------------------------------------
 mykeyspace     |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(1 rows)
```

Connexion programmatique à Amazon Keyspaces

Cette rubrique décrit les étapes requises pour se connecter à Amazon Keyspaces par programmation. Il vous guide dans la création d'informations d'identification IAM et répertorie les points de terminaison AWS de service disponibles. La dernière section explique comment se connecter à Amazon Keyspaces à l'aide de cqlsh. Pour des step-by-step didacticiels sur la connexion à Amazon Keyspaces à l'aide de différents pilotes Apache Cassandra, consultez. the section called “Utilisation d'un pilote client Cassandra” Pour un step-by-step didacticiel expliquant comment se connecter à Amazon Keyspaces depuis un point de terminaison Amazon VPC, consultez. the section called “Connexion aux points de terminaison VPC”

> **Note**
>
> Pour vous aider à démarrer, vous trouverez des exemples de end-to-end code de connexion à Amazon Keyspaces à l'aide de différents pilotes clients Cassandra dans le référentiel d'exemples de code Amazon Keyspaces sur. GitHub

Amazon Keyspaces prend en charge les pilotes et les clients compatibles avec Apache Cassandra 3.11.2. Cela suppose que vous avez déjà suivi les instructions de AWS configuration dansAccès à Amazon Keyspaces.

Si vous en avez déjà un Compte AWS, consultez les rubriques suivantes pour savoir comment accéder à Amazon Keyspaces à l'aide de cqlsh par programmation :

Rubriques

- Création d'informations d'identification pour accéder à Amazon Keyspaces par programmation
- Points de terminaison de service pour Amazon Keyspaces
- Utilisation cqlsh pour se connecter à Amazon Keyspaces
- Utilisation des AWS CLI
- Utilisation de l'API
- Utilisation d'Amazon Keyspaces avec un SDK AWS
- Utilisation d'un pilote client Cassandra pour accéder à Amazon Keyspaces par programmation
- Tutoriel : Connexion à Amazon Keyspaces depuis Amazon Elastic Kubernetes Service

Création d'informations d'identification pour accéder à Amazon Keyspaces par programmation

Pour fournir aux utilisateurs et aux applications des informations d'identification leur permettant d'accéder par programmation aux ressources Amazon Keyspaces, vous pouvez effectuer l'une des opérations suivantes :

- Créez des informations d'identification spécifiques au service similaires au nom d'utilisateur et au mot de passe traditionnels que Cassandra utilise pour l'authentification et la gestion des accès. AWS les informations d'identification spécifiques au service sont associées à un utilisateur AWS Identity and Access Management (IAM) spécifique et ne peuvent être utilisées que pour le service

pour lequel elles ont été créées. Pour plus d'informations, consultez la section Utilisation d'IAM avec Amazon Keyspaces (pour Apache Cassandra) dans le guide de l'utilisateur d'IAM.

> **Warning**
>
> Les utilisateurs IAM disposent d'informations d'identification à long terme, ce qui présente un risque de sécurité. Pour atténuer ce risque, nous vous recommandons de ne fournir à ces utilisateurs que les autorisations dont ils ont besoin pour effectuer la tâche et de supprimer ces utilisateurs lorsqu'ils ne sont plus nécessaires.

- Pour renforcer la sécurité, nous recommandons de créer des identités IAM utilisées dans tous les AWS services et d'utiliser des informations d'identification temporaires. Le plugin d'authentification Amazon Keyspaces SigV4 pour les pilotes clients Cassandra vous permet d'authentifier les appels vers Amazon Keyspaces à l'aide de clés d'accès IAM plutôt que de votre nom d'utilisateur et de votre mot de passe. Pour en savoir plus sur la façon dont le plugin Amazon Keyspaces SigV4 permet aux utilisateurs, aux rôles et aux identités fédérées d'IAM de s'authentifier dans les demandes d'API Amazon Keyspaces, AWS consultez le processus Signature Version 4 (SigV4).

  Vous pouvez télécharger les plug-ins SigV4 depuis les emplacements suivants.

  - Java :https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin.
  - Node.js :https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.
  - Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.
  - Allez :https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

  Pour des exemples de code montrant comment établir des connexions à l'aide du plug-in d'authentification SigV4, voirthe section called “Utilisation d'un pilote client Cassandra”.

Rubriques

- Générer des informations d'identification spécifiques au service
- Comment créer et configurer les AWS informations d'identification pour Amazon Keyspaces

Générer des informations d'identification spécifiques au service

Les informations d'identification spécifiques au service sont similaires au nom d'utilisateur et au mot de passe traditionnels que Cassandra utilise pour l'authentification et la gestion des accès. Les informations d'identification spécifiques au service permettent aux utilisateurs IAM d'accéder à un

service spécifique. AWS Ces informations d'identification à long terme ne peuvent pas être utilisées pour accéder à d'autres AWS services. Ils sont associés à un utilisateur IAM spécifique et ne peuvent pas être utilisés par d'autres utilisateurs IAM.

> **Important**
>
> Les informations d'identification spécifiques au service sont des informations d'identification à long terme associées à un utilisateur IAM spécifique et ne peuvent être utilisées que pour le service pour lequel elles ont été créées. Pour autoriser les rôles IAM ou les identités fédérées à accéder à toutes vos AWS ressources à l'aide d'informations d'identification temporaires, vous devez utiliser l'AWS authentification avec le plugin d'authentification SigV4 pour Amazon Keyspaces.

Utilisez l'une des procédures suivantes pour générer des informations d'identification spécifiques au service.

Générez des informations d'identification spécifiques au service à l'aide de la console

Pour générer des informations d'identification spécifiques au service à l'aide de la console

1. Connectez-vous à la AWS Identity and Access Management console AWS Management Console et ouvrez-la à l'adressehttps://console.aws.amazon.com/iam/home.
2. Dans le volet de navigation, choisissez Users, puis choisissez l'utilisateur que vous avez créé précédemment et qui dispose des autorisations Amazon Keyspaces (politique jointe).
3. Choisissez Informations d'identification de sécurité. Sous Credentials for Amazon Keyspaces, choisissez Generate credentials pour générer les identifiants spécifiques au service.

   Vos informations d'identification spécifiques au service sont disponibles. C'est la seule fois que vous pouvez télécharger ou afficher le mot de passe. Vous ne pourrez pas la récupérer plus tard. Cependant, vous pouvez réinitialiser votre mot de passe à tout moment. Enregistrez cet utilisateur et ce mot de passe dans un emplacement sécurisé, car vous en aurez besoin plus tard.

Générez des informations d'identification spécifiques au service à l'aide du AWS CLI

Pour générer des informations d'identification spécifiques au service à l'aide du AWS CLI

Avant de générer des informations d'identification spécifiques au service, vous devez télécharger, installer et configurer le AWS Command Line Interface ()AWS CLI :

1. Téléchargez-le AWS CLI à l'adresse http://aws.amazon.com/cli.

   > **Note**
   >
   > Il AWS CLI fonctionne sous Windows, macOS ou Linux.

2. Suivez les instructions d'installation de la AWS CLI et de configuration de la AWS CLI dans le guide de AWS Command Line Interface l'utilisateur.
3. À l'aide de AWS CLI, exécutez la commande suivante pour générer des informations d'identification spécifiques au service pour l'utilisateur`alice`, afin qu'elle puisse accéder à Amazon Keyspaces.

   ```
   aws iam create-service-specific-credential \
       --user-name alice \
       --service-name cassandra.amazonaws.com
   ```

Le résultat se présente comme suit.

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-10-09T16:12:04Z",
        "ServiceName": "cassandra.amazonaws.com",
        "ServiceUserName": "alice-at-111122223333",
        "ServicePassword": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "ServiceSpecificCredentialId": "ACCAYFI33SINPGJEBYESF",
        "UserName": "alice",
        "Status": "Active"
    }
}
```

Dans la sortie, notez les valeurs pour `ServiceUserName` et `ServicePassword`. Enregistrez ces valeurs dans un emplacement sécurisé, car vous en aurez besoin plus tard.

> **Important**
>
> C'est la seule fois que le `ServicePassword` sera disponible pour vous.

Comment créer et configurer les AWS informations d'identification pour Amazon Keyspaces

Pour accéder à Amazon Keyspaces par programmation avec le AWS SDK ou avec les AWS CLI pilotes clients Cassandra et le plugin SigV4, vous avez besoin d'un utilisateur ou d'un rôle IAM avec des clés d'accès. Lorsque vous utilisez AWS par programmation, vous fournissez vos clés AWS d'accès AWS afin de vérifier votre identité lors d'appels programmatiques. Vos clés d'accès se composent d'un identifiant de clé d'accès (par exemple, AKIAIOSFODNN7EXAMPLE) et d'une clé d'accès secrète (par exemple, WJALRXUTNFEMI/K7MDENG/CYEXAMPLEKEY). bPxRfi Cette rubrique décrit les étapes nécessaires à ce processus.

Les meilleures pratiques en matière de sécurité recommandent de créer des utilisateurs IAM dotés d'autorisations limitées et d'associer les rôles IAM aux autorisations nécessaires pour effectuer des tâches spécifiques. Les utilisateurs IAM peuvent ensuite assumer temporairement des rôles IAM pour effectuer les tâches requises. Par exemple, les utilisateurs IAM de votre compte utilisant la console Amazon Keyspaces peuvent passer à un rôle afin d'utiliser temporairement les autorisations du rôle dans la console. Les utilisateurs abandonnent leurs autorisations d'origine et acceptent les autorisations attribuées au rôle. Lorsque les utilisateurs quittent le rôle, leurs autorisations d'origine sont restaurées. Les informations d'identification utilisées par les utilisateurs pour assumer le rôle sont temporaires. Au contraire, les utilisateurs IAM disposent d'informations d'identification à long terme, ce qui présente un risque de sécurité si, au lieu d'assumer des rôles, des autorisations leur sont directement attribuées. Pour atténuer ce risque, nous vous recommandons de ne fournir à ces utilisateurs que les autorisations dont ils ont besoin pour effectuer la tâche et de supprimer ces utilisateurs lorsqu'ils ne sont plus nécessaires. Pour plus d'informations sur les rôles, consultez la section Scénarios courants pour les rôles : utilisateurs, applications et services dans le guide de l'utilisateur IAM.

Rubriques

- Informations d'identification requises par AWS CLI le AWS SDK ou le plugin Amazon Keyspaces SigV4 pour les pilotes clients Cassandra
- Création d'un utilisateur IAM pour un accès programmatique à Amazon Keyspaces dans votre compte AWS

- Création de nouvelles clés d'accès pour un utilisateur IAM
- Comment gérer les clés d'accès pour les utilisateurs IAM
- Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plug-in SigV4

Informations d'identification requises par AWS CLI le AWS SDK ou le plugin Amazon Keyspaces SigV4 pour les pilotes clients Cassandra

Les informations d'identification suivantes sont requises pour authentifier l'utilisateur ou le rôle IAM :

`AWS_ACCESS_KEY_ID`

Spécifie une clé AWS d'accès associée à un utilisateur ou à un rôle IAM.

La clé d'accès `aws_access_key_id` est requise pour se connecter à Amazon Keyspaces par programmation.

`AWS_SECRET_ACCESS_KEY`

Indique la clé secrète associée à la clé d'accès. Il s'agit du « mot de passe » de la clé d'accès.

`aws_secret_access_key`Il est nécessaire pour se connecter à Amazon Keyspaces par programmation.

`AWS_SESSION_TOKEN`— Facultatif

Spécifie la valeur du jeton de session requise si vous utilisez des informations d'identification de sécurité temporaires que vous avez extraites directement des opérations AWS Security Token Service . Pour plus d'informations, consultez the section called “Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces”.

Si vous vous connectez avec un utilisateur IAM, ce n'`aws_session_token`est pas obligatoire.

Création d'un utilisateur IAM pour un accès programmatique à Amazon Keyspaces dans votre compte AWS

Pour obtenir les informations d'identification nécessaires à l'accès programmatique à Amazon Keyspaces avec AWS CLI le plugin, AWS le SDK ou le plugin SigV4, vous devez d'abord créer un utilisateur ou un rôle IAM. Le processus de création d'un utilisateur IAM et de configuration de cet utilisateur IAM pour qu'il dispose d'un accès programmatique à Amazon Keyspaces est illustré dans les étapes suivantes :

1. Créez l'utilisateur dans AWS Management Console les AWS CLI outils pour Windows PowerShell ou à l'aide d'une opération d' AWS API. Si vous créez l'utilisateur dans le AWS Management Console, les informations d'identification sont créées automatiquement.
2. Si vous créez l'utilisateur par programmation, vous devez créer une clé d'accès (ID de clé d'accès et clé d'accès secrète) pour cet utilisateur lors d'une étape supplémentaire.
3. Donnez à l'utilisateur l'autorisation d'accéder à Amazon Keyspaces.

Pour plus d'informations sur les autorisations dont vous avez besoin pour créer un utilisateur, consultez la section Autorisations requises pour accéder aux ressources IAM.

Création d'utilisateurs IAM (console)

Vous pouvez utiliser le AWS Management Console pour créer des utilisateurs IAM.

Pour créer un utilisateur IAM doté d'un accès par programmation (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Users (Utilisateurs), puis Add users (Ajouter des utilisateurs).
3. Tapez le nom d'utilisateur du nouvel utilisateur. Il s'agit du nom de connexion pour AWS.

   > **Note**
   >
   > Les noms d'utilisateur peuvent combiner jusqu'à 64 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et tiret (-). Les noms doivent être uniques dans un compte. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux utilisateurs nommés TESTUSER et testuser.

4. Sélectionnez Clé d'accès - Accès par programmation pour créer une clé d'accès pour le nouvel utilisateur. Vous pouvez consulter ou télécharger la clé d'accès lorsque vous arrivez sur la page finale.

   Sélectionnez Next: Permissions (Étape suivante : autorisations).
5. Sur la page Définir les autorisations, choisissez Joindre directement les politiques existantes pour attribuer des autorisations au nouvel utilisateur.

Cette option affiche la liste des politiques AWS gérées et gérées par le client disponibles dans votre compte. Vous pouvez `keyspaces` entrer dans le champ de recherche pour afficher uniquement les politiques relatives à Amazon Keyspaces.

Pour Amazon Keyspaces, les politiques gérées disponibles sont `AmazonKeyspacesFullAccess` et. `AmazonKeyspacesReadOnlyAccess` Pour plus d'informations sur chaque politique, consultezthe section called “Politiques gérées par AWS”.

À des fins de test et pour suivre les didacticiels de connexion, sélectionnez la `AmazonKeyspacesReadOnlyAccess` politique pour le nouvel utilisateur IAM. Remarque : En tant que bonne pratique, nous vous recommandons de suivre le principe du moindre privilège et de créer des politiques personnalisées qui limitent l'accès à des ressources spécifiques et n'autorisent que les actions requises. Pour plus d'informations sur les politiques IAM et pour consulter des exemples de politiques pour Amazon Keyspaces, consultez. the section called “Politiques basées sur l'identité d'Amazon Keyspaces” Après avoir créé des politiques d'autorisation personnalisées, associez vos politiques aux rôles, puis laissez les utilisateurs assumer temporairement les rôles appropriés.

Choisissez Suivant : Balises.

6. Sur la page Ajouter des balises (facultatif), vous pouvez ajouter des balises pour l'utilisateur ou choisir Suivant : Réviser.
7. Sur la page Révision, vous pouvez voir tous les choix que vous avez faits jusqu'à présent. Lorsque vous êtes prêt à continuer, choisissez Create user.
8. Pour afficher les clés d'accès de l'utilisateur (ID de clé d'accès et clés d'accès secrètes), choisissez Afficher en regard du mot de passe et de la clé d'accès. Pour enregistrer les clés d'accès, choisissez télécharger .csv, puis enregistrez le fichier dans un emplacement sûr sur votre ordinateur.

> **⚠ Important**
>
> Il s'agit de votre seule opportunité de visualiser ou de télécharger les clés d'accès secrètes, et vous avez besoin de ces informations pour qu'ils puissent utiliser le plugin SigV4. Enregistrez les nouveaux ID de clé d'accès et clé d'accès secrète de l'utilisateur dans un endroit sûr et sécurisé. Vous ne pourrez plus accéder aux clés d'accès secrètes après cette étape.

Création d'utilisateurs IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour créer un utilisateur IAM.

Pour créer un utilisateur IAM avec accès par programmation ()AWS CLI

1. Créez un utilisateur avec le AWS CLI code suivant.
   - aws iam create-user
2. Donnez à l'utilisateur un accès programmatique. Cela nécessite des clés d'accès, qui peuvent être générées de différentes manières.
   - AWS CLI: aws iam create-access-key
   - Outils pour Windows PowerShell : New-IAMAccessKey
   - API IAM : CreateAccessKey

   > **Important**
   >
   > Il s'agit de votre seule opportunité de visualiser ou de télécharger les clés d'accès secrètes, et vous avez besoin de ces informations pour qu'ils puissent utiliser le plugin SigV4. Enregistrez les nouveaux ID de clé d'accès et clé d'accès secrète de l'utilisateur dans un endroit sûr et sécurisé. Vous ne pourrez plus accéder aux clés d'accès secrètes après cette étape.
3. Attachez à l'utilisateur la `AmazonKeyspacesReadOnlyAccess` politique qui définit les autorisations de l'utilisateur. Remarque : il est recommandé de gérer les autorisations des utilisateurs en ajoutant l'utilisateur à un groupe et en attachant une politique au groupe au lieu de l'associer directement à un utilisateur.
   - AWS CLI: aws iam attach-user-policy

Création de nouvelles clés d'accès pour un utilisateur IAM

Si vous avez déjà un utilisateur IAM, vous pouvez créer de nouvelles clés d'accès à tout moment. Pour plus d'informations sur la gestion des clés, par exemple comment faire pivoter les clés d'accès, consultez la section Gestion des clés d'accès pour les utilisateurs IAM.

Pour créer des clés d'accès pour un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez créer les clés d'accès.
4. Sur la page Résumé de l'utilisateur, choisissez l'onglet Informations d'identification de sécurité.
5. Dans la section Access keys (Clés d'accès), choisissez Create access key (Créer une clé d'accès).

   Pour afficher la nouvelle clé d'accès, choisissez Show (Afficher). Vos informations d'identification s'afficheront comme suit :

   - ID de clé d'accès : AKIAIOSFODNN7EXAMPLE
   - Clé d'accès secrète : bPxRfi WJALRxUTNFEMI/K7MDENG/CYEXAMPLEKEY

   > **Note**
   >
   > Vous ne pourrez pas accéder à la clé d’accès secrète à nouveau une fois que cette boîte de dialogue se sera fermée.

6. Pour télécharger la paire de clés, choisissez Télécharger le fichier .csv. Stockez les clés dans un emplacement sûr.
7. Après avoir téléchargé le fichier CSV, sélectionnez Close (Fermer).

Lorsque vous créez une clé d'accès, la paire de clés est activée par défaut et vous pouvez utiliser la paire immédiatement.

Comment gérer les clés d'accès pour les utilisateurs IAM

Il est recommandé de ne pas intégrer les clés d'accès directement dans le code. Les AWS SDK et les outils de ligne de AWS commande vous permettent de placer les clés d'accès dans des emplacements connus afin de ne pas avoir à les conserver dans le code. Placez les clés d'accès à l'un des emplacements suivants :

- Variables d'environnement : sur un système mutualisé, choisissez les variables d'environnement utilisateur et non les variables d'environnement système.

- Fichier d'informations d'identification CLI — Le `config` fichier `credentials` et est mis à jour lorsque vous exécutez la commande`aws configure`. Le `credentials` fichier se trouve sous `~/.aws/credentials` Linux, macOS ou Unix, ou `C:\Users\`*`USERNAME`*`\.aws\credentials` sous Windows. Ce fichier contient les informations d'identification du profil `default` et de tous les profils nommés.
- Fichier de configuration CLI — Le `config` fichier `credentials` et est mis à jour lorsque vous exécutez la commande`aws configure`. Le `config` fichier se trouve sous `~/.aws/config` Linux, macOS ou Unix, ou `C:\Users\`*`USERNAME`*`\.aws\config` sous Windows. Ce fichier contient les paramètres de configuration du profil par défaut et des profils nommés.

Le stockage des clés d'accès en tant que variables d'environnement est une condition préalable au. the section called “Plugin d'authentification pour Java 4.x” Le client recherche les informations d'identification à l'aide de la chaîne de fournisseurs d'informations d'identification par défaut, et les clés d'accès stockées sous forme de variables d'environnement ont priorité sur tous les autres emplacements, par exemple les fichiers de configuration. Pour plus d'informations, consultez la section Paramètres de configuration et priorité.

Les exemples suivants montrent comment vous pouvez configurer des variables d'environnement pour l'utilisateur par défaut.

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of security token>
```

La définition de la variable d'environnement permet de modifier la valeur utilisée jusqu'à la fin de votre session shell, ou jusqu'à ce que vous définissiez la variable sur une autre valeur. Vous pouvez rendre les variables persistantes dans de futures sessions en les définissant dans votre script de démarrage de shell.

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> setx AWS_SESSION_TOKEN AQoDYXdzEJr...<remainder of security token>
```

L'utilisation de `set` pour définir une variable d'environnement modifie la valeur utilisée jusqu'à la fin de la session d'invite de commande en cours, ou jusqu'à ce que vous définissiez la variable

sur une autre valeur. L'utilisation de `setx` pour définir une variable d'environnement modifie la valeur utilisée dans la session d'invite de commande en cours et toutes les sessions d'invite de commande que vous créez après l'exécution de la commande. Cela n'affecte pas les autres shells de commande qui sont déjà en cours d'exécution lorsque vous exécutez la commande.

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
PS C:\> $Env:AWS_SESSION_TOKEN="AQoDYXdzEJr...<remainder of security token>"
```

Si vous définissez une variable d'environnement à l' PowerShell invite, comme indiqué dans les exemples précédents, elle enregistre la valeur uniquement pendant la durée de la session en cours. Pour que le paramètre de variable d'environnement soit persistant dans toutes les sessions PowerShell et dans les sessions d'invite de commande, stockez-le à l'aide de l'application système du Panneau de configuration. Vous pouvez également définir la variable pour toutes les PowerShell sessions futures en l'ajoutant à votre PowerShell profil. Consultez la PowerShell documentation pour plus d'informations sur le stockage des variables d'environnement ou leur persistance d'une session à l'autre.

Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plug-in SigV4

Pour améliorer la sécurité, vous pouvez utiliser des informations d'identification temporaires pour vous authentifier avec le plug-in SigV4. Dans de nombreux cas, vous n'avez pas besoin d'une clé d'accès à long terme qui n'expire jamais (comme dans le cas d'un utilisateur IAM). Vous pouvez plutôt créer un rôle IAM et générer des informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires consistent en un ID de clé d'accès et une clé d'accès secrète, mais elles comprennent également un jeton de sécurité qui indique la date d'expiration des informations d'identification. Pour en savoir plus sur l'utilisation des rôles IAM plutôt que des clés d'accès à long terme, consultez la section Passer à un rôle IAM (AWS API).

Pour commencer à utiliser des informations d'identification temporaires, vous devez d'abord créer un rôle IAM.

Créez un rôle IAM qui accorde un accès en lecture seule à Amazon Keyspaces

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le volet de navigation, choisissez Rôles, puis Créer un rôle.
3. Sur la page Créer un rôle, sous Sélectionner le type d'entité de confiance, sélectionnez AWS service. Sous Choisir un cas d'utilisation, choisissez Amazon EC2, puis Next.
4. Sur la page Ajouter des autorisations, sous Politiques d'autorisations, choisissez Amazon Keyspaces Read Only Access dans la liste des politiques, puis choisissez Next.
5. Sur la page Nom, révision et création, entrez le nom du rôle et consultez les sections Sélectionner des entités fiables et Ajouter des autorisations. Vous pouvez également ajouter des balises facultatives pour le rôle sur cette page. Lorsque vous avez terminé, sélectionnez Créer un rôle. N'oubliez pas ce nom car vous en aurez besoin lorsque vous lancerez votre instance Amazon EC2.

Pour utiliser des informations d'identification de sécurité temporaires dans le code, vous devez appeler par programmation une AWS Security Token Service API similaire `AssumeRole` et extraire les informations d'identification et le jeton de session qui en résultent de votre rôle IAM que vous avez créé à l'étape précédente. Vous utilisez ensuite ces valeurs comme informations d'identification pour les appels suivants à AWS. L'exemple suivant montre un pseudocode expliquant comment utiliser les informations d'identification de sécurité temporaires :

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
cassandraRequest = CreateAmazoncassandraClient(tempCredentials);
```

Pour un exemple qui implémente des informations d'identification temporaires à l'aide du pilote Python pour accéder à Amazon Keyspaces, consultez. ???

Pour plus d'informations sur la façon d'appeler `AssumeRole`, `GetFederationToken` et d'autres opérations d'API, consultez la Référence sur l'API AWS Security Token Service. Pour plus d'informations sur l'obtention des informations d'identification de sécurité temporaires et du jeton de session à partir des résultats, consultez la documentation du kit SDK que vous utilisez. Vous trouverez la documentation de tous les AWS SDK sur la page de AWS documentation principale, dans la section SDK et boîtes à outils.

Points de terminaison de service pour Amazon Keyspaces

Rubriques

- Ports et protocoles
- Points de terminaison mondiaux
- AWS GovCloud (US) RegionPoints de terminaison FIPS
- Points de terminaison des régions chinoises

Ports et protocoles

Vous pouvez accéder à Amazon Keyspaces par programmation en exécutant un `cqlsh` client, avec un pilote Cassandra sous licence Apache 2.0, ou en utilisant le SDK et le SDK. AWS CLI AWS

Le tableau suivant indique les ports et les protocoles des différents mécanismes d'accès.

Accès programmatique	Port	Protocole
CQLSH	9142	TLS
Pilote Cassandra	9142	TLS
AWS CLI	443	HTTPS
AWS SDK	443	HTTPS

Pour les connexions TLS, Amazon Keyspaces utilise l'autorité de certification Starfield pour s'authentifier auprès du serveur. Pour plus d'informations, consultez the section called “Comment configurer manuellement les `cqlsh` connexions pour le protocole TLS” la section « Avant de commencer » de votre pilote dans le the section called “Utilisation d'un pilote client Cassandra” chapitre.

Points de terminaison mondiaux

Amazon Keyspaces est disponible dans les versions suivantes. Régions AWS Ce tableau indique le point de terminaison de service disponible pour chaque région.

Nom de la région	Région	Point de terminaison	Protocole
US East (Ohio)	us-east-2	cassandra.us-east-2.amazonaws.com	HTTPS et TLS
US East (N. Virginia)	us-east-1	cassandra.us-east-1.amazonaws.com cassandra-fips.us-east-1.amazonaws.com	HTTPS et TLS TLS
USA Ouest (Californie du Nord)	us-west-1	cassandra.us-west-1.amazonaws.com	HTTPS et TLS
US West (Oregon)	us-west-2	cassandra.us-west-2.amazonaws.com cassandra-fips.us-west-2.amazonaws.com	HTTPS et TLS TLS
Asie-Pacifique (Hong Kong)	ap-east-1	cassandra.ap-east-1.amazonaws.com	HTTPS et TLS
Asia Pacific (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com	HTTPS et TLS
Asia Pacific (Seoul)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com	HTTPS et TLS
Asia Pacific (Singapore)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com	HTTPS et TLS

Nom de la région	Région	Point de terminaison	Protocole	
Asia Pacific (Sydney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com	HTTPS et TLS	
Asia Pacific (Tokyo)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com	HTTPS et TLS	
Canada (Central)	ca-central-1	cassandra.ca-central-1.amazonaws.com	HTTPS et TLS	
Europe (Frankfurt)	eu-central-1	cassandra.eu-central-1.amazonaws.com	HTTPS et TLS	
Europe (Ireland)	eu-west-1	cassandra.eu-west-1.amazonaws.com	HTTPS et TLS	
Europe (London)	eu-west-2	cassandra.eu-west-2.amazonaws.com	HTTPS et TLS	
Europe (Paris)	eu-west-3	cassandra.eu-west-3.amazonaws.com	HTTPS et TLS	
Europe (Stockholm)	eu-north-1	cassandra.eu-north-1.amazonaws.com	HTTPS et TLS	
Moyen-Orient (Bahreïn)	me-south-1	cassandra.me-south-1.amazonaws.com	HTTPS et TLS	
South America (São Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com	HTTPS et TLS	

Nom de la région	Région	Point de terminaison	Protocole	
AWS GovCloud (USA Est)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	HTTPS et TLS	
AWS GovCloud (US-Ouest)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	HTTPS et TLS	

AWS GovCloud (US) RegionPoints de terminaison FIPS

Points de terminaison FIPS disponibles dans le. AWS GovCloud (US) Region Pour plus d'informations, consultez Amazon Keyspaces dans le guide de l'AWS GovCloud (US)utilisateur.

Nom de la région	Région	Point de terminaison FIPS	Protocole
AWS GovCloud (USA Est)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	HTTPS et TLS
AWS GovCloud (US-Ouest)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	HTTPS et TLS

Points de terminaison des régions chinoises

Les points de terminaison Amazon Keyspaces suivants sont disponibles dans les régions de AWS Chine.

Pour accéder à ces points de terminaison, vous devez vous inscrire pour obtenir un ensemble distinct d'informations d'identification propres aux régions de Chine. Pour plus d'informations, consultez la section Inscription, comptes et informations d'identification en Chine.

Nom de la région	Région	Point de terminaison	Protocole
Chine (Beijing)	cn-north-1	cassandra.cn-north-1.amazonaws.com .cn	HTTPS et TLS
Chine (Ningxia)	cn-northwest-1	cassandra.cn-northwest-1.amazonaws.com .cn	HTTPS et TLS

Utilisation **`cqlsh`** pour se connecter à Amazon Keyspaces

Pour vous connecter à Amazon Keyspaces à l'aide de`cqlsh`, vous pouvez utiliser le. `cqlsh-expansion` Il s'agit d'une boîte à outils qui contient des outils Apache Cassandra courants `cqlsh` et des aides préconfigurés pour Amazon Keyspaces tout en maintenant une compatibilité totale avec Apache Cassandra. Il `cqlsh-expansion` intègre le plugin d'authentification SigV4 et vous permet de vous connecter à l'aide de clés d'accès IAM au lieu du nom d'utilisateur et du mot de passe. Vous devez uniquement installer les `cqlsh` scripts pour établir une connexion, et non la distribution complète d'Apache Cassandra, car Amazon Keyspaces fonctionne sans serveur. Ce package d'installation léger inclut les `cqlsh` scripts classiques `cqlsh-expansion` et les scripts que vous pouvez installer sur n'importe quelle plateforme supportant Python.

Pour des informations générales sur`cqlsh`, voir `cqlsh: le shell CQL.`

Rubriques

- Utilisation du cqlsh-expansion pour se connecter à Amazon Keyspaces
- Comment configurer manuellement les cqlsh connexions pour le protocole TLS

Utilisation du **`cqlsh-expansion`** pour se connecter à Amazon Keyspaces

Installation et configuration du **`cqlsh-expansion`**

1. Pour installer le package `cqlsh-expansion` Python, vous pouvez exécuter une `pip` commande. Cela installe les `cqlsh-expansion` scripts sur votre machine à l'aide d'une installation pip avec un fichier contenant une liste de dépendances. Il `--user flag` indique `pip` d'utiliser le répertoire d'installation utilisateur Python pour votre plateforme. Sur un système basé sur Unix, ce devrait être le `~/.local/` répertoire.

Vous avez besoin de Python 3 pour installer le`cqlsh-expansion`, pour connaître votre version de Python, utilisez`Python --version`. Pour l'installer, vous pouvez exécuter la commande suivante.

```
python3 -m pip install --user cqlsh-expansion
```

La sortie doit ressembler à ceci.

```
Collecting cqlsh-expansion
  Downloading cqlsh_expansion-0.9.6-py3-none-any.whl (153 kB)
     ######################################## 153.7/153.7 KB 3.3 MB/s eta 0:00:00
Collecting cassandra-driver
  Downloading cassandra_driver-3.28.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (19.1 MB)
     ######################################## 19.1/19.1 MB 44.5 MB/s eta 0:00:00
Requirement already satisfied: six>=1.12.0 in /usr/lib/python3/dist-packages (from
 cqlsh-expansion) (1.16.0)
Collecting boto3
  Downloading boto3-1.29.2-py3-none-any.whl (135 kB)
     ######################################## 135.8/135.8 KB 17.2 MB/s eta 0:00:00
Collecting cassandra-sigv4>=4.0.2
  Downloading cassandra_sigv4-4.0.2-py2.py3-none-any.whl (9.8 kB)
Collecting botocore<1.33.0,>=1.32.2
  Downloading botocore-1.32.2-py3-none-any.whl (11.4 MB)
     ######################################## 11.4/11.4 MB 60.9 MB/s eta 0:00:00
Collecting s3transfer<0.8.0,>=0.7.0
  Downloading s3transfer-0.7.0-py3-none-any.whl (79 kB)
     ######################################## 79.8/79.8 KB 13.1 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting geomet<0.3,>=0.1
  Downloading geomet-0.2.1.post1-py3-none-any.whl (18 kB)
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ######################################## 247.7/247.7 KB 33.1 MB/s eta 0:00:00
Requirement already satisfied: urllib3<2.1,>=1.25.4 in /usr/lib/python3/dist-
packages (from botocore<1.33.0,>=1.32.2->boto3->cqlsh-expansion) (1.26.5)
Requirement already satisfied: click in /usr/lib/python3/dist-packages (from
 geomet<0.3,>=0.1->cassandra-driver->cqlsh-expansion) (8.0.3)
Installing collected packages: python-dateutil, jmespath, geomet, cassandra-driver,
 botocore, s3transfer, boto3, cassandra-sigv4, cqlsh-expansion
```

```
  WARNING: The script geomet is installed in '/home/ubuntu/.local/bin' which is not
 on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this
 warning, use --no-warn-script-location.
  WARNING: The scripts cqlsh, cqlsh-expansion and cqlsh-expansion.init are
 installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this
 warning, use --no-warn-script-location.
Successfully installed boto3-1.29.2 botocore-1.32.2 cassandra-driver-3.28.0
 cassandra-sigv4-4.0.2 cqlsh-expansion-0.9.6 geomet-0.2.1.post1 jmespath-1.0.1
 python-dateutil-2.8.2 s3transfer-0.7.0
```

Si le répertoire d'installation ne se trouve pas dans lePATH, vous devez l'ajouter en suivant les instructions de votre système d'exploitation. Vous trouverez ci-dessous un exemple pour Ubuntu Linux.

```
export PATH="$PATH:/home/ubuntu/.local/bin"
```

Pour confirmer que le package est installé, vous pouvez exécuter la commande suivante.

```
cqlsh-expansion --version
```

La sortie doit ressembler à ceci.

```
cqlsh 6.1.0
```

2. Pour configurer le`cqlsh-expansion`, vous pouvez exécuter un script de post-installation pour effectuer automatiquement les étapes suivantes :

   1. Créez le `.cassandra` répertoire dans le répertoire personnel de l'utilisateur s'il n'existe pas déjà.
   2. Copiez un fichier de `cqlshrc` configuration préconfiguré dans le `.cassandra` répertoire.
   3. Copiez le certificat numérique Starfield dans le `.cassandra` répertoire. Amazon Keyspaces utilise ce certificat pour configurer la connexion sécurisée avec le protocole TLS (Transport Layer Security). Le chiffrement en transit fournit une couche supplémentaire de protection des données en chiffrant vos données lorsqu'elles sont acheminées vers et depuis Amazon Keyspaces.

Pour consulter d'abord le script, vous pouvez y accéder dans le dépôt Github à l'adresse. post_install.py

Pour utiliser le script, vous pouvez exécuter la commande suivante.

```
cqlsh-expansion.init
```

> **Note**
>
> Le répertoire et le fichier créés par le script de post-installation ne sont pas supprimés lorsque vous `cqlsh-expansion` désinstallez `pip uninstall` l'utilisateur et doivent être supprimés manuellement.

Connexion à Amazon Keyspaces à l'aide du **`cqlsh-expansion`**

1. Configurez votre variable d'environnement Région AWS et ajoutez-la en tant que variable d'environnement utilisateur.

   Pour ajouter votre région par défaut en tant que variable d'environnement sur un système basé sur Unix, vous pouvez exécuter la commande suivante. Pour cet exemple, nous utilisons l'est des États-Unis (Virginie du Nord).

   ```
   export AWS_DEFAULT_REGION=us-east-1
   ```

   Pour plus d'informations sur la définition des variables d'environnement, y compris pour d'autres plateformes, consultez Comment définir des variables d'environnement.

2. Trouvez le point de terminaison de votre service.

   Choisissez le point de terminaison de service approprié pour votre région. Pour consulter les points de terminaison disponibles pour Amazon Keyspaces, consultez. the section called “Points de terminaison de service” Pour cet exemple, nous utilisons le point de terminaison`cassandra.us-east-1.amazonaws.com`.

3. Configurez la méthode d'authentification.

La connexion à l'aide de clés d'accès IAM (utilisateurs IAM, rôles et identités fédérées) est la méthode recommandée pour renforcer la sécurité.

Avant de pouvoir vous connecter à l'aide des clés d'accès IAM, vous devez suivre les étapes suivantes :

a. Créez un utilisateur IAM ou suivez les meilleures pratiques et créez un rôle IAM que les utilisateurs IAM peuvent assumer. Pour plus d'informations sur la création de clés d'accès IAM, consultezthe section called “Informations d'identification IAM pour l' AWS authentification”.

b. Créez une politique IAM qui accorde au rôle (ou à l'utilisateur IAM) au moins un accès en lecture seule à Amazon Keyspaces. Pour plus d'informations sur les autorisations requises pour que l'utilisateur ou le rôle IAM se connecte à Amazon Keyspaces, consultez. the section called “Accès aux tables Amazon Keyspaces”

c. Ajoutez les clés d'accès de l'utilisateur IAM aux variables d'environnement de l'utilisateur, comme indiqué dans l'exemple suivant.

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Pour plus d'informations sur la définition des variables d'environnement, y compris pour d'autres plateformes, consultez Comment définir des variables d'environnement.

> **Note**
>
> Si vous vous connectez depuis une instance Amazon EC2, vous devez également configurer une règle sortante dans le groupe de sécurité qui autorise le trafic depuis l'instance vers Amazon Keyspaces. Pour plus d'informations sur la façon d'afficher et de modifier les règles de sortie EC2, consultez la section Ajouter des règles à un groupe de sécurité dans le guide de l'utilisateur Amazon EC2.

4. Connectez-vous à Amazon Keyspaces à l'aide de l'authentification `cqlsh-expansion` et SigV4.

Pour vous connecter à Amazon Keyspaces avec le`cqlsh-expansion`, vous pouvez utiliser la commande suivante. Assurez-vous de remplacer le point de terminaison du service par le point de terminaison adapté à votre région.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

Si la connexion est établie, vous devriez obtenir un résultat similaire à celui de l'exemple suivant.

```
Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
[cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh current consistency level is ONE.
cqlsh>
```

Si vous rencontrez une erreur de connexion, consultez the section called “Erreurs de connexion Cqlsh” les informations de dépannage.

- Connectez-vous à Amazon Keyspaces avec des informations d'identification spécifiques au service.

  Pour vous connecter à la combinaison traditionnelle de nom d'utilisateur et de mot de passe utilisée par Cassandra pour l'authentification, vous devez d'abord créer des informations d'identification spécifiques au service pour Amazon Keyspaces, comme décrit dans. the section called “Informations d'identification spécifiques au service” Vous devez également autoriser cet utilisateur à accéder à Amazon Keyspaces. Pour plus d'informations, consultez. the section called “Accès aux tables Amazon Keyspaces”

  Après avoir créé des informations d'identification et des autorisations spécifiques au service pour l'utilisateur, vous devez mettre à jour le `cqlshrc` fichier, qui se trouve généralement dans le chemin du répertoire utilisateur. `~/.cassandra/` Dans le `cqlshrc` fichier, accédez à la `[authentication]` section Cassandra et commentez le module et la classe SigV4 `[auth_provider]` en utilisant le caractère « ; » comme indiqué dans l'exemple suivant.

  ```
  [auth_provider]

  ; module = cassandra_sigv4.auth

  ; classname = SigV4AuthProvider
  ```

  Après avoir mis à jour le `cqlshrc` fichier, vous pouvez vous connecter à Amazon Keyspaces avec des informations d'identification spécifiques au service à l'aide de la commande suivante.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 -u myUserName -
p myPassword --ssl
```

Nettoyage

- Pour supprimer le `cqlsh-expansion` package, vous pouvez utiliser la `pip uninstall` commande.

  ```
  pip3 uninstall cqlsh-expansion
  ```

  La `pip3 uninstall` commande ne supprime pas le répertoire et les fichiers associés créés par le script de post-installation. Pour supprimer le dossier et les fichiers créés par le script de post-installation, vous pouvez supprimer le `.cassandra` répertoire.

Comment configurer manuellement les **`cqlsh`** connexions pour le protocole TLS

Amazon Keyspaces accepte uniquement les connexions sécurisées utilisant le protocole TLS (Transport Layer Security). Vous pouvez utiliser l'`cqlsh-expansion`utilitaire qui télécharge automatiquement le certificat pour vous et installe un fichier de `cqlshrc` configuration préconfiguré. Pour plus d'informations, consultez the section called “Utilisation de l’`cqlsh-expansion`” cette page.

Si vous souhaitez télécharger le certificat et configurer la connexion manuellement, procédez comme suit.

1. Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

   ```
   curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
   ```

   > **Note**
   >
   > Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à

Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.

2. Ouvrez le fichier `cqlshrc` de configuration dans le répertoire de base de Cassandra, par exemple, `${HOME}/.cassandra/cqlshrc` et ajoutez les lignes suivantes.

```
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile =  path_to_file/sf-class2-root.crt
```

Utilisation des AWS CLI

Vous pouvez utiliser la AWS Command Line Interface (AWS CLI) pour contrôler plusieurs services AWS à partir de la ligne de commande et les automatiser à l'aide de scripts. Avec Amazon Keyspaces, vous pouvez utiliser les opérations DDL (AWS CLIfor Data Definition Language), telles que la création d'une table. En outre, vous pouvez utiliser des services et outils IaC (Infrastructure en tant que code) tels queAWS CloudFormation Terraform.

Pour pouvoir utiliser l'AWS CLIavec Amazon Keyspaces, vous devez obtenir un ID de clé d'accès et une clé d'accès secrète. Pour plus d'informations, veuillez consulter the section called “Informations d'identification IAM pour l' AWS authentification”.

Pour obtenir une liste complète de toutes les commandes disponibles pour Amazon Keyspaces dans l'AWS CLI, consultez la référence desAWS CLI commandes.

Rubriques

- Téléchargement et configuration de l'AWS CLI
- Utilisation de l'AWS CLIavec Amazon Keyspaces

Téléchargement et configuration de l'AWS CLI

LeAWS CLI est disponible à l'adressehttps://aws.amazon.com/cli. Elle s'exécute sous Windows, macOS ou Linux. Après avoir téléchargé leAWS CLI, procédez comme suit pour l'installer et le configurer :

1. Accédez au guide deAWS Command Line Interface l'utilisateur
2. Suivez les instructions d'installationAWS CLI et de configuration duAWS CLI

Utilisation de l'AWS CLIavec Amazon Keyspaces

Le format de ligne de commande se compose d'un nom d'opération Amazon Keyspaces, suivi des paramètres pour cette opération. l'AWS CLI prend en charge une version raccourcie de la syntaxe pour les valeurs des paramètres, ainsi que JSON. Les exemples Amazon Keyspaces suivants utilisent une syntaxeAWS CLI abrégée. Pour en savoir plus, veuillez consulter Utilisation de la syntaxe abrégée avec l'AWSCLI.

La commande suivante crée un espace de touches avec le catalogue de noms.

```
aws keyspaces create-keyspace --keyspace-name 'catalog'
```

La commande renvoie l'Amazon Resource Name (ARN) de la ressource en sortie.

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/"
}
```

Pour vérifier que le catalogue de keyspaces existe, vous pouvez utiliser la commande suivante.

```
aws keyspaces get-keyspace --keyspace-name 'catalog'
```

La sortie de la commande renvoie les valeurs suivantes.

```
{
    "keyspaceName": "catalog",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/"
}
```

La commande suivante crée une table nommée book_awards. La clé de partition de la table se compose des colonnes`yearaward` et la clé de regroupement comprend les colonnes`category`. Les deux colonnes de regroupement utilisent l'ordre de tri croissant.`rank` (Pour une lecture plus facile, les commandes longues dans cette section sont divisées en plusieurs lignes).

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'book_awards'
```

```
            --schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},
            {name=category,type=text}, {name=author,type=text},
{name=book_title,type=text},{name=publisher,type=text}],
            partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},{name=rank,orderBy=ASC}]'
```

Cette commande génèrera le résultat suivant.

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/table/
book_awards"
}
```

Pour confirmer les métadonnées et propriétés de la table, vous pouvez utiliser la commande suivante.

```
aws keyspaces get-table --keyspace-name 'catalog' --table-name 'book_awards'
```

Cette commande renvoie la sortie suivante.

```
{
    "keyspaceName": "catalog",
    "tableName": "book_awards",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/table/
book_awards",
    "creationTimestamp": 1645564368.628,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "year",
                "type": "int"
            },
            {
                "name": "award",
                "type": "text"
            },
            {
                "name": "category",
                "type": "text"
            },
```

```
            {
                "name": "rank",
                "type": "int"
            },
            {
                "name": "author",
                "type": "text"
            },
            {
                "name": "book_title",
                "type": "text"
            },
            {
                "name": "publisher",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "year"
            },
            {
                "name": "award"
            }
        ],
        "clusteringKeys": [
            {
                "name": "category",
                "orderBy": "ASC"
            },
            {
                "name": "rank",
                "orderBy": "ASC"
            }
        ],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1645564368.628
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
```

```
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Lorsque vous créez des tables avec des schémas complexes, il peut être utile de charger la définition du schéma de la table à partir d'un fichier JSON. En voici un exemple. Téléchargez l'exemple de fichier JSON de définition de schéma à partir du fichier schema_definition.zip etschema_definition.json extrayez-le en prenant note du chemin d'accès au fichier. Dans cet exemple, le fichier JSON de définition de schéma se trouve dans le répertoire actuelNT. Pour connaître les différentes options de chemin de fichier, voir Comment charger des paramètres à partir d'un fichier.

```
aws keyspaces create-table --keyspace-name 'catalog'
            --table-name 'book_awards' --schema-definition 'file://
schema_definition.json'
```

Les exemples suivants montrent comment créer un tableau simple portant le nom MyTable avec des options supplémentaires. Notez que les commandes sont réparties en lignes distinctes pour améliorer la lisibilité. Cette commande indique comment créer une table et :

- définir le mode de capacité de la table
- activer laoint-in-time restauration IP pour la table
- définir la valeur de durée de vie (TTL) par défaut pour le tableau à un an
- ajouter deux balises pour le tableau

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --capacity-specification
 'throughputMode=PROVISIONED,readCapacityUnits=5,writeCapacityUnits=5'
            --point-in-time-recovery 'status=ENABLED'
```

```
            --default-time-to-live '31536000'
            --tags 'key=env,value=test' 'key=dpt,value=sec'
```

Cet exemple montre comment créer une nouvelle table utilisant une clé gérée par le client pour le chiffrement et dont le TTL est activé pour vous permettre de définir des dates d'expiration pour les colonnes et les lignes. Pour exécuter cet exemple, vous devez remplacer l'ARN de ressource pour laAWS KMS clé gérée par le client par votre propre clé et vous assurer qu'Amazon Keyspaces y a accès.

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --encryption-specification
 'type=CUSTOMER_MANAGED_KMS_KEY,kmsKeyIdentifier=arn:aws:kms:us-
east-1:111222333444:key/11111111-2222-3333-4444-555555555555'
            --ttl 'status=ENABLED'
```

Utilisation de l'API

Vous pouvez utiliser le AWS SDK et le AWS Command Line Interface (AWS CLI) pour travailler de manière interactive avec Amazon Keyspaces. Vous pouvez utiliser l'API pour les opérations de définition du langage de données (DDL), telles que la création d'un espace de touches ou d'un tableau. En outre, vous pouvez utiliser des services et des outils d'infrastructure sous forme de code (IaC) tels que AWS CloudFormation Terraform.

Avant de pouvoir utiliser le AWS CLI avec Amazon Keyspaces, vous devez obtenir un identifiant de clé d'accès et une clé d'accès secrète. Pour plus d'informations, veuillez consulter the section called “Informations d'identification IAM pour l' AWS authentification”.

Pour obtenir la liste complète de toutes les opérations disponibles pour Amazon Keyspaces dans l'API, consultez la référence de l'API Amazon Keyspaces.

Utilisation d'Amazon Keyspaces avec un SDK AWS

AWS des kits de développement logiciel (SDK) sont disponibles pour de nombreux langages de programmation populaires. Chaque SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d’applications par les développeurs dans leur langage préféré.

Documentation SDK	Exemples de code
AWS SDK for C++	AWS SDK for C++ exemples de code
AWS CLI	AWS CLI exemples de code
AWS SDK for Go	AWS SDK for Go exemples de code
AWS SDK for Java	AWS SDK for Java exemples de code
AWS SDK for JavaScript	AWS SDK for JavaScript exemples de code
Kit AWS SDK pour Kotlin	Kit AWS SDK pour Kotlin exemples de code
AWS SDK for .NET	AWS SDK for .NET exemples de code
AWS SDK for PHP	AWS SDK for PHP exemples de code
AWS Tools for PowerShell	Outils pour des exemples PowerShell de code
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) exemples de code
AWS SDK for Ruby	AWS SDK for Ruby exemples de code
Kit AWS SDK pour Rust	Kit AWS SDK pour Rust exemples de code
AWS SDK pour SAP ABAP	AWS SDK pour SAP ABAP exemples de code
Kit AWS SDK pour Swift	Kit AWS SDK pour Swift exemples de code

Exemple de disponibilité

Vous n'avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Provide feedback (Fournir un commentaire) en bas de cette page.

Utilisation d'un pilote client Cassandra pour accéder à Amazon Keyspaces par programmation

Vous pouvez utiliser de nombreux pilotes Cassandra open source tiers pour vous connecter à Amazon Keyspaces. Amazon Keyspaces est compatible avec les pilotes Cassandra compatibles avec la version 3.11.2 d'Apache Cassandra. Voici les pilotes et les dernières versions que nous avons testés et que nous recommandons d'utiliser avec Amazon Keyspaces :

- `Java v3.3`
- `Java v4.17`
- `Python Cassandra-driver 3.29.1`
- `Node.js cassandra driver -v 4.7.2`
- `GO using GOCQL v1.6`
- `.NET CassandraCSharpDriver -v 3.20.1`

Pour de plus amples informations sur les pilotes Cassandra, veuillez consulter Pilotes client Apache Cassandra.

> **Note**
>
> Pour vous aider à démarrer, vous pouvez consulter et télécharger des exemples de end-to-end code permettant d'établir des connexions avec Amazon Keyspaces à l'aide de pilotes populaires. Consultez les exemples d'Amazon Keyspaces sur. GitHub

Les didacticiels de ce chapitre incluent une simple requête CQL pour confirmer que la connexion à Amazon Keyspaces a été correctement établie. Pour savoir comment utiliser les espaces de touches et les tables après vous être connecté à un point de terminaison Amazon Keyspaces, consultez. Référence du langage CQL Pour un step-by-step didacticiel expliquant comment se connecter à Amazon Keyspaces depuis un point de terminaison Amazon VPC, consultez. the section called “Connexion aux points de terminaison VPC”

Rubriques

- Utilisation d'un pilote client Cassandra Python pour accéder à Amazon Keyspaces par programmation
- Utilisation d'un pilote client Cassandra Node.js pour accéder à Amazon Keyspaces par programmation
- Utilisation d'un pilote client Cassandra .NET Core pour accéder à Amazon Keyspaces par programmation
- Utilisation d'un pilote client Cassandra Go pour accéder à Amazon Keyspaces par programmation
- Utilisation d'un pilote client Cassandra Perl pour accéder à Amazon Keyspaces par programmation

Utilisation d'un pilote client Cassandra Java pour accéder à Amazon Keyspaces par programmation

Cette section explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Java.

> **Note**
>
> Java 17 et le pilote DataStax Java 4.17 ne sont actuellement pris en charge que par la version bêta. Pour plus d'informations, consultez https://docs.datastax.com/en/developer/java-driver/4.17/upgrade_guide/.

Pour fournir aux utilisateurs et aux applications des informations d'identification leur permettant d'accéder par programmation aux ressources Amazon Keyspaces, vous pouvez effectuer l'une des opérations suivantes :

- Créez des informations d'identification spécifiques au service associées à un utilisateur AWS Identity and Access Management (IAM) spécifique.
- Pour renforcer la sécurité, nous recommandons de créer des clés d'accès IAM pour les identités IAM qui sont utilisées dans tous les AWS services. Le plugin d'authentification Amazon Keyspaces SigV4 pour les pilotes clients Cassandra vous permet d'authentifier les appels vers Amazon Keyspaces à l'aide de clés d'accès IAM plutôt que de votre nom d'utilisateur et de votre mot de passe. Pour plus d'informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

> **Note**
>
> Pour un exemple d'utilisation d'Amazon Keyspaces avec Spring Boot, consultez. https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

Rubriques

Avant de commencer

Pour vous connecter à Amazon Keyspaces, vous devez effectuer les tâches suivantes avant de pouvoir commencer.

1. Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients.

   a. Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

   ```
   curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
   ```

   > **Note**
   >
   > Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.

   b. Convertissez le certificat numérique Starfield en fichier TrustStore.

```
openssl x509 -outform der -in sf-class2-root.crt -out temp_file.der
keytool -import -alias cassandra -keystore cassandra_truststore.jks -file
 temp_file.der
```

Au cours de cette étape, vous devez créer un mot de passe pour le keystore et faire confiance à ce certificat. La commande interactive ressemble à ceci.

```
Enter keystore password:
Re-enter new password:
Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US
Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield
 Technologies, Inc.", C=US
Serial number: 0
Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034
Certificate fingerprints:
  MD5:  32:4A:4B:BB:C8:63:69:9B:BE:74:9A:C6:DD:1D:46:24
  SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A
  SHA256:
 14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB
Signature algorithm name: SHA1withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3
Extensions:
#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
[OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US]
SerialNumber: [    00]
]
#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]
#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
```

```
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
]
Trust this certificate? [no]:  y
```

2. Joignez le fichier trustStore dans les arguments de la JVM :

```
-Djavax.net.ssl.trustStore=path_to_file/cassandra_truststore.jks
-Djavax.net.ssl.trustStorePassword=my_password
```

Un tep-by-step tutoriel pour se connecter à Amazon Keyspaces à l'aide du pilote DataStax Java pour Apache Cassandra à l'aide d'informations d'identification spécifiques au service

Le step-by-step didacticiel suivant vous explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote Java pour Cassandra à l'aide d'informations d'identification spécifiques au service. Plus précisément, vous utiliserez la version 4.0 du pilote DataStax Java pour Apache Cassandra.

Rubriques

Étape 1 : Prérequis

Pour suivre ce didacticiel, vous devez générer des informations d'identification spécifiques au service et ajouter le pilote DataStax Java pour Apache Cassandra à votre projet Java.

- Générez des informations d'identification spécifiques au service pour votre utilisateur Amazon Keyspaces IAM en suivant les étapes décrites dans. the section called “Informations d'identification spécifiques au service” Si vous préférez utiliser les clés d'accès IAM pour l'authentification, consultezthe section called “Plugin d'authentification pour Java 4.x”.
- Ajoutez le pilote DataStax Java pour Apache Cassandra à votre projet Java. Assurez-vous que vous utilisez une version du pilote compatible avec Apache Cassandra 3.11.2. Pour plus d'informations, consultez la documentation du pilote DataStax Java pour Apache Cassandra.

Étape 2 : Configuration du pilote

Vous pouvez définir les paramètres du pilote DataStax Java Cassandra en créant un fichier de configuration pour votre application. Ce fichier de configuration remplace les paramètres par défaut et indique au pilote de se connecter au point de terminaison du service Amazon Keyspaces via le port 9142. Pour obtenir la liste des points de terminaison de service disponibles, consultezthe section called “Points de terminaison de service”.

Créez un fichier de configuration et enregistrez-le dans le dossier des ressources de l'application, par exemple,. `src/main/resources/application.conf` Ouvrez `application.conf` et ajoutez les paramètres de configuration suivants.

1. Fournisseur d'authentification : créez le fournisseur d'authentification avec la `PlainTextAuthProvider` classe. ServiceUserLe *`nom ServicePassword`*doit correspondre au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes décrites dans. Générer des informations d'identification spécifiques au service

   > **Note**
   >
   > Vous pouvez utiliser des informations d'identification à court terme en utilisant le plug-in d'authentification du pilote DataStax Java pour Apache Cassandra au lieu de les coder en dur dans le fichier de configuration du pilote. Pour en savoir plus, suivez les instructions duthe section called “Plugin d'authentification pour Java 4.x”.

2. Centre de données local : définissez la valeur `local-datacenter` pour la région à laquelle vous vous connectez. Par exemple, si l'application se connecte à`cassandra.us-east-2.amazonaws.com`, définissez le centre de données local sur`us-east-2`. Pour toutes les options disponibles Régions AWS, voir???. Paramétré `slow-replica-avoidance = false` pour équilibrer la charge en fonction d'un nombre réduit de nœuds.
3. SSL/TLS — Initialisez le SSL EngineFactory en ajoutant une section dans le fichier de configuration avec une seule ligne qui spécifie la classe avec. `class = DefaultSslEngineFactory` Indiquez le chemin d'accès au fichier TrustStore et le mot de passe que vous avez créés précédemment. Amazon Keyspaces ne prend pas en charge `hostname-validation` les pairs, alors définissez cette option sur false.

```
datastax-java-driver {
```

```
    basic.contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
    advanced.auth-provider{
        class = PlainTextAuthProvider
        username = "ServiceUserName"
        password = "ServicePassword"
    }
    basic.load-balancing-policy {
        local-datacenter = "us-east-2"
        slow-replica-avoidance = false
    }

    advanced.ssl-engine-factory {
        class = DefaultSslEngineFactory
        truststore-path = "./src/main/resources/cassandra_truststore.jks"
        truststore-password = "my_password"
        hostname-validation = false
      }
}
```

> **Note**
>
> Au lieu d'ajouter le chemin du TrustStore dans le fichier de configuration, vous pouvez également ajouter le chemin du TrustStore directement dans le code de l'application ou vous pouvez ajouter le chemin du TrustStore à vos arguments JVM.

Étape 3 : Exécuter l'exemple d'application

Cet exemple de code montre une application de ligne de commande simple qui crée un pool de connexions vers Amazon Keyspaces à l'aide du fichier de configuration que nous avons créé précédemment. Il confirme que la connexion est établie en exécutant une simple requête.

```
package <your package>;
// add the following imports to your project
import com.datastax.oss.driver.api.core.CqlSession;
import com.datastax.oss.driver.api.core.config.DriverConfigLoader;
import com.datastax.oss.driver.api.core.cql.ResultSet;
import com.datastax.oss.driver.api.core.cql.Row;

public class App
{
```

```
    public static void main( String[] args )
    {
        //Use DriverConfigLoader to load your configuration file
        DriverConfigLoader loader =
 DriverConfigLoader.fromClasspath("application.conf");
        try (CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build()) {

            ResultSet rs = session.execute("select * from system_schema.keyspaces");
            Row row = rs.one();
            System.out.println(row.getString("keyspace_name"));
        }
    }
}
```

> **Note**
>
> Utilisez un `try` bloc pour établir la connexion afin de garantir qu'elle est toujours fermée. Si vous n'utilisez pas de `try` blocage, pensez à fermer votre connexion pour éviter toute fuite de ressources.

Un tep-by-step tutoriel pour se connecter à Amazon Keyspaces à l'aide du pilote DataStax Java 4.x pour Apache Cassandra et du plugin d'authentification SigV4

La section suivante décrit comment utiliser le plug-in d'authentification SigV4 pour le pilote DataStax Java 4.x open source pour Apache Cassandra afin d'accéder à Amazon Keyspaces (pour Apache Cassandra). Le plugin est disponible dans le GitHubréférentiel.

Le plugin d'authentification SigV4 vous permet d'utiliser les informations d'identification IAM pour les utilisateurs ou les rôles lorsque vous vous connectez à Amazon Keyspaces. Au lieu de demander un nom d'utilisateur et un mot de passe, ce plugin signe les demandes d'API à l'aide de clés d'accès. Pour plus d’informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

Étape 1 : Prérequis

Pour suivre ce didacticiel, vous devez effectuer les tâches suivantes.

- Si ce n'est pas déjà fait, créez des informations d'identification pour votre utilisateur ou rôle IAM en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”. Ce didacticiel part du principe que les clés d'accès sont stockées sous forme de variables d'environnement. Pour plus d’informations, consultez the section called “Comment gérer les clés d'accès”.
- Ajoutez le pilote DataStax Java pour Apache Cassandra à votre projet Java. Assurez-vous que vous utilisez une version du pilote compatible avec Apache Cassandra 3.11.2. Pour plus d'informations, consultez la documentation du pilote DataStax Java pour Apache Cassandra.
- Ajoutez le plugin d'authentification à votre application. Le plugin d'authentification prend en charge la version 4.x du pilote DataStax Java pour Apache Cassandra. Si vous utilisez Apache Maven, ou un système de génération qui peut utiliser les dépendances Maven, ajoutez les dépendances suivantes à votre fichier `pom.xml`.

> **Important**
>
> Remplacez la version du plugin par la dernière version, comme indiqué dans le GitHub référentiel.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</artifactId>
        <version>4.0.9</version>
</dependency>
```

Étape 2 : Configuration du pilote

Vous pouvez définir les paramètres du pilote DataStax Java Cassandra en créant un fichier de configuration pour votre application. Ce fichier de configuration remplace les paramètres par défaut et indique au pilote de se connecter au point de terminaison du service Amazon Keyspaces via le port 9142. Pour obtenir la liste des points de terminaison de service disponibles, consultezthe section called “Points de terminaison de service”.

Créez un fichier de configuration et enregistrez-le dans le dossier des ressources de l'application, par exemple,. `src/main/resources/application.conf` Ouvrez `application.conf` et ajoutez les paramètres de configuration suivants.

1. Fournisseur d'authentification : définissez une nouvelle instance `desoftware.aws.mcs.auth.SigV4AuthProvider`. `advanced.auth-provider.class` Le SigV4 AuthProvider est le gestionnaire d'authentification fourni par le plugin pour effectuer l'authentification Sigv4.
2. Centre de données local : définissez la valeur `local-datacenter` pour la région à laquelle vous vous connectez. Par exemple, si l'application se connecte à`cassandra.us-east-2.amazonaws.com`, définissez le centre de données local sur`us-east-2`. Pour toutes les options disponibles Régions AWS, voir???. Réglé `slow-replica-avoidance = false` pour équilibrer la charge par rapport à tous les nœuds disponibles.
3. Idempotentie — Définissez la valeur par défaut `idempotence` pour que l'application configure le pilote pour qu'`true`il réessaie toujours les demandes de lecture/écriture/préparation/exécution qui ont échoué. Il s'agit d'une bonne pratique pour les applications distribuées qui permet de gérer les défaillances transitoires en réessayant les demandes ayant échoué.
4. SSL/TLS — Initialisez le SSL EngineFactory en ajoutant une section dans le fichier de configuration avec une seule ligne qui spécifie la classe avec. `class = DefaultSslEngineFactory` Indiquez le chemin d'accès au fichier TrustStore et le mot de passe que vous avez créés précédemment. Amazon Keyspaces ne prend pas en charge `hostname-validation` les pairs, alors définissez cette option sur false.
5. Connexions — Créez au moins 3 connexions locales par point de terminaison en définissant`local.size = 3`. Il s'agit d'une bonne pratique qui aide votre application à gérer les surcharges et les pics de trafic. Pour plus d'informations sur le calcul du nombre de connexions locales par point de terminaison dont votre application a besoin en fonction des modèles de trafic attendus, consultezthe section called “Comment configurer les connexions”.
6. Politique de nouvelle tentative — La `AmazonKeyspacesExponentialRetryPolicy` politique de nouvelle tentative d'Amazon Keyspaces est une alternative à celle fournie avec `DefaultRetryPolicy` le pilote Cassandra. La principale différence entre les deux politiques de nouvelle tentative est que vous pouvez configurer le nombre de tentatives en fonction `AmazonKeyspacesExponentialRetryPolicy` de vos besoins. Par défaut, le nombre de nouvelles tentatives pour le `AmazonKeyspacesExponentialRetryPolicy` est défini sur 3. En outre, la politique de réessai d'Amazon Keyspaces ne renvoie pas le générique. `NoHostAvailableException` La politique de nouvelles tentatives d'Amazon Keyspaces renvoie plutôt l'exception initiale renvoyée par le service. Pour plus d'exemples de code implémentant des politiques de nouvelle tentative, consultez les politiques de nouvelle tentative d'Amazon Keyspaces sur Github.

7. Instructions préparées : définissez la valeur `prepare-on-all-nodes` sur false pour optimiser l'utilisation du réseau.

```
datastax-java-driver {
    basic {
        contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
        request {
            timeout = 2 seconds
            consistency = LOCAL_QUORUM
            page-size = 1024
            default-idempotence = true
        }
        load-balancing-policy {
            local-datacenter = "us-east-2"
            class = DefaultLoadBalancingPolicy
            slow-replica-avoidance = false
        }
    }
    advanced {
        auth-provider {
            class = software.aws.mcs.auth.SigV4AuthProvider
            aws-region = us-east-2
        }
        ssl-engine-factory {
            class = DefaultSslEngineFactory
            truststore-path = "./src/main/resources/cassandra_truststore.jks"
            truststore-password = "my_password"
            hostname-validation = false
        }
        connection {
      connect-timeout = 5 seconds
      max-requests-per-connection = 512
      pool {
                local.size = 3
      }
        }
       retry-policy {
           class =  com.aws.ssa.keyspaces.retry.AmazonKeyspacesExponentialRetryPolicy
     max-attempts = 3
     min-wait = 10 mills
     max-wait = 100 mills
       }
```

```
        prepared-statements {
     prepare-on-all-nodes = false
        }
    }
}
```

> **Note**
>
> Au lieu d'ajouter le chemin du TrustStore dans le fichier de configuration, vous pouvez également ajouter le chemin du TrustStore directement dans le code de l'application ou vous pouvez ajouter le chemin du TrustStore à vos arguments JVM.

Étape 3 : Exécuter l'application

Cet exemple de code montre une application de ligne de commande simple qui crée un pool de connexions vers Amazon Keyspaces à l'aide du fichier de configuration que nous avons créé précédemment. Il confirme que la connexion est établie en exécutant une simple requête.

```
package <your package>;
// add the following imports to your project
import com.datastax.oss.driver.api.core.CqlSession;
import com.datastax.oss.driver.api.core.config.DriverConfigLoader;
import com.datastax.oss.driver.api.core.cql.ResultSet;
import com.datastax.oss.driver.api.core.cql.Row;

public class App
{

    public static void main( String[] args )
    {
        //Use DriverConfigLoader to load your configuration file
        DriverConfigLoader loader =
 DriverConfigLoader.fromClasspath("application.conf");
        try (CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build()) {

            ResultSet rs = session.execute("select * from system_schema.keyspaces");
            Row row = rs.one();
            System.out.println(row.getString("keyspace_name"));
        }
```

```
    }
}
```

> **Note**
>
> Utilisez un `try` bloc pour établir la connexion afin de garantir qu'elle est toujours fermée. Si vous n'utilisez pas de `try` blocage, pensez à fermer votre connexion pour éviter toute fuite de ressources.

Connectez-vous à Amazon Keyspaces à l'aide du pilote DataStax Java 3.x pour Apache Cassandra et du plugin d'authentification SigV4

La section suivante décrit comment utiliser le plug-in d'authentification SigV4 pour le pilote DataStax Java open source 3.x pour Apache Cassandra afin d'accéder à Amazon Keyspaces. Le plugin est disponible dans le GitHub référentiel.

Le plugin d'authentification SigV4 vous permet d'utiliser les informations d'identification IAM pour les utilisateurs et les rôles lorsque vous vous connectez à Amazon Keyspaces. Au lieu de demander un nom d'utilisateur et un mot de passe, ce plugin signe les demandes d'API à l'aide de clés d'accès. Pour plus d'informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

Étape 1 : Prérequis

Pour exécuter cet exemple de code, vous devez d'abord effectuer les tâches suivantes.

- Créez des informations d'identification pour votre utilisateur ou rôle IAM en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”. Ce didacticiel part du principe que les clés d'accès sont stockées sous forme de variables d'environnement. Pour plus d'informations, consultez the section called “Comment gérer les clés d'accès”.
- Suivez les étapes décrites the section called “Avant de commencer” pour télécharger le certificat numérique Starfield, le convertir en fichier TrustStore et joindre le fichier TrustStore dans les arguments de la JVM à votre application.
- Ajoutez le pilote DataStax Java pour Apache Cassandra à votre projet Java. Assurez-vous que vous utilisez une version du pilote compatible avec Apache Cassandra 3.11.2. Pour plus d'informations, consultez la documentation du pilote DataStax Java pour Apache Cassandra.

- Ajoutez le plugin d'authentification à votre application. Le plugin d'authentification prend en charge la version 3.x du pilote DataStax Java pour Apache Cassandra. Si vous utilisez Apache Maven, ou un système de génération qui peut utiliser les dépendances Maven, ajoutez les dépendances suivantes à votre fichier `pom.xml`. Remplacez la version du plugin par la dernière version, comme indiqué dans le GitHub référentiel.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin_3</artifactId>
        <version>3.0.3</version>
</dependency>
```

Étape 2 : Exécuter l'application

Cet exemple de code montre une application de ligne de commande simple qui crée un pool de connexions vers Amazon Keyspaces. Il confirme que la connexion est établie en exécutant une simple requête.

```
package <your package>;
// add the following imports to your project

import software.aws.mcs.auth.SigV4AuthProvider;
import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.ResultSet;
import com.datastax.driver.core.Row;
import com.datastax.driver.core.Session;

public class App
{

    public static void main( String[] args )
    {
        String endPoint = "cassandra.us-east-2.amazonaws.com";
        int portNumber = 9142;
        Session session = Cluster.builder()
                                 .addContactPoint(endPoint)
                                 .withPort(portNumber)
                                 .withAuthProvider(new SigV4AuthProvider("us-east-2"))

                                 .withSSL()
                                 .build()
```

```
                                    .connect();

        ResultSet rs = session.execute("select * from system_schema.keyspaces");
        Row row = rs.one();
        System.out.println(row.getString("keyspace_name"));
    }
}
```

Remarques d'utilisation :

Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Consultez le référentiel suivant pour obtenir des politiques, des exemples et des meilleures pratiques utiles en matière de pilote Java lors de l'utilisation du pilote Java avec Amazon Keyspaces :. https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers

Utilisation d'un pilote client Cassandra Python pour accéder à Amazon Keyspaces par programmation

Dans cette section, nous vous expliquons comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Python. Pour fournir aux utilisateurs et aux applications des informations d'identification leur permettant d'accéder par programmation aux ressources Amazon Keyspaces, vous pouvez effectuer l'une des opérations suivantes :

- Créez des informations d'identification spécifiques au service associées à un utilisateur AWS Identity and Access Management (IAM) spécifique.
- Pour renforcer la sécurité, nous recommandons de créer des clés d'accès IAM pour les utilisateurs ou les rôles IAM qui sont utilisées dans tous les AWS services. Le plugin d'authentification Amazon Keyspaces SigV4 pour les pilotes clients Cassandra vous permet d'authentifier les appels vers Amazon Keyspaces à l'aide de clés d'accès IAM plutôt que de votre nom d'utilisateur et de votre mot de passe. Pour plus d’informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

Rubriques

- Avant de commencer
- Connectez-vous à Amazon Keyspaces à l'aide du pilote Python pour Apache Cassandra et des informations d'identification spécifiques au service

- Connectez-vous à Amazon Keyspaces à l'aide du pilote DataStax Python pour Apache Cassandra et du plugin d'authentification SigV4

Avant de commencer

Vous devez effectuer la tâche suivante avant de pouvoir commencer.

Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients. Pour vous connecter à Amazon Keyspaces via le protocole TLS, vous devez télécharger un certificat numérique Amazon et configurer le pilote Python pour qu'il utilise le protocole TLS.

Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```


Connectez-vous à Amazon Keyspaces à l'aide du pilote Python pour Apache Cassandra et des informations d'identification spécifiques au service

L'exemple de code suivant vous montre comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Python et d'informations d'identification spécifiques au service.

```
from cassandra.cluster import Cluster
from ssl import SSLContext, PROTOCOL_TLSv1_2 , CERT_REQUIRED
from cassandra.auth import PlainTextAuthProvider

ssl_context = SSLContext(PROTOCOL_TLSv1_2 )
ssl_context.load_verify_locations('path_to_file/sf-class2-root.crt')
```

```
ssl_context.verify_mode = CERT_REQUIRED
auth_provider = PlainTextAuthProvider(username='ServiceUserName',
 password='ServicePassword')
cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,
 auth_provider=auth_provider, port=9142)
session = cluster.connect()
r = session.execute('select * from system_schema.keyspaces')
print(r.current_rows)
```

Remarques d'utilisation :

1. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.
2. Assurez-vous que le *ServiceUsernom* et le mot de passe *ServicePassword*correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes de. Générer des informations d'identification spécifiques au service
3. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Connectez-vous à Amazon Keyspaces à l'aide du pilote DataStax Python pour Apache Cassandra et du plugin d'authentification SigV4

La section suivante explique comment utiliser le plugin d'authentification SigV4 pour le pilote DataStax Python open source pour Apache Cassandra afin d'accéder à Amazon Keyspaces (pour Apache Cassandra).

Si ce n'est pas déjà fait, commencez par créer des informations d'identification pour votre rôle IAM en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”. Ce didacticiel utilise des informations d'identification temporaires, qui nécessitent un rôle IAM. Pour plus d'informations sur les informations d'identification temporaires, consultezthe section called “Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces”.

Ajoutez ensuite le plugin d'authentification Python SigV4 à votre environnement depuis le GitHub référentiel.

```
pip install cassandra-sigv4
```

L'exemple de code suivant montre comment se connecter à Amazon Keyspaces à l'aide du pilote DataStax Python open source pour Cassandra et du plugin d'authentification SigV4. Le plugin dépend du AWS SDK pour Python (Boto3). Il est utilisé `boto3.session` pour obtenir des informations d'identification temporaires.

```
from cassandra.cluster import Cluster
from ssl import SSLContext, PROTOCOL_TLSv1_2 , CERT_REQUIRED
from cassandra.auth import PlainTextAuthProvider
import boto3
from cassandra_sigv4.auth import SigV4AuthProvider

ssl_context = SSLContext(PROTOCOL_TLSv1_2)
ssl_context.load_verify_locations('path_to_file/sf-class2-root.crt')
ssl_context.verify_mode = CERT_REQUIRED

# use this if you want to use Boto to set the session parameters.
boto_session = boto3.Session(aws_access_key_id="AKIAIOSFODNN7EXAMPLE",
                             aws_secret_access_key="wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
                             aws_session_token="AQoDYXdzEJr...<remainder of token>",
                             region_name="us-east-2")
auth_provider = SigV4AuthProvider(boto_session)

# Use this instead of the above line if you want to use the Default Credentials and not
 bother with a session.
# auth_provider = SigV4AuthProvider()

cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,
 auth_provider=auth_provider,
                  port=9142)
session = cluster.connect()
r = session.execute('select * from system_schema.keyspaces')
print(r.current_rows)
```

Remarques d'utilisation :

1. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.
2. *Assurez-vous que le aws_access_key_id, le aws_secret_access_key et le aws_session_token correspondent au, et que vous avez obtenu en utilisant.*

`Access Key` `Secret Access Key` `Session Token` `boto3.session` Pour de plus amples informations, veuillez consulter Informations d'identification dans le AWS SDK for Python (Boto3).

3. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Utilisation d'un pilote client Cassandra Node.js pour accéder à Amazon Keyspaces par programmation

Cette section explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Node.js. Pour fournir aux utilisateurs et aux applications des informations d'identification leur permettant d'accéder par programmation aux ressources Amazon Keyspaces, vous pouvez effectuer l'une des opérations suivantes :

- Créez des informations d'identification spécifiques au service associées à un utilisateur AWS Identity and Access Management (IAM) spécifique.
- Pour renforcer la sécurité, nous recommandons de créer des clés d'accès IAM pour les utilisateurs ou les rôles IAM qui sont utilisées dans tous les AWS services. Le plugin d'authentification Amazon Keyspaces SigV4 pour les pilotes clients Cassandra vous permet d'authentifier les appels vers Amazon Keyspaces à l'aide de clés d'accès IAM plutôt que de votre nom d'utilisateur et de votre mot de passe. Pour plus d’informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

Rubriques

- Avant de commencer
- Connectez-vous à Amazon Keyspaces à l'aide du DataStax pilote Node.js pour Apache Cassandra et des informations d'identification spécifiques au service
- Connectez-vous à Amazon Keyspaces à l'aide du pilote DataStax Node.js pour Apache Cassandra et du plugin d'authentification SigV4

Avant de commencer

Vous devez effectuer la tâche suivante avant de pouvoir commencer.

Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients. Pour vous connecter à Amazon Keyspaces via le protocole TLS,

vous devez télécharger un certificat numérique Amazon et configurer le pilote Python pour qu'il utilise le protocole TLS.

Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```


Connectez-vous à Amazon Keyspaces à l'aide du DataStax pilote Node.js pour Apache Cassandra et des informations d'identification spécifiques au service

Configurez votre chauffeur pour qu'il utilise le certificat numérique Starfield pour TLS et qu'il s'authentifie à l'aide d'informations d'identification spécifiques au service. Par exemple :

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const auth = new cassandra.auth.PlainTextAuthProvider('ServiceUserName',
 'ServicePassword');
const sslOptions1 = {
         ca: [
                   fs.readFileSync('path_to_file/sf-class2-root.crt', 'utf-8')],
                   host: 'cassandra.us-west-2.amazonaws.com',
                   rejectUnauthorized: true
        };
const client = new cassandra.Client({
                  contactPoints: ['cassandra.us-west-2.amazonaws.com'],
                  localDataCenter: 'us-west-2',
                  authProvider: auth,
                  sslOptions: sslOptions1,
                  protocolOptions: { port: 9142 }
```

```
        });
const query = 'SELECT * FROM system_schema.keyspaces';

client.execute(query)
                    .then( result => console.log('Row from Keyspaces %s',
 result.rows[0]))
                    .catch( e=> console.log(`${e}`));
```

Remarques d'utilisation :

1. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.
2. Assurez-vous que le *ServiceUsernom* et le mot de passe *ServicePassword*correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes de. Générer des informations d'identification spécifiques au service
3. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Connectez-vous à Amazon Keyspaces à l'aide du pilote DataStax Node.js pour Apache Cassandra et du plugin d'authentification SigV4

La section suivante explique comment utiliser le plugin d'authentification SigV4 pour le pilote open source DataStax Node.js pour Apache Cassandra afin d'accéder à Amazon Keyspaces (pour Apache Cassandra).

Si ce n'est pas déjà fait, créez des informations d'identification pour votre utilisateur ou rôle IAM en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”.

Ajoutez le plugin d'authentification SigV4 Node.js à votre application depuis le GitHub référentiel. Le plugin prend en charge la version 4.x du pilote DataStax Node.js pour Cassandra et dépend du AWS SDK pour Node.js. Il est utilisé `AWSCredentialsProvider` pour obtenir des informations d'identification.

```
$ npm install aws-sigv4-auth-cassandra-plugin --save
```

Cet exemple de code montre comment définir une instance spécifique à une région `SigV4AuthProvider` comme fournisseur d'authentification.

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const sigV4 = require('aws-sigv4-auth-cassandra-plugin');

const auth = new sigV4.SigV4AuthProvider({
    region: 'us-west-2',
    accessKeyId:'AKIAIOSFODNN7EXAMPLE',
    secretAccessKey: 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY'});

const sslOptions1 = {
  ca: [
      fs.readFileSync('path_to_filecassandra/sf-class2-root.crt', 'utf-8')],
  host: 'cassandra.us-west-2.amazonaws.com',
  rejectUnauthorized: true
};


const client = new cassandra.Client({
  contactPoints: ['cassandra.us-west-2.amazonaws.com'],
  localDataCenter: 'us-west-2',
  authProvider: auth,
  sslOptions: sslOptions1,
  protocolOptions: { port: 9142 }
});


const query = 'SELECT * FROM system_schema.keyspaces';

client.execute(query).then(
    result => console.log('Row from Keyspaces %s', result.rows[0]))
    .catch( e=> console.log(`${e}`));
```

Remarques d'utilisation :

1. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.
2. Assurez-vous que l'*accès KeyId* et le *secret AccessKey* correspondent à la clé d'accès et à la clé d'accès secrète que vous avez obtenues à l'aideAWSCredentialsProvider. Pour plus d'informations, consultez la section Configuration des informations d'identification dans Node.js dans le AWS SDK pour JavaScript Node.js.

3. Pour stocker les clés d'accès en dehors du code, consultez les meilleures pratiques surthe section called “Comment gérer les clés d'accès”.
4. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Utilisation d'un pilote client Cassandra .NET Core pour accéder à Amazon Keyspaces par programmation

Cette section explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client .NET Core. Les étapes de configuration varient en fonction de votre environnement et de votre système d'exploitation, vous devrez peut-être les modifier en conséquence. Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients. Pour vous connecter à Amazon Keyspaces via le protocole TLS, vous devez télécharger un certificat numérique Starfield et configurer votre pilote pour qu'il utilise le protocole TLS.

1. Téléchargez le certificat Starfield et enregistrez-le dans un répertoire local en prenant note du chemin. Voici un exemple d'utilisation de PowerShell.

```
$client = new-object System.Net.WebClient
$client.DownloadFile("https://certs.secureserver.net/repository/sf-class2-
root.crt","path_to_file\sf-class2-root.crt")
```

2. Installez le CassandraC SharpDriver via Nuget, à l'aide de la console Nuget.

```
PM> Install-Package CassandraCSharpDriver
```

3. L'exemple suivant utilise un projet de console .NET Core C# pour se connecter à Amazon Keyspaces et exécuter une requête.

```
using Cassandra;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Security;
using System.Runtime.ConstrainedExecution;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;

namespace CSharpKeyspacesExample
```

```
{
    class Program
    {
        public Program(){}

        static void Main(string[] args)
        {
            X509Certificate2Collection certCollection = new
 X509Certificate2Collection();
            X509Certificate2 amazoncert = new X509Certificate2(@"path_to_file\sf-
class2-root.crt");
            var userName = "ServiceUserName";
            var pwd = "ServicePassword";
            certCollection.Add(amazoncert);

            var awsEndpoint =  "cassandra.us-east-2.amazonaws.com" ;

            var cluster = Cluster.Builder()
                     .AddContactPoints(awsEndpoint)
                     .WithPort(9142)
                     .WithAuthProvider(new PlainTextAuthProvider(userName, pwd))
                     .WithSSL(new
 SSLOptions().SetCertificateCollection(certCollection))
                     .Build();

            var session = cluster.Connect();
            var rs = session.Execute("SELECT * FROM system_schema.tables;");
            foreach (var row in rs)
            {
                var name = row.GetValue<String>("keyspace_name");
                Console.WriteLine(name);
            }
        }
    }
}
```

Remarques d'utilisation :

a. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.

b. Assurez-vous que le *ServiceUsernom* et le mot de passe *ServicePassword*correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré

les informations d'identification spécifiques au service en suivant les étapes de. Générer des informations d'identification spécifiques au service

c. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Utilisation d'un pilote client Cassandra Go pour accéder à Amazon Keyspaces par programmation

Cette section explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Go. Pour fournir aux utilisateurs et aux applications des informations d'identification leur permettant d'accéder par programmation aux ressources Amazon Keyspaces, vous pouvez effectuer l'une des opérations suivantes :

- Créez des informations d'identification spécifiques au service associées à un utilisateur AWS Identity and Access Management (IAM) spécifique.
- Pour renforcer la sécurité, nous recommandons de créer des clés d'accès IAM pour les utilisateurs et les rôles IAM qui soient utilisées dans tous les AWS services. Le plugin d'authentification Amazon Keyspaces SigV4 pour les pilotes clients Cassandra vous permet d'authentifier les appels vers Amazon Keyspaces à l'aide de clés d'accès IAM plutôt que de votre nom d'utilisateur et de votre mot de passe. Pour plus d’informations, consultez the section called “Informations d'identification IAM pour l' AWS authentification”.

Rubriques

- Avant de commencer
- Connectez-vous à Amazon Keyspaces à l'aide du pilote Gocql pour Apache Cassandra et des informations d'identification spécifiques au service
- Connectez-vous à Amazon Keyspaces à l'aide du pilote Go pour Apache Cassandra et du plugin d'authentification SigV4

Avant de commencer

Vous devez effectuer la tâche suivante avant de pouvoir commencer.

Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients. Pour vous connecter à Amazon Keyspaces via le protocole TLS,

vous devez télécharger un certificat numérique Amazon et configurer le pilote Python pour qu'il utilise le protocole TLS.

Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -0
```

> **Note**
>
> Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -0
```


Connectez-vous à Amazon Keyspaces à l'aide du pilote Gocql pour Apache Cassandra et des informations d'identification spécifiques au service

1. Créez un nouveau répertoire pour votre application.

   ```
   mkdir ./gocqlexample
   ```

2. Accédez au nouveau répertoire.

   ```
   cd gocqlexample
   ```

3. Créez un fichier pour votre candidature.

   ```
   touch cqlapp.go
   ```

4. Téléchargez le pilote Go.

   ```
   go get github.com/gocql/gocql
   ```

5. Ajoutez l'exemple de code suivant au fichier cqlapp.go.

   ```
   package main
   ```

```
import (
     "fmt"
     "github.com/gocql/gocql"
     "log"
)

func main() {

    // add the Amazon Keyspaces service endpoint
    cluster := gocql.NewCluster("cassandra.us-east-2.amazonaws.com")
    cluster.Port=9142
    // add your service specific credentials
    cluster.Authenticator = gocql.PasswordAuthenticator{
            Username: "ServiceUserName",
            Password: "ServicePassword"}
    // provide the path to the sf-class2-root.crt
    cluster.SslOpts = &gocql.SslOptions{
            CaPath: "path_to_file/sf-class2-root.crt",
            EnableHostVerification: false,
     }

     // Override default Consistency to LocalQuorum
     cluster.Consistency = gocql.LocalQuorum
     cluster.DisableInitialHostLookup = false

     session, err := cluster.CreateSession()
     if err != nil {
            fmt.Println("err>", err)
     }
     defer session.Close()

     // run a sample query from the system keyspace
     var text string
     iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
     for iter.Scan(&text) {
            fmt.Println("keyspace_name:", text)
     }
     if err := iter.Close(); err != nil {
            log.Fatal(err)
     }
     session.Close()
}
```

Remarques d'utilisation :

a. `"path_to_file/sf-class2-root.crt"`Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.

b. Assurez-vous que le *`ServiceUsernom`* et le mot de passe *`ServicePassword`*correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes de. Générer des informations d'identification spécifiques au service

c. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

6. Créez le programme.

```
go build cqlapp.go
```

7. Exécutez le programme.

```
./cqlapp
```

Connectez-vous à Amazon Keyspaces à l'aide du pilote Go pour Apache Cassandra et du plugin d'authentification SigV4

L'exemple de code suivant montre comment utiliser le plugin d'authentification SigV4 pour le pilote Go open source afin d'accéder à Amazon Keyspaces (pour Apache Cassandra).

Si ce n'est pas déjà fait, créez des informations d'identification pour votre utilisateur ou rôle IAM en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”.

Ajoutez le plugin d'authentification Go SigV4 à votre application depuis le GitHubréférentiel. Le plugin prend en charge la version 1.2.x du pilote open source Go pour Cassandra et dépend du SDK pour AWS Go.

```
$ go mod init
$ go get github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin
```

Dans cet exemple de code, le point de terminaison Amazon Keyspaces est représenté par la `Cluster` classe. Il utilise la propriété `AwsAuthenticator` for the authenticator du cluster pour obtenir des informations d'identification.

```
package main

import (
        "fmt"
        "github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin/sigv4"
        "github.com/gocql/gocql"
        "log"
)

func main() {
    // configuring the cluster options
    cluster := gocql.NewCluster("cassandra.us-west-2.amazonaws.com")
    cluster.Port=9142
    var auth sigv4.AwsAuthenticator = sigv4.NewAwsAuthenticator()
    auth.Region = "us-west-2"
    auth.AccessKeyId = "AKIAIOSFODNN7EXAMPLE"
    auth.SecretAccessKey = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"

    cluster.Authenticator = auth

    cluster.SslOpts = &gocql.SslOptions{
            CaPath: "path_to_file/sf-class2-root.crt",
            EnableHostVerification: false,
    }
    cluster.Consistency = gocql.LocalQuorum
    cluster.DisableInitialHostLookup = false

    session, err := cluster.CreateSession()
    if err != nil {
     fmt.Println("err>", err)
     return
    }
    defer session.Close()

    // doing the query
    var text string
    iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
    for iter.Scan(&text) {
     fmt.Println("keyspace_name:", text)
    }
    if err := iter.Close(); err != nil {
     log.Fatal(err)
    }
```

```
}
```

Remarques d'utilisation :

1. "*path_to_file*/sf-class2-root.crt"Remplacez-le par le chemin d'accès au certificat enregistré lors de la première étape.
2. Assurez-vous que l'*AccessKeyidentifiant* et la *SecretAccessclé correspondent à la clé* d'accès et à la clé d'accès secrète que vous avez obtenue à l'aideAwsAuthenticator. Pour plus d'informations, consultez la section Configuration du AWS SDK pour Go dans le AWS SDK for Go.
3. Pour stocker les clés d'accès en dehors du code, consultez les meilleures pratiques surthe section called “Comment gérer les clés d'accès”.
4. Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

Utilisation d'un pilote client Cassandra Perl pour accéder à Amazon Keyspaces par programmation

Cette section explique comment vous connecter à Amazon Keyspaces à l'aide d'un pilote client Perl. Pour cet exemple de code, nous avons utilisé Perl 5. Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients.

> **Important**
>
> Pour créer une connexion sécurisée, nos exemples de code utilisent le certificat numérique Starfield pour authentifier le serveur avant d'établir la connexion TLS. Le pilote Perl ne valide pas le certificat SSL Amazon du serveur, ce qui signifie que vous ne pouvez pas confirmer que vous êtes connecté à Amazon Keyspaces. La deuxième étape, qui consiste à configurer le pilote pour utiliser le protocole TLS lors de la connexion à Amazon Keyspaces, est toujours requise et garantit que les données transférées entre le client et le serveur sont cryptées.

1. Téléchargez le pilote Cassandra DBI depuis https://metacpan.org/pod/DBD::Cassandra et installez-le dans votre environnement Perl. Les étapes exactes dépendent de l'environnement. Voici un exemple courant.

```
cpanm DBD::Cassandra
```

2. Créez un fichier pour votre candidature.

```
touch cqlapp.pl
```

3. Ajoutez l'exemple de code suivant au fichier cqlapp.pl.

```
use DBI;
my $user = "ServiceUserName";
my $password = "ServicePassword";
my $db = DBI->connect("dbi:Cassandra:host=cassandra.us-
east-2.amazonaws.com;port=9142;tls=1;",
$user, $password);

my $rows = $db->selectall_arrayref("select * from system_schema.keyspaces");
print "Found the following Keyspaces...\n";
for my $row (@$rows) {
      print join(" ",@$row['keyspace_name']),"\n";
}

$db->disconnect;
```

> **Important**
>
> Assurez-vous que le *ServiceUsernom* et le mot de passe *ServicePassword*correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes de. Générer des informations d'identification spécifiques au service

> **Note**
>
> Pour obtenir la liste des points de terminaison disponibles, reportez-vous à la section the section called “Points de terminaison de service”.

4. Exécutez l'application.

```
perl cqlapp.pl
```

Tutoriel : Connexion à Amazon Keyspaces depuis Amazon Elastic Kubernetes Service

Ce didacticiel explique les étapes nécessaires à la configuration d'un cluster Amazon Elastic Kubernetes Service (Amazon EKS) afin d'héberger une application conteneurisée qui se connecte à Amazon Keyspaces à l'aide de l'authentification Sigv4.

Amazon EKS est un service géré qui élimine le besoin d'installer, d'exploiter et de maintenir votre propre plan de contrôle Kubernetes. Kubernetes est un système open source qui automatise la gestion, la mise à l'échelle et le déploiement d'applications conteneurisées.

Ce didacticiel fournit step-by-step des conseils pour configurer, créer et déployer une application Java conteneurisée sur Amazon EKS. Dans la dernière étape, vous exécutez l'application pour écrire des données dans une table Amazon Keyspaces.

Rubriques

Prérequis du didacticiel

Créez les AWS ressources suivantes avant de commencer le didacticiel

1. Avant de commencer ce didacticiel, suivez les instructions de AWS configuration indiquées dansAccès à Amazon Keyspaces (pour Apache Cassandra). Ces étapes incluent l'inscription AWS et la création d'un AWS Identity and Access Management (IAM) principal ayant accès à Amazon Keyspaces.

2. Créez un espace de touches Amazon Keyspaces avec le nom `aws` et une table avec le nom dans `user` lequel vous pouvez écrire depuis l'application conteneurisée exécutée dans Amazon EKS plus loin dans ce didacticiel. Vous pouvez le faire avec AWS CLI ou en utilisant`cqlsh`.

   AWS CLI

   ```
   aws keyspaces create-keyspace --keyspace-name 'aws'
   ```

   Pour confirmer que le keyspace a été créé, vous pouvez utiliser la commande suivante.

   ```
   aws keyspaces list-keyspaces
   ```

   Pour créer la table, vous pouvez utiliser la commande suivante.

   ```
   aws keyspaces create-table --keyspace-name 'aws' --table-name 'user' --schema-
   definition 'allColumns=[
               {name=username,type=text}, {name=fname,type=text},
   {name=last_update_date,type=timestamp},{name=lname,type=text}],
               partitionKeys=[{name=username}]'
   ```

   Pour confirmer que votre table a été créée, vous pouvez utiliser la commande suivante.

   ```
   aws keyspaces list-tables --keyspace-name 'aws'
   ```

   Pour plus d'informations, voir créer un espace de touches et créer une table dans la référence des AWS CLI commandes.

   cqlsh

   ```
   CREATE KEYSPACE aws WITH replication = {'class': 'SimpleStrategy',
    'replication_factor': '3'}  AND durable_writes = true;
   CREATE TABLE aws.user (
       username text PRIMARY KEY,
       fname text,
       last_update_date timestamp,
       lname text
   );
   ```

   Pour vérifier que votre table a été créée, vous pouvez utiliser l'instruction suivante.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

Votre table doit être répertoriée dans le résultat de cette instruction. Notez qu'il peut y avoir un délai avant que la table ne soit créée. Pour de plus amples informations, veuillez consulter the section called “CREATE TABLE”.

3. Créez un cluster Amazon EKS avec un nœud de type Fargate - Linux. Fargate est un moteur de calcul sans serveur qui vous permet de déployer des pods Kubernetes sans gérer les instances Amazon Amazon EC2. Pour suivre ce didacticiel sans avoir à mettre à jour le nom du cluster dans tous les exemples de commandes, créez un cluster portant ce nom en `my-eks-cluster` suivant les instructions de la section Getting started with Amazon EKS, `eksctl` dans le guide de l'utilisateur Amazon EKS. Lorsque votre cluster est créé, vérifiez que vos nœuds et les deux pods par défaut fonctionnent et fonctionnent correctement. Vous pouvez le faire à l'aide de la commande suivante.

   ```
   kubectl get pods -A -o wide
   ```

   Vous devriez voir quelque chose de similaire à cette sortie.

   ```
   NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE   IP
        NODE                                                 NOMINATED NODE   READINESS
    GATES
   kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
    192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
    <none>
   kube-system   coredns-1234567890-12345   1/1     Running   0          18m
    192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
    <none>
   ```

4. Installez Docker. Pour savoir comment installer Docker sur une instance Amazon EC2, consultez Installer Docker dans le guide de l'utilisateur d'Amazon Elastic Container Registry.

   Docker est disponible pour plusieurs systèmes d'exploitation, notamment les distributions Linux les plus modernes, comme Ubuntu et même MacOS et Windows. Pour en savoir plus sur la façon d'installer Docker sur votre système d'exploitation, consultez le guide d'installation Docker.

5. Créez un référentiel Amazon ECR. Amazon ECR est un service de registre d'images de conteneurs AWS géré que vous pouvez utiliser avec votre CLI préférée pour envoyer, extraire et gérer des images Docker. Pour plus d'informations sur les référentiels Amazon ECR, consultez

le guide de l'utilisateur d'Amazon Elastic Container Registry. Vous pouvez utiliser la commande suivante pour créer un référentiel portant ce nom`my-ecr-repository`.

```
aws ecr create-repository --repository-name my-ecr-repository
```

Après avoir effectué les étapes préalables, passez àthe section called “Étape 1 : Configuration du cluster Amazon EKS”.

Étape 1 : configurer le cluster Amazon EKS et configurer les autorisations IAM

Configurez le cluster Amazon EKS et créez les ressources IAM nécessaires pour permettre à un compte de service Amazon EKS de se connecter à votre table Amazon Keyspaces

1. Créez un fournisseur Open ID Connect (OIDC) pour le cluster Amazon EKS. Cela est nécessaire pour utiliser les rôles IAM pour les comptes de service. Pour plus d'informations sur les fournisseurs OIDC et sur la façon de les créer, consultez la section Création d'un fournisseur OIDC IAM pour votre cluster dans le guide de l'utilisateur Amazon EKS.

   a. Créez votre fournisseur d'identité OIDC IAM pour votre cluster avec la commande suivante. Cet exemple suppose que le nom de votre cluster est`my-eks-cluster`. Si vous avez un cluster portant un nom différent, pensez à le mettre à jour dans toutes les futures commandes.

   ```
   eksctl utils associate-iam-oidc-provider --cluster my-eks-cluster --approve
   ```

   b. Vérifiez que le fournisseur d'identité OIDC a été enregistré auprès d'IAM à l'aide de la commande suivante.

   ```
   aws iam list-open-id-connect-providers --region aws-region
   ```

   La sortie doit ressembler à ceci. Prenez note du nom de ressource Amazon (ARN) de l'OIDC, vous en aurez besoin à l'étape suivante lorsque vous créerez une politique de confiance pour le compte de service.

   ```
   {
       "OpenIDConnectProviderList": [
           ..
           {
   ```

```
            "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.aws-
region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
        }
    ]
}
```

2. Créez un compte de service pour le cluster Amazon EKS. Les comptes de service fournissent une identité pour les processus exécutés dans un pod. Un Pod est l'objet Kubernetes le plus petit et le plus simple que vous puissiez utiliser pour déployer une application conteneurisée. Créez ensuite un rôle IAM que le compte de service peut assumer pour obtenir des autorisations d'accès aux ressources. Vous pouvez accéder à n'importe quel AWS service à partir d'un Pod configuré pour utiliser un compte de service pouvant assumer un rôle IAM avec des autorisations d'accès à ce service.

   a. Créez un nouvel espace de noms pour le compte de service. Un espace de noms permet d'isoler les ressources du cluster créées pour ce didacticiel. Vous pouvez créer un nouvel espace de noms à l'aide de la commande suivante.

   ```
   kubectl create namespace my-eks-namespace
   ```

   b. Pour utiliser un espace de noms personnalisé, vous devez l'associer à un profil Fargate. Le code suivant en est un exemple.

   ```
   eksctl create fargateprofile \
       --cluster my-eks-cluster \
       --name my-fargate-profile \
       --namespace my-eks-namespace \
       --labels *=*
   ```

   c. Créez un compte de service dont le nom figure `my-eks-serviceaccount` dans l'espace de noms `my-eks-namespace` de votre cluster Amazon EKS à l'aide de la commande suivante.

   ```
   cat >my-serviceaccount.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     name: my-eks-serviceaccount
     namespace: my-eks-namespace
   EOF
   ```

```
kubectl apply -f my-serviceaccount.yaml
```

d. Exécutez la commande suivante pour créer un fichier de politique de confiance qui indique au rôle IAM de faire confiance à votre compte de service. Cette relation de confiance est requise avant qu'un mandant puisse assumer un rôle. Vous devez apporter les modifications suivantes au fichier :

- Pour le`Principal`, entrez l'ARN renvoyé par IAM à la `list-open-id-connect-providers` commande. L'ARN contient votre numéro de compte et votre région.
- Dans l'`condition`instruction, remplacez le Région AWS et l'identifiant OIDC.
- Vérifiez que le nom du compte de service et l'espace de noms sont corrects.

Vous devez joindre le fichier de politique de confiance à l'étape suivante lorsque vous créez le rôle IAM.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.aws-region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.aws-region.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount",
                    "oidc.eks.aws-region.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
EOF
```

Facultatif : vous pouvez également ajouter plusieurs entrées dans les `StringLike` conditions `StringEquals` ou pour permettre à plusieurs comptes de service ou espaces de noms d'assumer le rôle. Pour autoriser votre compte de service à assumer un rôle IAM dans un autre AWS compte, consultez la section Autorisations IAM entre comptes dans le guide de l'utilisateur Amazon EKS.

3. Créez un rôle IAM avec le nom `my-iam-role` que le compte de service Amazon EKS doit assumer. Joignez le fichier de politique de confiance créé lors de la dernière étape au rôle. La politique de confiance spécifie le compte de service et le fournisseur OIDC auxquels le rôle IAM peut faire confiance.

```
aws iam create-role --role-name my-iam-role --assume-role-policy-document file://
trust-relationship.json --description "EKS service account role"
```

4. Attribuez les autorisations du rôle IAM à Amazon Keyspaces en joignant une politique d'accès.

   a. Joignez une politique d'accès pour définir les actions que le rôle IAM peut effectuer sur des ressources Amazon Keyspaces spécifiques. Pour ce didacticiel, nous utilisons la politique AWS géréeAmazonKeyspacesFullAccess, car notre application va écrire des données dans votre table Amazon Keyspaces. À titre de bonne pratique, il est toutefois recommandé de créer des politiques d'accès personnalisées qui mettent en œuvre le principe du moindre privilège. Pour de plus amples informations, veuillez consulter the section called “Comment Amazon Keyspaces fonctionne avec IAM”.

```
aws iam attach-role-policy --role-name my-iam-role --policy-
arn=arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
```

   Vérifiez que la politique a été correctement attachée au rôle IAM à l'aide de l'instruction suivante.

```
aws iam list-attached-role-policies --role-name my-iam-role
```

   La sortie doit ressembler à ceci.

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonKeyspacesFullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess"
```

```
        }
    ]
}
```

b. Annotez le compte de service avec l'Amazon Resource Name (ARN) du rôle IAM qu'il peut assumer. Assurez-vous de mettre à jour l'ARN du rôle avec votre identifiant de compte.

```
kubectl annotate serviceaccount -n my-eks-namespace my-eks-serviceaccount
 eks.amazonaws.com/role-arn=arn:aws:iam::111122223333:role/my-iam-role
```

5. Vérifiez que le rôle IAM et le compte de service sont correctement configurés.

a. Vérifiez que la politique de confiance du rôle IAM est correctement configurée à l'aide de l'instruction suivante.

```
aws iam get-role --role-name my-iam-role --query Role.AssumeRolePolicyDocument
```

La sortie doit ressembler à ceci.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.aws-region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.aws-region/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.aws-region.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount"
                }
            }
        }
    ]
}
```

b. Vérifiez que le compte de service Amazon EKS est annoté avec le rôle IAM.

```
kubectl describe serviceaccount my-eks-serviceaccount -n my-eks-namespace
```

La sortie doit ressembler à ceci.

```
Name: my-eks-serviceaccount
Namespace:my-eks-namespace
Labels: <none>
Annotations: eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-iam-
role
Image pull secrets: <none>
Mountable secrets: <none>
Tokens: <none>
[...]
```

Après avoir créé le compte de service Amazon EKS, le rôle IAM et configuré les relations et les autorisations requises, passez àthe section called “Étape 2 : Configuration de l'application”.

Étape 2 : Configuration de l'application

Au cours de cette étape, vous allez créer votre application qui se connecte à Amazon Keyspaces à l'aide du plugin SigV4. Vous pouvez consulter et télécharger l'exemple d'application Java depuis le dépôt d'exemples de code Amazon Keyspaces sur Github. Vous pouvez également suivre le processus en utilisant votre propre application, en vous assurant de terminer toutes les étapes de configuration.

Configurez votre application et ajoutez les dépendances requises.

1. Vous pouvez télécharger l'exemple d'application Java en clonant le référentiel Github à l'aide de la commande suivante.

```
git clone https://github.com/aws-samples/amazon-keyspaces-examples.git
```

2. Après avoir téléchargé le dépôt Github, décompressez le fichier téléchargé et accédez au `resources` répertoire du fichier. `application.conf`

   a. Configuration de l'application

Au cours de cette étape, vous configurez le plugin d'authentification SigV4. Vous pouvez utiliser l'exemple suivant dans votre application. Si ce n'est pas déjà fait, vous devez générer vos clés d'accès IAM (un identifiant de clé d'accès et une clé d'accès secrète) et les enregistrer dans votre fichier de AWS configuration ou en tant que variables d'environnement. Pour obtenir des instructions complètes, veuillez consulter the section called “Informations d'identification requises pour AWS l'authentification”. Mettez à jour la AWS région et le point de terminaison du service pour Amazon Keyspaces selon les besoins. Pour plus de points de terminaison de service, voirthe section called “Points de terminaison de service”. Remplacez l'emplacement du truststore, le nom du truststore et le mot de passe du truststore par les vôtres.

```
datastax-java-driver {
  basic.contact-points = ["cassandra.aws-region.amazonaws.com:9142"]
  basic.load-balancing-policy.local-datacenter = "aws-region"
  advanced.auth-provider {
    class = software.aws.mcs.auth.SigV4AuthProvider
    aws-region = "aws-region"
  }
  advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "truststore_locationtruststore_name.jks"
    truststore-password = "truststore_password;"
  }
}
```

b. Ajoutez la dépendance du module STS.

Cela permet d'utiliser un `WebIdentityTokenCredentialsProvider` qui renvoie les AWS informations d'identification que l'application doit fournir pour que le compte de service puisse assumer le rôle IAM. Vous pouvez le faire en vous basant sur l'exemple suivant.

```
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-sts</artifactId>
            <version>1.11.717</version>
        </dependency>
```

c. Ajoutez la dépendance SigV4.

Ce package implémente le plugin d'authentification SigV4 qui est nécessaire pour s'authentifier auprès d'Amazon Keyspaces.

```
        <dependency>
            <groupId>software.aws.mcs</groupId>
            <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</
artifactId>
            <version>4.0.3</version>
        </dependency>
```

3. Ajoutez une dépendance de journalisation.

Sans journaux, il est impossible de résoudre les problèmes de connexion. Dans ce didacticiel, nous l'utilisons `slf4j` comme framework de journalisation et nous l'utilisons `logback.xml` pour stocker la sortie du journal. Nous avons défini le niveau de journalisation `debug` sur pour établir la connexion. Vous pouvez utiliser l'exemple suivant pour ajouter la dépendance.

```
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>2.0.5</version>
        </dependency>
```

Vous pouvez utiliser l'extrait de code suivant pour configurer la journalisation.

```
<configuration>
    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">

        <encoder>
            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</
pattern>
        </encoder>
    </appender>

    <root level="debug">
        <appender-ref ref="STDOUT" />
    </rootv
</configuration>
```

> **Note**
>
> Ce `debug` niveau est nécessaire pour étudier les défaillances de connexion. Une fois que vous vous êtes connecté avec succès à Amazon Keyspaces depuis votre application, vous pouvez modifier le niveau de journalisation en fonction de vos `info` besoins`warning`.

Étape 3 : créer l'image de l'application et télécharger le fichier Docker dans votre référentiel Amazon ECR

Au cours de cette étape, vous compilez l'exemple d'application, vous créez une image Docker et vous envoyez l'image dans votre référentiel Amazon ECR.

Créez votre application, créez une image Docker et soumettez-la à Amazon Elastic Container Registry

1. Définissez des variables d'environnement pour la version qui définissent votre Région AWS. Remplacez les régions dans les exemples par les vôtres.

```
export CASSANDRA_HOST=cassandra.aws-region.amazonaws.com:9142
export CASSANDRA_DC=aws-region
```

2. Compilez votre application avec Apache Maven version 3.6.3 ou supérieure à l'aide de la commande suivante.

```
mvn clean install
```

   Cela crée un `JAR` fichier avec toutes les dépendances incluses dans le `target` répertoire.

3. Récupérez l'URI de votre référentiel ECR nécessaire pour l'étape suivante à l'aide de la commande suivante. Assurez-vous de mettre à jour la région avec celle que vous avez utilisée.

```
aws ecr describe-repositories --region aws-region
```

   La sortie doit ressembler à celle de l'exemple suivant.

```
"repositories": [
```

```
 {
 "repositoryArn": "arn:aws:ecr:aws-region:111122223333:repository/my-ecr-
repository",
 "registryId": "111122223333",
 "repositoryName": "my-ecr-repository",
 "repositoryUri": "111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-
repository",
 "createdAt": "2023-11-02T03:46:34+00:00",
 "imageTagMutability": "MUTABLE",
 "imageScanningConfiguration": {
 "scanOnPush": false
 },
 "encryptionConfiguration": {
 "encryptionType": "AES256"
 }
 },
```

4. À partir du répertoire racine de l'application, créez l'image Docker en utilisant l'URI du référentiel indiqué à la dernière étape. Modifiez le fichier Docker selon vos besoins. Dans la commande build, assurez-vous de remplacer votre identifiant de compte et de Région AWS définir la région dans laquelle se trouve le référentiel `my-ecr-repository` Amazon ECR.

```
docker build -t 111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-
repository:latest .
```

5. Récupérez un jeton d'authentification pour envoyer l'image Docker vers Amazon ECR. Vous pouvez le faire à l'aide de la commande suivante.

```
aws ecr get-login-password --region aws-region | docker login --username AWS --
password-stdin 111122223333.dkr.ecr.aws-region.amazonaws.com
```

6. Tout d'abord, vérifiez les images existantes dans votre référentiel Amazon ECR. Vous pouvez utiliser la commande suivante :

```
aws ecr describe-images --repository-name my-ecr-repository --region aws-region
```

Ensuite, placez l'image Docker dans le dépôt. Vous pouvez utiliser la commande suivante :

```
docker push 111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-repository:latest
```

Étape 4 : Déployer l'application sur Amazon EKS et écrire des données dans votre table Amazon Keyspaces

Dans cette étape du didacticiel, vous configurez le déploiement d'Amazon EKS pour votre application, puis vous confirmez que l'application est en cours d'exécution et qu'elle peut se connecter à Amazon Keyspaces.

Pour déployer une application sur Amazon EKS, vous devez configurer tous les paramètres pertinents dans un fichier appelé`deployment.yaml`. Ce fichier est ensuite utilisé par Amazon EKS pour déployer l'application. Les métadonnées du fichier doivent contenir les informations suivantes :

- Nom de l'application : nom de l'application. Pour ce tutoriel, nous utilisons`my-keyspaces-app`.
- Espace de noms Kubernetes : espace de noms du cluster Amazon EKS. Pour ce tutoriel, nous utilisons`my-eks-namespace`.
- Le nom du compte de service Amazon EKS est le nom du compte de service Amazon EKS. Pour ce tutoriel, nous utilisons`my-eks-serviceaccount`.
- nom de l'image : nom de l'image de l'application. Pour ce tutoriel, nous utilisons`my-keyspaces-app`.
- URI de l'image : URI de l'image Docker provenant d'Amazon ECR.
- AWS identifiant de compte : votre identifiant de AWS compte.
- ARN du rôle IAM : ARN du rôle IAM créé pour que le compte de service puisse l'assumer. Pour ce tutoriel, nous utilisons`my-iam-role`.
- Région AWS du cluster Amazon EKS dans lequel Région AWS vous avez créé votre cluster Amazon EKS.

Au cours de cette étape, vous déployez et exécutez l'application qui se connecte à Amazon Keyspaces et écrit des données dans la table.

1. Configurez le fichier `deployment.yaml`. Vous devez remplacer les valeurs suivantes :
    - `name`
    - `namespace`
    - `serviceAccountName`
    - `image`
    - `AWS_ROLE_ARN value`

- Le Région AWS in `CASSANDRA_HOST`
- `AWS_REGION`

Vous pouvez utiliser le fichier suivant comme exemple.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-keyspaces-app
  namespace: my-eks-namespace
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-keyspaces-app
  template:
    metadata:
      labels:
        app: my-keyspaces-app
    spec:
      serviceAccountName: my-eks-serviceaccount
      containers:
      - name: my-keyspaces-app
        image: 111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-
repository:latest
        ports:
        - containerPort: 8080
        env:
        - name: CASSANDRA_HOST
          value: "cassandra.aws-region.amazonaws.com:9142"
        - name: CASSANDRA_DC
          value: "aws-region"
        - name: AWS_WEB_IDENTITY_TOKEN_FILE
          value: /var/run/secrets/eks.amazonaws.com/serviceaccount/token
        - name: AWS_ROLE_ARN
          value: "arn:aws:iam::111122223333:role/my-iam-role"
        - name: AWS_REGION
          value: "aws-region"
```

2. Déployez `deployment.yaml`.

```
kubectl apply -f deployment.yaml
```

La sortie doit ressembler à ceci.

```
deployment.apps/my-keyspaces-app created
```

3. Vérifiez l'état du Pod dans votre espace de noms du cluster Amazon EKS.

```
kubectl get pods -n my-eks-namespace
```

La sortie doit ressembler à cet exemple.

```
NAME                    READY STATUS RESTARTS AGE
my-keyspaces-app-123abcde4f-g5hij 1/1 Running 0 75s
```

Pour plus de détails, vous pouvez utiliser la commande suivante.

```
kubectl describe pod my-keyspaces-app-123abcde4f-g5hij -n my-eks-namespace
```

```
Name:                 my-keyspaces-app-123abcde4f-g5hij
Namespace:            my-eks-namespace
Priority:             2000001000
Priority Class Name:  system-node-critical
Service Account:      my-eks-serviceaccount
Node:                 fargate-ip-192-168-102-209.ec2.internal/192.168.102.209
Start Time:           Thu, 23 Nov 2023 12:15:43 +0000
Labels:               app=my-keyspaces-app
                      eks.amazonaws.com/fargate-profile=my-fargate-profile
                      pod-template-hash=6c56fccc56
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
Status:               Running
IP:                   192.168.102.209
IPs:
  IP:           192.168.102.209
Controlled By:  ReplicaSet/my-keyspaces-app-6c56fccc56
Containers:
  my-keyspaces-app:
```

```
    Container ID:
 containerd://41ff7811d33ae4bc398755800abcdc132335d51d74f218ba81da0700a6f8c67b
    Image:          111122223333.dkr.ecr.aws-region.amazonaws.com/
my_eks_repository:latest
    Image ID:       111122223333.dkr.ecr.aws-region.amazonaws.com/
my_eks_repository@sha256:fd3c6430fc5251661efce99741c72c1b4b03061474940200d0524b84a951439c
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Thu, 23 Nov 2023 12:15:19 +0000
      Finished:     Thu, 23 Nov 2023 12:16:17 +0000
    Ready:          True
    Restart Count:  1
    Environment:
      CASSANDRA_HOST:               cassandra.aws-region.amazonaws.com:9142
      CASSANDRA_DC:                 aws-region
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
      AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-iam-role
      AWS_REGION:                   aws-region
      AWS_STS_REGIONAL_ENDPOINTS:   regional
    Mounts:
      /var/run/secrets/eks.amazonaws.com/serviceaccount from aws-iam-token (ro)
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-fssbf (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  aws-iam-token:
    Type:                     Projected (a volume that contains injected data from
 multiple sources)
    TokenExpirationSeconds:  86400
  kube-api-access-fssbf:
    Type:                     Projected (a volume that contains injected data from
 multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
```

```
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for
 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for
 300s
Events:
  Type     Reason           Age                From               Message
  ----     ------           ----               ----               -------
  Warning  LoggingDisabled  2m13s              fargate-scheduler  Disabled logging
 because aws-logging configmap was not found. configmap "aws-logging" not found
  Normal   Scheduled        89s                fargate-scheduler  Successfully
 assigned my-eks-namespace/my-keyspaces-app-6c56fccc56-mgs2m to fargate-
ip-192-168-102-209.ec2.internal
  Normal   Pulled           75s                kubelet
 Successfully pulled image "111122223333.dkr.ecr.aws-region.amazonaws.com/
my_eks_repository:latest" in 13.027s (13.027s including waiting)
  Normal   Pulling          54s (x2 over 88s)  kubelet            Pulling image
 "111122223333.dkr.ecr.aws-region.amazonaws.com/my_eks_repository:latest"
  Normal   Created          54s (x2 over 75s)  kubelet            Created container
 my-keyspaces-app
  Normal   Pulled           54s                kubelet
 Successfully pulled image "111122223333.dkr.ecr.aws-region.amazonaws.com/
my_eks_repository:latest" in 222ms (222ms including waiting)
  Normal   Started          53s (x2 over 75s)  kubelet            Started container
 my-keyspaces-app
```

4. Consultez les journaux du Pod pour vérifier que votre application est en cours d'exécution et qu'elle peut se connecter à votre table Amazon Keyspaces. Vous pouvez le faire à l'aide de la commande suivante. Assurez-vous de remplacer le nom de votre déploiement.

```
kubectl logs -f my-keyspaces-app-123abcde4f-g5hij -n my-eks-namespace
```

Vous devriez pouvoir voir les entrées du journal de l'application confirmant la connexion à Amazon Keyspaces, comme dans l'exemple ci-dessous.

```
2:47:20.553 [s0-admin-0] DEBUG c.d.o.d.i.c.metadata.MetadataManager
 - [s0] Adding initial contact points [Node(endPoint=cassandra.aws-
region.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)]
22:47:20.562 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection - [s0] Initializing
 with event types [SCHEMA_CHANGE, STATUS_CHANGE, TOPOLOGY_CHANGE]
22:47:20.564 [s0-admin-1] DEBUG c.d.o.d.i.core.context.EventBus - [s0] Registering
 com.datastax.oss.driver.internal.core.metadata.LoadBalancingPolicyWrapper$$Lambda
```

```
$812/0x0000000801105e88@769afb95 for class
 com.datastax.oss.driver.internal.core.metadata.NodeStateEvent
22:47:20.566 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection -
 [s0] Trying to establish a connection to Node(endPoint=cassandra.us-
east-1.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)
```

5. Exécutez la requête CQL suivante sur votre table Amazon Keyspaces pour confirmer qu'une ligne de données a été écrite dans votre table :

```
SELECT * from aws.user;
```

Vous devriez voir la sortie suivante :

```
fname    | lname | username | last_update_date
----------+-------+----------+---------------------------
random    | k     | test     | 2023-12-07 13:58:31.57+0000
```

Étape 5 : Nettoyage (facultatif)

Suivez ces étapes pour supprimer toutes les ressources créées dans ce didacticiel.

Supprimer les ressources créées dans ce didacticiel

1. Supprimez votre déploiement. Pour ce faire, vous pouvez utiliser la commande suivante.

```
kubectl delete deployment my-keyspaces-app -n my-eks-namespace
```

2. Supprimez le cluster Amazon EKS et tous les pods qu'il contient. Cela supprime également les ressources associées telles que le compte de service et le fournisseur d'identité OIDC. Pour ce faire, vous pouvez utiliser la commande suivante.

```
eksctl delete cluster --name my-eks-cluster --region aws-region
```

3. Supprimez le rôle IAM utilisé pour le compte de service Amazon EKS avec des autorisations d'accès à Amazon Keyspaces. Tout d'abord, vous devez supprimer la politique gérée attachée au rôle.

```
aws iam detach-role-policy --role-name my-iam-role --policy-arn
 arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
```

Vous pouvez ensuite supprimer le rôle à l'aide de la commande suivante.

```
aws iam delete-role --role-name my-iam-role
```

Pour plus d'informations, consultez la section Suppression d'un rôle IAM (AWS CLI) dans le guide de l'utilisateur IAM.

4. Supprimez le référentiel Amazon ECR, y compris toutes les images qui y sont stockées. Vous pouvez le faire à l'aide de la commande suivante.

```
aws ecr delete-repository \
      --repository-name my-ecr-repository \
      --force \
      --region aws-region
```

Notez que l'`force`indicateur est nécessaire pour supprimer un dépôt contenant des images. Pour supprimer d'abord votre image, vous pouvez le faire à l'aide de la commande suivante.

```
aws ecr batch-delete-image \
      --repository-name my-ecr-repository \
      --image-ids imageTag=latest \
      --region aws-region
```

Pour plus d'informations, consultez Supprimer une image dans le guide de l'utilisateur d'Amazon Elastic Container Registry.

5. Supprimez le keyspace et le tableau Amazon Keyspaces. La suppression de l'espace-clavier entraîne automatiquement la suppression de toutes les tables de cet espace-clavier. Pour ce faire, vous pouvez utiliser l'une des options suivantes.

   AWS CLI

   ```
   aws keyspaces delete-keyspace --keyspace-name 'aws'
   ```

   Pour confirmer que le keyspace a été supprimé, vous pouvez utiliser la commande suivante.

   ```
   aws keyspaces list-keyspaces
   ```

   Pour supprimer d'abord le tableau, vous pouvez utiliser la commande suivante.

```
aws keyspaces delete-table --keyspace-name 'aws' --table-name 'user'
```

Pour confirmer que votre table a été supprimée, vous pouvez utiliser la commande suivante.

```
aws keyspaces list-tables --keyspace-name 'aws'
```

Pour plus d'informations, voir supprimer un espace de touches et supprimer une table dans le manuel de référence des AWS CLI commandes.

cqlsh

```
DROP KEYSPACE IF EXISTS "aws";
```

Pour vérifier que vos espaces de touches ont été supprimés, vous pouvez utiliser l'instruction suivante.

```
SELECT * FROM system_schema.keyspaces ;
```

Votre espace de touche ne doit pas être répertorié dans le résultat de cette instruction. Notez qu'il peut y avoir un délai avant que les espaces clés ne soient supprimés. Pour de plus amples informations, veuillez consulter the section called “DROP KEYSPACE”.

Pour supprimer d'abord le tableau, vous pouvez utiliser la commande suivante.

```
DROP TABLE "aws.user"
```

Pour confirmer que votre table a été supprimée, vous pouvez utiliser la commande suivante.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

Votre table ne doit pas être répertoriée dans le résultat de cette instruction. Notez qu'il peut y avoir un délai avant que la table ne soit supprimée. Pour de plus amples informations, veuillez consulter the section called “DROP TABLE”.

Tutoriel : Connexion à Amazon Keyspaces à l'aide d'un point de terminaison VPC d'interface

Ce didacticiel explique comment configurer et utiliser un point de terminaison VPC d'interface pour Amazon Keyspaces.

Les points de terminaison VPC d'interface permettent une communication privée entre votre cloud privé virtuel (VPC) exécuté dans Amazon VPC et Amazon Keyspaces. Les points de terminaison VPC d'interface sont alimentés par AWS PrivateLink un AWS service qui permet une communication privée entre les VPC et les services. AWS Pour plus d'informations, consultez the section called "Utilisation des points de terminaison de VPC d'interface ".

Rubriques

Conditions préalables et considérations relatives au didacticiel

Avant de commencer ce didacticiel, suivez les instructions de AWS configuration indiquées dansAccès à Amazon Keyspaces (pour Apache Cassandra). Ces étapes incluent l'inscription AWS et la création d'un AWS Identity and Access Management (IAM) principal ayant accès à Amazon Keyspaces. Prenez note du nom de l'utilisateur IAM et des clés d'accès, car vous en aurez besoin plus tard dans ce didacticiel.

Créez un espace de touches avec le nom `myKeyspace` et au moins une table pour tester la connexion à l'aide du point de terminaison VPC plus loin dans ce didacticiel. Vous trouverez des instructions détaillées dansPremiers pas.

Après avoir effectué les étapes préalables, passez àÉtape 1 : lancer une instance Amazon EC2.

Étape 1 : lancer une instance Amazon EC2

Au cours de cette étape, vous allez lancer une instance Amazon EC2 dans votre VPC Amazon par défaut. Vous pouvez ensuite créer et utiliser un point de terminaison VPC pour Amazon Keyspaces.

Pour lancer une instance Amazon EC2

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Lancer une instance , puis procédez comme suit :

   Dans le tableau de bord de la console EC2, dans la zone Launch instance, choisissez Launch instance, puis choisissez Launch instance parmi les options qui s'affichent.

   Sous Nom et balises, pour Nom, entrez un nom descriptif pour votre instance.

   Sous Images de l'application et du système d'exploitation (Amazon Machine Image) :

   - Choisissez Quick Start, puis Ubuntu. Il s'agit du système d'exploitation (OS) de votre instance.
   - Sous Amazon Machine Image (AMI), vous pouvez utiliser l'image par défaut marquée comme éligible au niveau gratuit. Une Amazon Machine Image (AMI) est une configuration de base qui sert de modèle à votre instance.

   Sous Type d'instance :

   - Dans la liste Type d'instance, choisissez le type d'instance t2.micro, qui est sélectionné par défaut.

   Sous Paire de clés (connexion), pour Nom de la paire de clés, choisissez l'une des options suivantes pour ce didacticiel :

   - Si vous n'avez pas de paire de clés Amazon EC2, choisissez Create a new key pair (Créer une paire de clés), puis suivez les instructions. Il vous sera demandé de télécharger un fichier de clé privée (fichier .pem). Vous aurez besoin de ce fichier ultérieurement lorsque vous vous connecterez à votre instance Amazon EC2. Prenez donc note du chemin du fichier.
   - Si vous possédez déjà une paire de clés Amazon EC2, accédez à Select a key pair (Sélectionner une paire de clés), puis choisissez votre paire de clés dans la liste. Vous devez déjà posséder le fichier de clé privée (fichier .pem) pour pouvoir vous connecter à votre instance Amazon EC2.

Sous Paramètres réseau :

- Choisissez Modifier.
- Choisissez Select an existing security group.
- Dans la liste des groupes de sécurité, choisissez default. Il s'agit du groupe de sécurité par défaut pour votre VPC.

Passez au résumé.

- Consultez un résumé de la configuration de votre instance dans le panneau Résumé. Une fois que vous êtes prêt, choisissez Lancer l'instance.

3. Sur l'écran de finalisation de la nouvelle instance Amazon EC2, choisissez la vignette Connect to instance. L'écran suivant affiche les informations nécessaires et les étapes requises pour vous connecter à votre nouvelle instance. Prenez note des informations suivantes :

   - Exemple de commande pour protéger le fichier clé
   - La chaîne de connexion
   - Le nom DNS IPv4 public

   Après avoir pris note des informations de cette page, vous pouvez passer à l'étape suivante de ce didacticiel (Étape 2 : configurer votre instance Amazon EC2).

> **Note**
>
> Votre instance Amazon EC2 devient disponible en l'espace de quelques minutes. Avant de passer à l'étape suivante, assurez-vous que l'Instance State (État de l'instance) est `running` et que tous ses Status Checks (Contrôles d'état) ont réussi.

Étape 2 : configurer votre instance Amazon EC2

Lorsque votre instance Amazon EC2 est disponible, vous pouvez vous y connecter et la préparer pour la première utilisation.

> **Note**
>
> Les étapes suivantes supposent que vous vous connectez à votre instance Amazon EC2 depuis un ordinateur exécutant Linux. Pour d'autres méthodes de connexion, consultez Connect to your Linux instance dans le guide de l'utilisateur Amazon EC2.

Pour configurer votre instance Amazon EC2

1. Vous devez autoriser le trafic SSH entrant vers votre instance Amazon EC2. Pour ce faire, créez un nouveau groupe de sécurité EC2, puis attribuez-le à votre instance EC2.

   a. Dans le panneau de navigation, choisissez Groupes de sécurité.

   b. Sélectionnez Créer un groupe de sécurité. Dans la fenêtre Créer un groupe de sécurité, procédez comme suit :

      - Nom du groupe de sécurité — Entrez le nom de votre groupe de sécurité. Par exemple : `my-ssh-access`
      - Description — Entrez une brève description du groupe de sécurité.
      - VPC — Choisissez votre VPC par défaut.
      - Dans la section Règles de trafic entrant, choisissez Ajouter une règle et procédez comme suit :
        - Type — Choisissez SSH.
        - Source — Choisissez mon adresse IP.
        - Choisissez Ajouter une règle.

      Au bas de la page, confirmez les paramètres de configuration et choisissez Create Security Group.

   c. Dans le panneau de navigation, sélectionnez Instances.

   d. Choisissez l'instance Amazon EC2 que vous avez lancée dans Étape 1 : lancer une instance Amazon EC2.

   e. Choisissez Actions, sélectionnez Sécurité, puis sélectionnez Modifier les groupes de sécurité.

   f. Dans Modifier les groupes de sécurité, sélectionnez le groupe de sécurité que vous avez créé précédemment dans cette procédure (par exemple,`my-ssh-access`). Le groupe de

sécurité `default` existant doit également être sélectionné. Confirmez les paramètres de configuration et choisissez Attribuer des groupes de sécurité.

2. Utilisez la commande suivante pour empêcher l'accès à votre fichier de clé privée. Si vous ignorez cette étape, la connexion échoue.

```
chmod 400 path_to_file/my-keypair.pem
```

3. Utilisez la commande `ssh` pour vous connecter à votre instance Amazon EC2, comme dans l'exemple suivant.

```
ssh -i path_to_file/my-keypair.pem ubuntu@public-dns-name
```

Vous devez spécifier votre fichier de clé privée (fichier .pem) et le nom DNS public de votre instance. (Consultez Étape 1 : lancer une instance Amazon EC2).

L'ID de connexion est `ubuntu`. Aucun mot de passe n'est requis.

Pour plus d'informations sur l'autorisation des connexions à votre instance Amazon EC2 et pour AWS CLI obtenir des instructions, consultez Autoriser le trafic entrant pour vos instances Linux dans le guide de l'utilisateur Amazon EC2.

4. Téléchargez et installez la dernière version du AWS Command Line Interface.

   a. Installer `unzip`.

   ```
   sudo apt install unzip
   ```

   b. Téléchargez le `zip` fichier avec le AWS CLI.

   ```
   curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
    "awscliv2.zip"
   ```

   c. Décompressez le fichier.

   ```
   unzip awscliv2.zip
   ```

   d. Installez le AWS CLI.

   ```
   sudo ./aws/install
   ```

   e. Confirmez la version de l' AWS CLI installation.

```
aws --version
```

La sortie doit se présenter comme suit :

```
aws-cli/2.9.19 Python/3.9.11 Linux/5.15.0-1028-aws exe/x86_64.ubuntu.22 prompt/
off
```

5. Configurez vos AWS informations d'identification, comme indiqué dans l'exemple suivant. Entrez l'ID de votre clé d' AWS accès, votre clé secrète et le nom de région par défaut lorsque vous y êtes invité.

```
aws configure

AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-east-1
Default output format [None]:
```

6. Vous devez utiliser une `cqlsh` connexion à Amazon Keyspaces pour confirmer que votre point de terminaison VPC a été correctement configuré. Si vous utilisez votre environnement local ou l'éditeur CQL d'Amazon Keyspaces dans le AWS Management Console, la connexion passe automatiquement par le point de terminaison public plutôt que par votre point de terminaison VPC. À utiliser `cqlsh` pour tester la connexion de votre point de terminaison VPC dans ce didacticiel, suivez les instructions de configuration dans. Utilisation `cqlsh` pour se connecter à Amazon Keyspaces

Vous êtes maintenant prêt à créer un point de terminaison VPC pour Amazon Keyspaces.

Étape 3 : créer un point de terminaison VPC pour Amazon Keyspaces

Au cours de cette étape, vous créez un point de terminaison VPC pour Amazon Keyspaces à l'aide du. AWS CLIPour créer le point de terminaison VPC à l'aide de la console VPC, vous pouvez suivre les instructions de création d'un point de terminaison VPC du guide.AWS PrivateLink Lorsque vous filtrez le nom du service, entrez**`Cassandra`**.

Pour créer un point de terminaison VPC à l'aide du AWS CLI

1. Avant de commencer, vérifiez que vous pouvez communiquer avec Amazon Keyspaces via son point de terminaison public.

```
aws keyspaces list-tables --keyspace-name 'myKeyspace'
```

Le résultat affiche une liste des tables Amazon Keyspaces contenues dans l'espace de touches spécifié. Si vous n'avez aucune table, la liste est vide.

```
{
    "tables": [
        {
            "keyspaceName": "myKeyspace",
            "tableName": "myTable1",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
catalog/table/myTable1"
        },
        {
            "keyspaceName": "myKeyspace",
            "tableName": "myTable2",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
catalog/table/myTable2"
        }
    ]
}
```

2. Vérifiez qu'Amazon Keyspaces est un service disponible pour créer des points de terminaison VPC dans la région actuelle. AWS (la commande est affichée en gras, suivie d'un exemple de sortie).

```
aws ec2 describe-vpc-endpoint-services

{
    "ServiceNames": [
        "com.amazonaws.us-east-1.cassandra",
        "com.amazonaws.us-east-1.cassandra-fips"
    ]
}
```

Dans l'exemple de sortie, Amazon Keyspaces est l'un des services disponibles. Vous pouvez donc procéder à la création d'un point de terminaison VPC pour ce service.

3. Déterminez votre identifiant de VPC.

```
aws ec2 describe-vpcs

{
    "Vpcs": [
        {
            "VpcId": "vpc-a1234bcd",
            "InstanceTenancy": "default",
            "State": "available",
            "DhcpOptionsId": "dopt-8454b7e1",
            "CidrBlock": "111.31.0.0/16",
            "IsDefault": true
        }
    ]
}
```

Dans l'exemple de sortie, l'ID de VPC est `vpc-a1234bcd`.

4. Utilisez un filtre pour recueillir des informations sur les sous-réseaux du VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-a1234bcd"

{
     {
    "Subnets":[
       {
          "AvailabilityZone":"us-east-1a",
          "AvailabilityZoneId":"use2-az1",
          "AvailableIpAddressCount":4085,
          "CidrBlock":"111.31.0.0/20",
          "DefaultForAz":true,
          "MapPublicIpOnLaunch":true,
          "MapCustomerOwnedIpOnLaunch":false,
          "State":"available",
          "SubnetId":"subnet-920aacf9",
          "VpcId":"vpc-a1234bcd",
          "OwnerId":"111122223333",
          "AssignIpv6AddressOnCreation":false,
          "Ipv6CidrBlockAssociationSet":[
```

```
            ],
            "SubnetArn":"arn:aws:ec2:us-east-1:111122223333:subnet/subnet-920aacf9",
            "EnableDns64":false,
            "Ipv6Native":false,
            "PrivateDnsNameOptionsOnLaunch":{
               "HostnameType":"ip-name",
               "EnableResourceNameDnsARecord":false,
               "EnableResourceNameDnsAAAARecord":false
            }
         },
         {
            "AvailabilityZone":"us-east-1c",
            "AvailabilityZoneId":"use2-az3",
            "AvailableIpAddressCount":4085,
            "CidrBlock":"111.31.32.0/20",
            "DefaultForAz":true,
            "MapPublicIpOnLaunch":true,
            "MapCustomerOwnedIpOnLaunch":false,
            "State":"available",
            "SubnetId":"subnet-4c713600",
            "VpcId":"vpc-a1234bcd",
            "OwnerId":"111122223333",
            "AssignIpv6AddressOnCreation":false,
            "Ipv6CidrBlockAssociationSet":[

            ],
            "SubnetArn":"arn:aws:ec2:us-east-1:111122223333:subnet/subnet-4c713600",
            "EnableDns64":false,
            "Ipv6Native":false,
            "PrivateDnsNameOptionsOnLaunch":{
               "HostnameType":"ip-name",
               "EnableResourceNameDnsARecord":false,
               "EnableResourceNameDnsAAAARecord":false
            }
         },
         {
            "AvailabilityZone":"us-east-1b",
            "AvailabilityZoneId":"use2-az2",
            "AvailableIpAddressCount":4086,
            "CidrBlock":"111.31.16.0/20",
            "DefaultForAz":true,
            "MapPublicIpOnLaunch":true,
```

```
        }
    ]
}
```

Dans l'exemple de sortie, deux ID de sous-réseau sont disponibles : `subnet-920aacf9` et`subnet-4c713600`.

5. Créez le point de terminaison de VPC. Pour le paramètre `--vpc-id`, spécifiez l'ID de VPC de l'étape précédente. Pour le `--subnet-id` paramètre, spécifiez les ID de sous-réseau de l'étape précédente. Utilisez le `--vpc-endpoint-type` paramètre pour définir le point de terminaison en tant qu'interface. Pour plus d'informations sur la commande, reportez-vous `create-vpc-endpoint` à la référence des AWS CLI commandes.

```
aws ec2 create-vpc-endpoint --vpc-endpoint-type Interface --vpc-id vpc-a1234bcd
 --service-name com.amazonaws.us-east-1.cassandra --subnet-id subnet-920aacf9
 subnet-4c713600

{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-000ab1cdef23456789",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-a1234bcd",
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-920aacf9",
            "subnet-4c713600"
        ],
        "Groups": [
            {
                "GroupId": "sg-ac1b0e8d",
                "GroupName": "default"
            }
        ],
        "IpAddressType": "ipv4",
        "DnsOptions": {
            "DnsRecordIpType": "ipv4"
        },
        "PrivateDnsEnabled": true,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
```

```
            "eni-043c30c78196ad82e",
            "eni-06ce37e3fd878d9fa"
        ],
        "DnsEntries": [
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz.cassandra.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1a.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1c.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1b.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1d.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "cassandra.us-east-1.amazonaws.com",
                "HostedZoneId": "ZONEIDPENDING"
            }
        ],
        "CreationTimestamp": "2023-01-27T16:12:36.834000+00:00",
        "OwnerId": "111122223333"
    }
}

}
```

Étape 4 : configurer les autorisations pour la connexion du point de terminaison VPC

Les procédures de cette étape montrent comment configurer les règles et les autorisations pour utiliser le point de terminaison VPC avec Amazon Keyspaces.

Pour configurer une règle entrante pour le nouveau point de terminaison afin d'autoriser le trafic entrant TCP

1. Dans la console Amazon VPC, sur le panneau de gauche, choisissez Endpoints et choisissez le point de terminaison que vous avez créé à l'étape précédente.
2. Choisissez Groupes de sécurité, puis choisissez le groupe de sécurité associé à ce point de terminaison.
3. Choisissez Règles entrantes, puis sélectionnez Modifier les règles entrantes.
4. Ajoutez une règle entrante dont le type est CQLSH/CASSANDRA. Cela définit automatiquement la plage de ports à 9142.
5. Pour enregistrer la nouvelle règle de trafic entrant, choisissez Enregistrer les règles.

Pour configurer les autorisations des utilisateurs IAM

1. Vérifiez que l'utilisateur IAM utilisé pour se connecter à Amazon Keyspaces dispose des autorisations appropriées. Dans AWS Identity and Access Management (IAM), vous pouvez utiliser la politique AWS gérée `AmazonKeyspacesReadOnlyAccess` pour accorder à l'utilisateur IAM un accès en lecture à Amazon Keyspaces.
    a. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
    b. Dans le tableau de bord de la console IAM, choisissez Users (Utilisateurs), puis votre utilisateur IAM dans la liste.
    c. Sur la page Récapitulatif, choisissez Ajouter des autorisations.
    d. Choisissez Attacher directement les stratégies existantes.
    e. Dans la liste des politiques, choisissez AmazonKeyspacesReadOnlyAccess, puis Next : Review.
    f. Choisissez Add permissions (Ajouter des autorisations).
2. Vérifiez que vous pouvez accéder à Amazon Keyspaces via le point de terminaison VPC.

```
aws keyspaces list-tables --keyspace-name 'my_Keyspace'
```

Si vous le souhaitez, vous pouvez essayer d'autres AWS CLI commandes pour Amazon Keyspaces. Pour plus d'informations, consultez la référence de la commande AWS CLI.

> **Note**
>
> Les autorisations minimales requises pour qu'un utilisateur ou un rôle IAM accède à Amazon Keyspaces sont des autorisations de lecture sur la table système, comme indiqué dans la politique suivante. Pour plus d'informations sur les autorisations basées sur des politiques, consultezthe section called "Exemples de politiques basées sur l'identité".

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:555555555555:/keyspace/system*"
         ]
      }
   ]
}
```

3. Accordez à l'utilisateur IAM un accès en lecture à l'instance Amazon EC2 avec le VPC.

   Lorsque vous utilisez Amazon Keyspaces avec des points de terminaison VPC, vous devez accorder à l'utilisateur ou au rôle IAM qui accède à Amazon Keyspaces des autorisations en lecture seule sur votre instance Amazon EC2 et au VPC pour recueillir les données des points de terminaison et d'interface réseau. Amazon Keyspaces enregistre ces informations dans le `system.peers` tableau et les utilise pour gérer les connexions.

> **Note**
>
> Les politiques gérées `AmazonKeyspacesFullAccess` incluent `AmazonKeyspacesReadOnlyAccess_v2` les autorisations requises pour permettre à Amazon Keyspaces d'accéder à l'instance Amazon EC2 afin de lire les informations sur les points de terminaison VPC d'interface disponibles.

a. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.

b. Sur le tableau de bord de la console IAM, sélectionnez Policies.

c. Choisissez Create policy, puis sélectionnez l'onglet JSON.

d. Copiez la politique suivante et choisissez Next : Tags.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource": "*"
      }
   ]
}
```

e. Choisissez Suivant : Réviser, entrez le nom `keyspacesVPCendpoint` de la politique, puis choisissez Créer une politique.

f. Dans le tableau de bord de la console IAM, choisissez Users (Utilisateurs), puis votre utilisateur IAM dans la liste.

g. Sur la page Récapitulatif, choisissez Ajouter des autorisations.

h. Choisissez Attacher directement les stratégies existantes.

i. Dans la liste des politiques, choisissez KeyspacesVPCEndpoint, puis Next : Review.

j. Choisissez Add permissions (Ajouter des autorisations).

4. Pour vérifier que la `system.peers` table Amazon Keyspaces est mise à jour avec les informations VPC, exécutez la requête suivante depuis votre instance Amazon EC2 à l'aide de. `cqlsh` Si vous ne l'avez pas déjà installé `cqlsh` sur votre instance Amazon EC2 à l'étape 2, suivez les instructions de. the section called “Utilisation de l’`cqlsh-expansion`”

```
SELECT peer FROM system.peers;
```

La sortie renvoie des nœuds dotés d'adresses IP privées, en fonction de la configuration de votre VPC et de votre sous-réseau dans votre région. AWS

```
peer
---------------
112.11.22.123
112.11.22.124
112.11.22.125
```

> **Note**
>
> Vous devez utiliser une `cqlsh` connexion à Amazon Keyspaces pour confirmer que votre point de terminaison VPC a été correctement configuré. Si vous utilisez votre environnement local ou l'éditeur CQL d'Amazon Keyspaces dans le AWS Management Console, la connexion passe automatiquement par le point de terminaison public plutôt que par votre point de terminaison VPC. Si vous voyez neuf adresses IP, il s'agit des entrées qu'Amazon Keyspaces écrit automatiquement dans le `system.peers` tableau pour les connexions aux points de terminaison publics.

Étape 5 : Configuration de la surveillance avec CloudWatch

Cette étape explique comment utiliser Amazon CloudWatch pour surveiller la connexion du point de terminaison VPC à Amazon Keyspaces.

AWS PrivateLink publie des points de données CloudWatch concernant les points de terminaison de votre interface. Vous pouvez utiliser les métriques pour vérifier que le système fonctionne comme prévu. L'espace de `AWS/PrivateLinkEndpoints` noms CloudWatch inclut les métriques pour les points de terminaison de l'interface. Pour plus d'informations, consultez AWS PrivateLink les CloudWatch statistiques du AWS PrivateLink Guide.

Pour créer un CloudWatch tableau de bord avec des métriques de point de terminaison VPC

1. Ouvrez la CloudWatch console à l'adressehttps://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Dashboards (Tableaux de bord). Choisissez ensuite Créer un tableau de bord. Entrez un nom pour le tableau de bord et choisissez Create.
3. Sous Ajouter un widget, choisissez Numéro.
4. Sous Metrics, sélectionnez AWS/ PrivateLink Endpoints.
5. Choisissez le type de point de terminaison, le nom du service, l'ID de point de terminaison VPC, l'ID de VPC.
6. Sélectionnez les métriques `ActiveConnections` et `NewConnections` choisissez Create Widget.
7. Enregistrez le tableau de bord.

La `ActiveConnections` métrique est définie comme le nombre de connexions actives simultanées que le terminal a reçues au cours de la dernière minute. La `NewConnections` métrique est définie comme le nombre de nouvelles connexions établies via le point de terminaison au cours de la dernière minute.

Pour plus d'informations sur la création de tableaux de bord, voir Créer un tableau de bord dans le guide de CloudWatch l'utilisateur.

Étape 6 : (Facultatif) Meilleures pratiques pour configurer la taille du pool de connexions pour votre application

Dans cette section, nous expliquons comment déterminer la taille idéale du pool de connexions en fonction des exigences de débit de requêtes de votre application.

Amazon Keyspaces autorise un maximum de 3 000 requêtes CQL par seconde et par connexion TCP. Il n'y a donc pratiquement aucune limite quant au nombre de connexions qu'un conducteur peut établir avec Amazon Keyspaces. Toutefois, nous vous recommandons d'adapter la taille du pool de connexions aux exigences de votre application et de prendre en compte les points de terminaison disponibles lorsque vous utilisez Amazon Keyspaces avec des connexions de point de terminaison VPC.

Vous configurez la taille du pool de connexions dans le pilote client. Par exemple, sur la base d'un pool local de 2 et d'un point de terminaison d'interface VPC créé dans 3 zones de disponibilité, le pilote établit 6 connexions pour les requêtes (7 au total, y compris une connexion de contrôle). À

l'aide de ces 6 connexions, vous pouvez prendre en charge un maximum de 18 000 requêtes CQL par seconde.

Si votre application doit prendre en charge 40 000 requêtes CQL par seconde, revenez au nombre de requêtes nécessaires pour déterminer la taille du pool de connexions requis. Pour prendre en charge 40 000 requêtes CQL par seconde, vous devez configurer la taille du pool local sur au moins 5, ce qui prend en charge un minimum de 45 000 requêtes CQL par seconde.

Vous pouvez contrôler si vous dépassez le quota du nombre maximum d'opérations par seconde et par connexion en utilisant la `PerConnectionRequestRateExceeded` CloudWatch métrique dans l'espace de `AWS/Cassandra` noms. La `PerConnectionRequestRateExceeded` métrique indique le nombre de demandes adressées à Amazon Keyspaces qui dépassent le quota de taux de demandes par connexion.

Les exemples de code présentés dans cette étape montrent comment estimer et configurer le regroupement de connexions lorsque vous utilisez des points de terminaison VPC d'interface.

Java

Vous pouvez configurer le nombre de connexions par pool dans le pilote Java. Pour un exemple complet de connexion d'un pilote client Java, consultezthe section called “Utilisation d'un pilote client Cassandra Java”.

Lorsque le pilote client est démarré, la connexion de contrôle est d'abord établie pour les tâches administratives, telles que les modifications de schéma et de topologie. Ensuite, les connexions supplémentaires sont créées.

Dans l'exemple suivant, la configuration du pilote de taille de pool local est spécifiée comme 2. Si le point de terminaison du VPC est créé sur 3 sous-réseaux au sein du VPC, cela donne 7 points CloudWatch pour le point de terminaison de l'interface, comme indiqué `NewConnections` dans la formule suivante.

```
NewConnections = 3 (VPC subnet endpoints created across) * 2 (pool size) + 1
 ( control connection)
```

```
datastax-java-driver {

    basic.contact-points = [ "cassandra.us-east-1.amazonaws.com:9142"]
    advanced.auth-provider{
        class = PlainTextAuthProvider
```

```
            username = "ServiceUserName"
            password = "ServicePassword"
    }
    basic.load-balancing-policy {
        local-datacenter = "us-east-1"
        slow-replica-avoidance = false
    }

    advanced.ssl-engine-factory {
        class = DefaultSslEngineFactory
        truststore-path = "./src/main/resources/cassandra_truststore.jks"
        truststore-password = "my_password"
        hostname-validation = false
      }
    advanced.connection {
        pool.local.size = 2
          }
}
```

Si le nombre de connexions actives ne correspond pas à la taille de pool que vous avez configurée (agrégation entre sous-réseaux) +1 connexion de contrôle, cela signifie que quelque chose empêche la création des connexions.

Node.js

Vous pouvez configurer le nombre de connexions par pool dans le pilote Node.js. Pour un exemple complet de connexion au pilote client Node.js, consultezthe section called “Utilisation d'un pilote client Cassandra Node.js”.

Dans l'exemple de code suivant, la configuration du pilote de taille de pool local est spécifiée comme 1. Si le point de terminaison du VPC est créé sur 4 sous-réseaux au sein du VPC, cela donne 5 points CloudWatch pour le point de terminaison de l'interface, comme indiqué `NewConnections` dans la formule suivante.

```
NewConnections = 4 (VPC subnet endpoints created across) * 1 (pool size) + 1
 ( control connection)
```

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const types = cassandra.types;
const auth = new cassandra.auth.PlainTextAuthProvider('ServiceUserName',
 'ServicePassword');
```

```
const sslOptions1 = {
         ca: [
                    fs.readFileSync('/home/ec2-user/sf-class2-root.crt', 'utf-8')],
                    host: 'cassandra.us-east-1.amazonaws.com',
                    rejectUnauthorized: true
        };
const client = new cassandra.Client({
                   contactPoints: ['cassandra.us-east-1.amazonaws.com'],
                   localDataCenter: 'us-east-1',
                   pooling: { coreConnectionsPerHost: { [types.distance.local]:
 1 } },
                   consistency: types.consistencies.localQuorum,
                   queryOptions: { isIdempotent: true },
                   authProvider: auth,
                   sslOptions: sslOptions1,
                   protocolOptions: { port: 9142 }
        });
```

Étape 7 : (Facultatif) Nettoyer

Si vous souhaitez supprimer les ressources que vous avez créées dans ce didacticiel, suivez ces procédures.

Pour supprimer votre point de terminaison VPC pour Amazon Keyspaces

1. Connectez-vous à votre instance Amazon EC2.
2. Déterminez l'ID de point de terminaison VPC utilisé pour Amazon Keyspaces. Si vous omettez les `grep` paramètres, les informations de point de terminaison VPC sont affichées pour tous les services.

```
aws ec2 describe-vpc-endpoint-services | grep ServiceName | grep cassandra

{
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",
        "VpcId": "vpc-0bbc736e",
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "RouteTableIds": [],
        "VpcEndpointId": "vpce-9b15e2f2",
```

```
        "CreationTimestamp": "2017-07-26T22:00:14Z"
    }
}
```

Dans l'exemple de sortie, l'ID du point de terminaison de VPC est `vpce-9b15e2f2`.

3. Supprimez le point de terminaison de VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-9b15e2f2

{
    "Unsuccessful": []
}
```

Le tableau vide `[]` indique le succès de l'opération (il n'y a pas eu de demande infructueuse).

Pour résilier votre instance Amazon EC2

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, sélectionnez Instances.
3. Sélectionnez votre instance Amazon EC2.
4. Choisissez Actions, choisissez Instance State, puis Terminate.
5. Dans la fenêtre de confirmation, choisissez Yes, Terminate (Oui, Résilier).

Configuration de l'accès entre comptes pour Amazon Keyspaces

Vous pouvez créer et utiliser des ressources distinctesComptes AWS pour isoler les ressources et les utiliser dans différents environnements, par exemple en développement et en production. Cette rubrique vous explique comment accéder entre comptes à Amazon Keyspaces à l'aide des points de terminaison VPC de l'interface dans unAmazon Virtual Private Cloud. Pour de plus amples informations sur la configuration de l'accès entre comptes dans le Guide de l'utilisateur IAM, veuillez consulter Exemple de scénario utilisant des comptes de développement et de production séparés dans le Guide de l'utilisateur IAM.

Pour de plus amples informations sur Amazon Keyspaces et les points de terminaison VPC privés, veuillez consulterthe section called “Utilisation des points de terminaison de VPC d'interface ”.

Rubriques

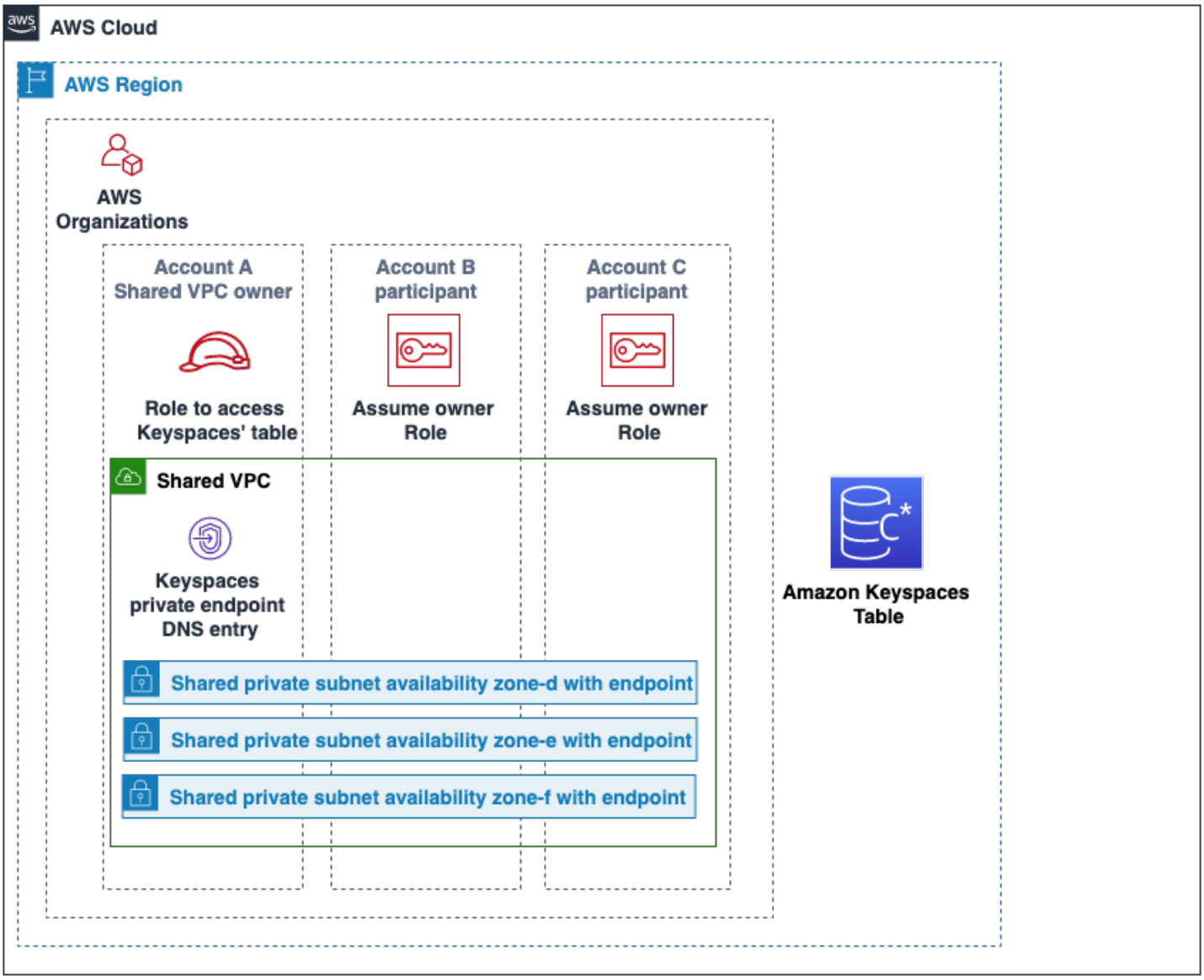
Configuration de l'accès entre comptes pour Amazon Keyspaces dans un VPC partagé

Vous pouvez créer des ressources différentesComptes AWS pour les séparer des applications. Par exemple, vous pouvez créer un compte pour vos tables Amazon Keyspaces, un compte différent pour les applications d'un environnement de développement et un autre compte pour les applications d'un environnement de production. Cette rubrique explique les étapes de configuration requises pour configurer l'accès entre comptes à Amazon Keyspaces à l'aide des points de terminaison VPC de l'interface dans un VPC partagé.

Pour savoir comment configurer un point de terminaison VPC pour Amazon Keyspaces, consultezthe section called “Étape 3 : créer un point de terminaison VPC pour Amazon Keyspaces”.

Dans cet exemple, nous utilisons les trois comptes suivants dans un VPC partagé :

- `Account A`— Ce compte contient l'infrastructure, notamment les points de terminaison VPC, les sous-réseaux VPC et les tables Amazon Keyspaces.
- `Account B`— Ce compte contient une application dans un environnement de développement qui doit se connecter à la table Amazon Keyspaces dans`Account A`.
- `Account C`— Ce compte contient une application dans un environnement de production qui doit se connecter à la table Amazon Keyspaces dans`Account A`.



`Account A` est le compte qui contient les ressources auxquelles`Account B`et`Account C` vous devez accéder, tout`Account A` comme le compte de confiance. `Account B`et`Account C` sont les comptes dont les principaux utilisateurs ont besoin d'accéder aux ressources de`Account A`. Ce`Account C` sont donc`Account B` les comptes de confiance. Le compte de confiance accorde les autorisations aux comptes de confiance en partageant un rôle IAM. La procédure suivante présente les étapes de configuration à suivre dans`Account A`.

Configuration pour**`Account A`**

1. Permet AWS Resource Access Manager de créer un partage de ressources pour le sous-réseau et de partager le sous-réseau privé avec`Account B` et`Account C`.

`Account B`et`Account C` peuvent désormais voir et créer des ressources dans le sous-réseau qui a été partagé avec le compte en question.

2. Créez un point de terminaison VPC privé Amazon Keyspaces alimenté parAWS PrivateLink. Cela crée plusieurs points de terminaison sur des sous-réseaux partagés et des entrées DNS pour le point de terminaison du service Amazon Keyspaces.
3. Créez un espace clé et une table Amazon Keyspaces.
4. Créez un rôle IAM disposant d'un accès complet à la table Amazon Keyspaces, d'un accès en lecture aux tables système Amazon Keyspaces et capable de décrire les ressources Amazon EC2 VPC, comme indiqué dans l'exemple de politique suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountAccess",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcEndpoints",
                "cassandra:*"
            ],
            "Resource": "*"
        }
    ]
}
```

5. Configurez la politique de confiance des rôles IAM que`Account B`Account C vous pouvez considérer comme des comptes de confiance, comme indiqué dans l'exemple suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111111111111:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {}
        }
```

```
    ]
}
```

Pour de plus amples informations sur les stratégies IAM entre comptes, veuillez consulter Stratégies entre comptes, veuillez consulter Stratégies entre comptes dans le Guide de l'utilisateur IAM.

Configuration dans**`Account B`** et**`Account C`**

1. Dans`Account B` et`Account C`, créez de nouveaux rôles et associez la politique suivante qui permet au principal d'assumer le rôle partagé créé dans`Account A`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

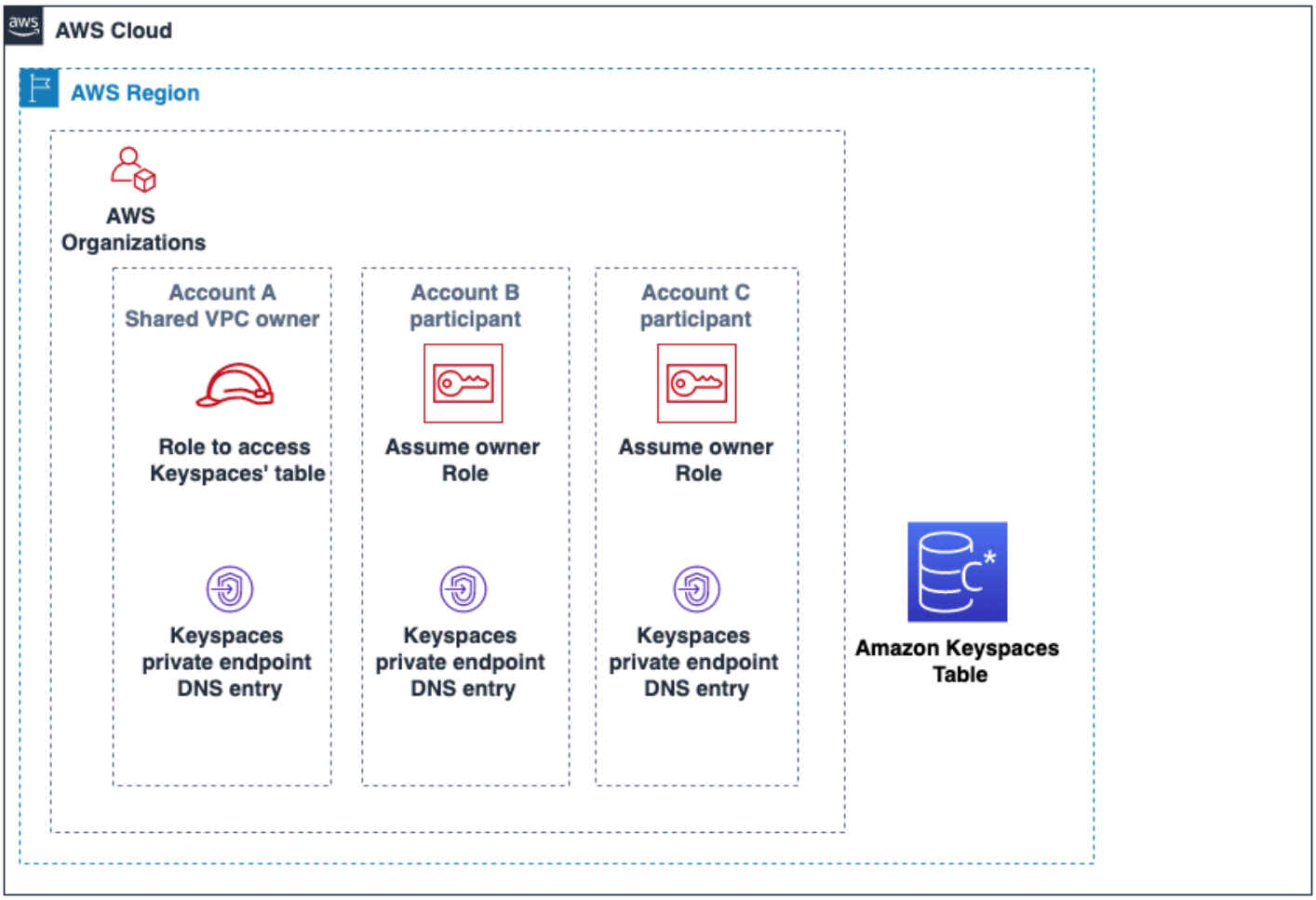
   Permettre au principal d'assumer le rôle partagé est implémenté à l'aide de l'`AssumeRole`API deAWS Security Token Service (AWS STS). Pour de plus amples informations, veuillez consulter Octroi de l'accès à unCompte AWS utilisateur IAM dans le Guide de l'utilisateur IAM.

2. Dans`Account B` et`Account C`, vous pouvez créer des applications qui utilisent le plug-in d'authentification SIGV4, qui permet à une application d'assumer le rôle partagé pour se connecter à la table Amazon Keyspaces située`Account A` via le point de terminaison du VPC dans le VPC partagé. Pour de plus amples informations sur le plug-in d'authentification SIGV4, veuillez consulterthe section called “Création d'informations d'identification”.

Configuration de l'accès entre comptes pour Amazon Keyspaces sans VPC partagé

Si la table Amazon Keyspaces et le point de terminaison VPC privé appartiennent à des comptes différents mais ne partagent pas un VPC, les applications peuvent toujours se connecter entre comptes à l'aide de points de terminaison VPC. Étant donné que les comptes ne partagent pas les points de terminaison VPC,`Account A`,`Account B`, et`Account C` ont besoin de leurs propres points de terminaison VPC. Pour le pilote du client Cassandra, Amazon Keyspaces apparaît comme un nœud unique au lieu d'un cluster à plusieurs nœuds. Lors de la connexion, le pilote client atteint le serveur DNS qui renvoie l'un des points de terminaison disponibles dans le VPC du compte.

Vous pouvez également accéder aux tables Amazon Keyspaces sur différents comptes sans point de terminaison VPC partagé en utilisant le point de terminaison public ou en déployant un point de terminaison VPC privé dans chaque compte. Lorsque vous n'utilisez pas de VPC partagé, chaque compte nécessite son propre point de terminaison VPC. Dans cet exemple`Account A``Account B`, et`Account C` ont besoin de leurs propres points de terminaison VPC pour accéder à la table dans`Account A`. Lorsque vous utilisez des points de terminaison VPC dans cette configuration, Amazon Keyspaces apparaît comme un cluster à nœud unique sur le pilote client Cassandra au lieu d'un cluster à plusieurs nœuds. Lors de la connexion, le pilote client atteint le serveur DNS qui renvoie l'un des points de terminaison disponibles dans le VPC du compte. Mais le pilote client n'est pas en mesure d'accéder au`system.peers` tableau pour découvrir des points de terminaison supplémentaires. Comme il y a moins d'hôtes disponibles, le pilote établit moins de connexions. Pour régler cela, multipliez par trois le paramètre du pool de connexions du pilote.


AWS Cloud
AWS Region
AWS Organizations
Account A
Shared VPC owner
Role to access Keyspaces' table
Keyspaces private endpoint DNS entry
Account B
participant
Assume owner Role
Keyspaces private endpoint DNS entry
Account C
participant
Assume owner Role
Keyspaces private endpoint DNS entry
Amazon Keyspaces Table

Commencer à utiliser Amazon Keyspaces (pour Apache Cassandra)

Ce didacticiel est pour vous si vous débutez avec Apache Cassandra et Amazon Keyspaces (pour Apache Cassandra). Dans ce didacticiel, vous allez installer tous les programmes et pilotes dont vous avez besoin pour utiliser correctement Amazon Keyspaces.

Pour des didacticiels sur la connexion programmatique à Amazon Keyspaces à l'aide de différents pilotes clients Cassandra, consultez. the section called “Utilisation d'un pilote client Cassandra”

Rubriques

Conditions préalables et considérations relatives au didacticiel

Avant de commencer ce didacticiel, suivez les instructions de AWS configuration indiquées dansAccès à Amazon Keyspaces (pour Apache Cassandra). Ces étapes incluent l'inscription AWS et la création d'un utilisateur AWS Identity and Access Management (IAM) ayant accès à Amazon Keyspaces.

En outre, si vous complétez le didacticiel à l'aide de `cqlsh` ou d'un pilote client Cassandra sous licence Apache 2.0, suivez les instructions d'installation de la section Utilisation `cqlsh` pour se connecter à Amazon Keyspaces.

Après avoir effectué les étapes préalables, passez àÉtape 1 du didacticiel : Création d'un keyspace et d'un tableau dans Amazon Keyspaces.

Étape 1 du didacticiel : Création d'un keyspace et d'un tableau dans Amazon Keyspaces

Dans cette section, vous allez créer un espace de touches et y ajouter un tableau à l'aide de la console.

> **Note**
>
> Avant de commencer, assurez-vous d'avoir configuré tous les prérequis du didacticiel.

Rubriques

- Création d'un keyspace
- Création d'une table

Création d'un keyspace

Un keyspace regroupe les tables associées qui sont pertinentes pour une ou plusieurs applications. Un keyspace contient une ou plusieurs tables et définit la stratégie de réplication pour toutes les tables qu'il contient. Pour de plus amples informations sur les keyspaces, veuillez consulter les rubriques suivantes :

- Utilisation des espaces de touches : the section called “Création d'espaces clés”
- Déclarations du langage de définition des données (DDL) : Keyspaces
- Quotas pour Amazon Keyspaces (pour Apache Cassandra)

Lorsque vous créez un keyspace, vous devez spécifier le nom du keyspace.

> **Note**
>
> La stratégie de réplication du keyspace doit être. `SingleRegionStrategy` `SingleRegionStrategy`réplique les données sur trois zones de disponibilité dans son Région AWS.

Utilisation de la console

Pour créer un keyspace à l'aide de la console

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces.

3. Choisissez Create keyspace.
4. Dans la zone Nom du keyspace entrez **`myGSGKeyspace`** comme nom de votre keyspace.

   Contraintes de nom :

   - Ne peut pas être vide.
   - Caractères autorisés : caractères alphanumériques et soulignement (_).
   - La longueur maximale est de 48 caractères.
5. Pour créer le keyspace, choisissez Create keyspace (Créer un keyspace).
6. Vérifiez que le keyspace `myGSGKeyspace` a été créé en procédant comme suit :

   a. Dans le panneau de navigation, choisissez Keyspaces.

   b. Localisez votre keyspace `myGSGKeyspace` dans la liste des keyspaces.

Utilisation de CQL

La procédure suivante crée un keyspace à l'aide de CQL.

Pour créer un keyspace avec CQL

1. Ouvrez un shell de commande et entrez ce qui suit :

   `cqlsh`
2. Créez votre keyspace à l'aide de la commande CQL suivante.

```
CREATE KEYSPACE IF NOT EXISTS "myGSGKeyspace"
   WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

   `SingleRegionStrategy`utilise un facteur de réplication de trois et réplique les données dans trois zones de AWS disponibilité de sa région.

   > **Note**
   >
   > Amazon Keyspaces met par défaut toutes les entrées en minuscules, sauf si vous les mettez entre guillemets. Dans ce cas, notez `"myGSGKeyspace"`.
3. Vérifiez que votre keyspace a été créé.

```
SELECT * from system_schema.keyspaces ;
```

Votre keyspace doit être répertorié.

Création d'une table

Une table est l'endroit où vos données sont organisées et stockées. La clé primaire de votre table détermine comment les données seront partitionnées dans votre table. La clé primaire est composée d'une clé de partition obligatoire et d'une ou plusieurs colonnes de clustering facultatives. Les valeurs combinées qui composent la clé primaire doivent être uniques pour toutes les données de la table. Pour de plus amples informations sur les balises, veuillez consulter les rubriques suivantes :

- Utilisation des tables : the section called “Création de tables”
- Déclarations DDL : Tables
- Gestion des ressources des tables : Gestion des ressources sans serveur
- Surveillance de l'utilisation des ressources des tables : the section called “Surveillance avec CloudWatch”
- Quotas pour Amazon Keyspaces (pour Apache Cassandra)

Lorsque vous créez une table, vous spécifiez les éléments suivants :

- Nom de la table.
- Nom et type de données de chaque colonne de la table.
- Clé primaire de la table.
  - Clé de partition — Obligatoire
  - Colonnes de regroupement — Facultatif

Utilisez la procédure suivante pour créer une table avec les colonnes, les types de données, la clé de partition et la colonne de clustering spécifiés.

Utilisation de la console

La procédure suivante crée la table `employees_tbl` avec ces colonnes et types de données.

```
ID           text
```

```
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Pour créer une table à l'aide de la console

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces.
3. Choisissez `myGSGKeyspace` comme keyspace dans lequel vous souhaitez créer la table.
4. Choisissez Créer un tableau.
5. Dans la zone Nom de la table entrez **`employees_tbl`** comme nom pour votre table.

   Contraintes de nom :

   - Ne peut pas être vide.
   - Caractères autorisés : caractères alphanumériques et soulignement (_).
   - La longueur maximale est de 48 caractères.
6. Dans la section Colonnes répétez les étapes suivantes pour chaque colonne à ajouter à cette table.

   Ajoutez les colonnes et types de données suivants.

   ```
   id            text
   name          text
   region        text
   division      text
   project       text
   role          text
   pay_scale     int
   vacation_hrs  float
   manager_id    text
   ```

   a. Nom — Entrez le nom de la colonne.

Contraintes de nom :

- Ne peut pas être vide.
- Caractères autorisés : caractères alphanumériques et soulignement (_).
- La longueur maximale est de 48 caractères.

b. Type : dans la liste des types de données, choisissez le type de données pour cette colonne.

c. Si vous souhaitez ajouter une autre colonne, choisissez Ajouter une colonne.

7. Choisissez `id` comme clé de partition sous Clé de partition. Une clé de partition est requise pour chaque table. Une clé de partition peut être constituée d'une ou plusieurs colonnes.
8. Ajoutez `division` comme colonne de mise en cluster. Les colonnes de clustering sont facultatives et déterminent l'ordre de tri dans chaque partition.

   a. Pour ajouter une colonne de clustering, choisissez Ajouter une colonne de clustering.

   b. Dans la liste Colonne choisissez division. Dans la liste Ordre choisissez ASC pour trier par ordre croissant les valeurs de cette colonne. (Choisissez DESC pour l'ordre décroissant.)

9. Dans la section Paramètres du tableau, choisissez Paramètres par défaut.
10. Choisissez Créer un tableau.
11. Vérifiez que votre table a été créée.

    a. Dans le volet de navigation, choisissez Tables.

    b. Confirmez que votre table figure dans la liste des tables.

    c. Choisissez le nom de votre table.

    d. Vérifiez que toutes vos colonnes et tous vos types de données sont corrects.

> **Note**
>
> Les colonnes peuvent ne pas être répertoriées dans le même ordre que celui où vous les avez ajoutées à la table.

    e. Dans la colonne Clustering, confirmez que Division a la valeur true. Toutes les autres colonnes de table doivent être false.

Utilisation de CQL

La procédure suivante crée une table avec les colonnes et types de données suivants à l'aide de CQL. La colonne `id` doit être la clé de partition.

```
id            text
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Pour créer une table avec CQL

1. Ouvrez un shell de commande et entrez ce qui suit :

   `cqlsh`

2. À l'invite `cqlsh` (`cqlsh>`), spécifiez un keyspace dans lequel créer votre table.

   ```
   USE "myGSGKeyspace" ;
   ```

3. À l'invite d'espace de touche (`cqlsh:keyspace_name>`), créez votre table en entrant le code suivant dans votre fenêtre de commande.

   ```
   CREATE TABLE IF NOT EXISTS "myGSGKeyspace".employees_tbl (
      id text,
      name text,
      region text,
      division text,
      project text,
      role text,
      pay_scale int,
      vacation_hrs float,
      manager_id text,
      PRIMARY KEY (id,division))
      WITH CLUSTERING ORDER BY (division ASC) ;
   ```

> **Note**
>
> ASC est l'ordre de mise en cluster par défaut. Vous pouvez également spécifier DESC pour l'ordre décroissant.

Notez que la colonne `id` doit être la clé de partition. Ensuite, `division` est la colonne de clustering ordonnée par ordre croissant (ASC).

4. Vérifiez que votre table a été créée.

```
SELECT * from system_schema.tables WHERE keyspace_name='myGSGKeyspace' ;
```

Votre table doit s'afficher.

5. Vérifiez la structure de votre table.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'myGSGKeyspace' AND
 table_name = 'employees_tbl' ;
```

Confirmez que toutes les colonnes et tous les types de données sont tels que vous l'escomptiez. L'ordre des colonnes peut être différent de celui de l'instruction `CREATE`.

Pour effectuer des opérations CRUD (création, lecture, mise à jour et suppression) sur les données de votre table, passez à the section called “Étape 2 : opérations CRUD”.

Étape 2 du didacticiel : créer, lire, mettre à jour et supprimer des données (CRUD)

Dans cette section, vous utilisez l'éditeur CQL de la console pour effectuer des opérations CRUD (création, lecture, mise à jour et suppression) sur les données de votre table. Vous pouvez également exécuter les commandes en utilisant `cqlsh`.

Rubriques

- Tutoriel : Insertion et chargement de données dans une table Amazon Keyspaces
- Tutoriel : lire à partir d'un tableau Amazon Keyspaces
- Tutoriel : Mettre à jour les données d'une table Amazon Keyspaces

- Tutoriel : Supprimer des données dans une table Amazon Keyspaces

Tutoriel : Insertion et chargement de données dans une table Amazon Keyspaces

Pour créer des données dans votre table `employees_tbl`, utilisez l'instruction `INSERT` pour ajouter une seule ligne.

1. Avant de pouvoir écrire des données dans votre table Amazon Keyspaces à l'aide de cqlsh, vous devez définir la cohérence d'écriture de la session cqlsh en cours sur. `LOCAL_QUORUM` Pour plus d'informations sur les niveaux de cohérence pris en charge, consultezthe section called “Niveaux de cohérence en écriture”. Notez que cette étape n'est pas obligatoire si vous utilisez l'éditeur CQL dans le AWS Management Console.

   ```
   CONSISTENCY LOCAL_QUORUM;
   ```

2. Pour insérer un seul enregistrement, exécutez la commande suivante dans l'éditeur CQL.

   ```
   INSERT INTO "myGSGKeyspace".employees_tbl (id, name, project, region, division,
    role, pay_scale, vacation_hrs, manager_id)
   VALUES ('012-34-5678','Russ','NightFlight','US','Engineering','IC',3,12.5,
    '234-56-7890') ;
   ```

3. Vérifiez que les données ont été correctement ajoutées à votre table en exécutant la commande suivante.

   ```
   SELECT * FROM "myGSGKeyspace".employees_tbl ;
   ```

Pour insérer plusieurs enregistrements à partir d'un fichier avec cqlsh

1. Téléchargez l'exemple de fichier de données (employees.csv) contenu dans le fichier d'archive suivant sampledata.zip. Ce fichier CSV (valeurs séparées par des virgules) contient les données suivantes. Rappelez-vous le chemin d'accès sur lequel vous enregistrez le fichier.

ID	name	project	region	division	role	pay_scale	vacation_hrs	manager_id
123-45-6789	Bob	NightFlight	US	Engineering	Intern	1	0	234-56-7890
234-56-7890	Bob	NightFlight	US	Engineering	Manager	6	72	789-01-2345
345-67-8901	Sarah	Storm	US	Engineering	IC	4	108	234-56-7890
456-78-9012	Beth	NightFlight	US	Engineering	IC	7	100.5	234-56-7890
567-89-0123	Ahmed	NightFlight	US	Marketing	IC	4	88	678-90-1234
678-90-1234	Alan	Storm	US	Marketing	Manager	3	18.4	789-01-2345
789-01-2345	Roberta	All	US	Executive	CEO	15	184	None

2. Ouvrez un shell de commande et entrez ce qui suit :

   `cqlsh`

3. À l'invite `cqlsh` (`cqlsh>`), spécifiez un keyspace.

```
USE "myGSGKeyspace" ;
```

4. Définissez la cohérence d'écriture sur`LOCAL_QUORUM`. Pour plus d'informations sur les niveaux de cohérence pris en charge, consultezthe section called “Niveaux de cohérence en écriture”.

```
CONSISTENCY LOCAL_QUORUM;
```

5. À l'invite du keyspace (`cqlsh:keyspace_name>`), exécutez la requête suivante.

```
COPY employees_tbl
 (id,name,project,region,division,role,pay_scale,vacation_hrs,manager_id)
FROM 'path-to-the-csv-file/employees.csv' WITH delimiter=',' AND header=TRUE ;
```

6. Vérifiez que les données ont été correctement ajoutées à votre table en exécutant la requête suivante.

```
SELECT * FROM employees_tbl ;
```

Tutoriel : lire à partir d'un tableau Amazon Keyspaces

Dans la section Tutoriel : Insertion et chargement de données dans une table Amazon Keyspaces, vous avez utilisé l'instruction `SELECT` pour vérifier que vous avez correctement ajouté des données à votre table. Dans cette section, vous affinez votre utilisation de `SELECT` pour afficher des colonnes spécifiques, et uniquement des lignes qui répondent à des critères spécifiques.

La forme générale de l'instruction `SELECT` est la suivante.

```
SELECT column_list FROM table_name [WHERE condition [ALLOW FILTERING]] ;
```

Rubriques

Sélection de toutes les données de votre tableau

La forme la plus simple de l'instruction `SELECT` renvoie toutes les données de votre table.

> **Important**
>
> Dans un environnement de production, il n'est généralement pas recommandé d'exécuter cette commande, qui renvoie toutes les données de votre table.

Pour sélectionner toutes les données de votre table

- Exécutez la requête suivante :

  ```
  SELECT * FROM "myGSGKeyspace".employees_tbl ;
  ```

  Utilisation du caractère générique (`*`) pour que `column_list` sélectionne toutes les colonnes.

Sélection d'un sous-ensemble de colonnes

Pour interroger un sous-ensemble de colonnes

- Pour récupérer uniquement les colonnes `id`, `name` et `manager_id`, exécutez la requête suivante.

  ```
  SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
  ```

  La sortie contiendra uniquement les colonnes spécifiées dans l'ordre indiqué dans l'instruction `SELECT`.

Sélection d'un sous-ensemble de lignes

Lors de l'interrogation d'un jeu de données volumineux, vous ne souhaitez peut-être que les enregistrements répondant à certains critères. Pour ce faire, vous pouvez ajouter une clause `WHERE` à la fin de notre instruction `SELECT`.

Pour interroger un sous-ensemble de lignes

- Pour récupérer uniquement l'enregistrement de l'employé avec l'ID `'234-56-7890'`, exécutez la requête suivante.

  ```
  SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='234-56-7890' ;
  ```

  L'instruction `SELECT` précédente renvoie uniquement les lignes où `id` a la valeur `234-56-7890`.

Comprendre la **`WHERE`** clause

La clause `WHERE` est utilisée pour filtrer les données et renvoyer uniquement celles qui répondent aux critères spécifiés. Les critères spécifiés peuvent être une condition simple ou une condition composée.

Comment utiliser les conditions dans une clause **`WHERE`**

- Une condition simple : une seule colonne.

  ```
  WHERE column_name=value
  ```

  Vous pouvez utiliser une condition simple dans une clause `WHERE` si l'une des conditions suivantes est remplie :

  - La colonne est la seule colonne de la clé primaire de la table.
  - Vous ajoutez `ALLOW FILTERING` après la condition de la clause `WHERE`.

    Sachez que l'utilisation d'`ALLOW FILTERING` peut entraîner des performances incohérentes, en particulier avec les tables volumineuses et à plusieurs partitions.
- Une condition composée — Plusieurs conditions simples reliées par`AND`.

  ```
  WHERE column_name1=value1 AND column_name2=value2 AND column_name3=value3...
  ```

Vous pouvez utiliser des conditions composées dans une clause `WHERE` si l'une des conditions suivantes est remplie :

- Les colonnes de la `WHERE` clause correspondent exactement aux colonnes de la clé primaire de la table, ni plus ni moins.
- Vous ajoutez `ALLOW FILTERING` après la condition composée dans la clause `WHERE`, comme dans l'exemple suivant.

  ```
  SELECT * FROM my_table WHERE col1=5 AND col2='Bob' ALLOW FILTERING ;
  ```

  Sachez que l'utilisation d'`ALLOW FILTERING` peut entraîner des performances incohérentes, en particulier avec les tables volumineuses et à plusieurs partitions.

Essayez-le

Créez vos propres requêtes CQL pour trouver les éléments suivants dans votre table `employees_tbl` :

- Trouvez les `name`, `project` et `id` de tous les employés.
- Trouvez sur quel projet le stagiaire Bob travaille (incluez au moins son nom, son projet et son rôle dans la sortie).
- Avancé : créez une application pour trouver tous les employés qui ont le même manager que Bob le stagiaire. ASTUCE : plusieurs requêtes peuvent être nécessaires.
- Avancé : créez une application pour rechercher les colonnes sélectionnées de tous les employés travaillant sur le projet `NightFlight`. ASTUCE : la résolution du problème peut nécessiter plusieurs instructions.

Tutoriel : Mettre à jour les données d'une table Amazon Keyspaces

Pour mettre à jour les données de votre table `employees_tbl`, utilisez l'instruction `UPDATE`.

La forme générale de l'instruction `UPDATE` est la suivante.

```
UPDATE table_name SET column_name=new_value WHERE primary_key=value ;
```

Tip

- Vous pouvez mettre à jour plusieurs colonnes à l'aide d'une liste de `column_names` et de valeurs séparées par des virgules, comme dans l'exemple suivant.

```
UPDATE my_table SET col1='new_value_1', col2='new_value2' WHERE id='12345' ;
```

- Si la clé primaire est composée de plusieurs colonnes, toutes les colonnes de clé primaire et leurs valeurs doivent être incluses dans la clause `WHERE`.
- Vous ne pouvez pas mettre à jour une colonne de la clé primaire car cela modifierait la clé primaire de l'enregistrement.

Pour mettre à jour une cellule unique

En utilisant votre table `employees_tbl`, donnez une augmentation à l'employé ayant l'id `567-89-0123`.

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale=5 WHERE id='567-89-0123' AND
 division='Marketing' ;
```

Vérifiez que l'échelle de rémunération de l'employé est maintenant 5.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='567-89-0123' ;
```

Essayez-le

Avancé : votre entreprise a engagé Bob le stagiaire. Changez son enregistrement afin que son rôle soit `'IC'` et son échelle salariale 2.

Tutoriel : Supprimer des données dans une table Amazon Keyspaces

Pour supprimer les données de votre table `employees_tbl`, utilisez l'instruction `DELETE`.

Vous pouvez supprimer des données d'une ligne ou d'une partition. Soyez prudent lorsque vous supprimez des données, car les suppressions sont irréversibles.

La suppression d'une ou de toutes les lignes d'une table ne supprime pas la table. Ainsi, vous pouvez le remplir à nouveau avec des données. La suppression d'une table supprime la table et toutes les

données qu'elle contient. Pour réutiliser la table, vous devez la recréer et y ajouter des données. La suppression d'un keyspace supprime le keyspace et toutes les tables qu'il contient. Pour utiliser le keyspace et les tables, vous devez les recréer, puis les remplir avec des données.

Supprimer des cellules

La suppression d'une colonne d'une ligne supprime les données de la cellule spécifiée. Lorsque vous affichez cette colonne à l'aide d'une instruction `SELECT` instruction, les données sont affichées comme *`null`*, bien qu'une valeur nulle ne soit pas stockée à cet emplacement.

La syntaxe générale permettant de supprimer une ou plusieurs colonnes spécifiques est la suivante.

```
DELETE column_name1[, column_name2...] FROM table_name WHERE condition ;
```

Dans votre table `employees_tbl`, vous pouvez voir que le PDG a `"None"` comme manager. Tout d'abord, supprimez cette cellule afin que vous n'y transportiez aucune donnée.

Pour supprimer une cellule spécifique

1. Exécutez la requête `DELETE` suivante :

   ```
   DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
    division='Executive';
   ```

2. Vérifiez que la suppression a été effectuée comme prévu.

   ```
   SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
    division='Executive';
   ```

Supprimer des lignes

Il se peut que vous deviez supprimer une ligne entière, par exemple lorsqu'un employé prend sa retraite. La syntaxe générale pour supprimer une ligne est la suivante.

```
DELETE FROM table_name WHERE condition ;
```

Pour supprimer une ligne

1. Exécutez la requête `DELETE` suivante :

```
DELETE FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
 division='Engineering';
```

2. Vérifiez que la suppression a été effectuée comme prévu.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
 division='Engineering';
```

Étape 3 du didacticiel : supprimer une table et un espace de touches dans Amazon Keyspaces

Pour éviter d'être facturé pour les tables et les données dont vous n'avez pas besoin, supprimez toutes les tables et les keyspaces que vous n'utilisez pas. Lorsque vous supprimez une table, la table et ses données sont supprimées et vous arrêtez d'accumuler des frais pour eux. Cependant, le keyspace demeure. Lorsque vous supprimez un keyspace, le keyspace et toutes ses tables sont supprimés et vous cessez d'accumuler des frais pour eux.

Suppression d'une table

Vous pouvez supprimer une table à l'aide de la console ou du CQL. Lorsque vous supprimez une table, la table et toutes ses données sont supprimées.

Utilisation de la console

La procédure suivante supprime une table et toutes ses données avec AWS Management Console.

Pour supprimer une table à l'aide de la console

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le volet de navigation, choisissez Tables.
3. Sélectionnez la case située à gauche du nom de chaque table à supprimer.
4. Sélectionnez Delete (Supprimer).
5. Dans l'écran Supprimer la table entrez **`Delete`** dans la zone. Puis, choisissez Delete Table (Supprimer la table).

6. Pour vérifier que la table a été supprimée, choisissez Tables dans le panneau de navigation et vérifiez que la table `employees_tbl` n'est plus répertoriée.

Utilisation de CQL

La procédure suivante supprime une table et toutes ses données à l'aide de CQL.

Pour supprimer une table avec CQL

1. Ouvrez un shell de commande et entrez ce qui suit :

   `cqlsh`

2. Supprimez votre table en entrant la commande suivante à l'invite de keyspace (`cqlsh:`*`keyspace_name`*`>`).

   ```
   DROP TABLE IF EXISTS "myGSGKeyspace".employees_tbl ;
   ```

3. Vérifiez que votre table a été supprimée.

   ```
   SELECT * FROM system_schema.tables WHERE keyspace_name = 'myGSGKeyspace' ;
   ```

   Votre table ne doit pas s'afficher.

Supprimer un keyspace

Vous pouvez supprimer un espace de touches à l'aide du code CQL AWS Management Console ou du code CQL. Lorsque vous supprimez un keyspace, le keyspace ainsi que toutes ses tables et données sont supprimés.

Utilisation de l' AWS Management Console

La procédure suivante supprime un keyspace ainsi que toutes ses tables et données avec AWS Management Console.

Pour supprimer un keyspace à l'aide de la console

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces.

3. Sélectionnez la case située à gauche du nom de chaque keyspace que vous souhaitez supprimer.
4. Sélectionnez Delete (Supprimer).
5. Dans l'écran Supprimer le keyspace entrez **`Delete`** dans la zone. Ensuite, choisissez Supprimer le keyspace.
6. Pour vérifier que le keyspace `myGSGKeyspace` a été supprimé, choisissez Keyspaces dans le panneau de navigation et vérifiez qu'il n'est plus répertorié. Étant donné que vous avez supprimé son keyspace, la table `employees_tbl` sous Tables ne doit pas non plus être répertoriée.

Utilisation de CQL

La procédure suivante supprime un keyspace et toutes ses tables et données à l'aide de CQL.

Pour supprimer un keyspace avec CQL

1. Ouvrez un shell de commande et entrez ce qui suit :

   `cqlsh`
2. Supprimez votre keyspace en entrant la commande suivante à l'invite de keyspace (`cqlsh:`*`keyspace_name`*`>`).

   ```
   DROP KEYSPACE IF EXISTS "myGSGKeyspace" ;
   ```
3. Vérifiez que votre keyspace a été supprimé.

   ```
   SELECT * from system_schema.keyspaces ;
   ```

   Votre keyspace ne doit pas être répertorié. Notez qu'étant donné qu'il s'agit d'une opération asynchrone, il peut y avoir un délai avant que le keyspace ne soit supprimé.

Migrer vers Amazon Keyspaces

La migration vers Amazon Keyspaces (pour Apache Cassandra) présente de nombreux avantages convaincants pour les entreprises et les organisations. Voici quelques avantages clés qui font d'Amazon Keyspaces un choix intéressant pour la migration.

- Évolutivité — Amazon Keyspaces est conçu pour gérer des charges de travail massives et évoluer de manière fluide afin de s'adapter à l'augmentation des volumes de données et du trafic. Avec le Cassandra traditionnel, le dimensionnement n'est pas effectué à la demande et nécessite de planifier les pics futurs. Avec Amazon Keyspaces, vous pouvez facilement augmenter ou diminuer vos tableaux en fonction de la demande, afin que vos applications puissent gérer des pics de trafic soudains sans compromettre les performances.
- Performances — Amazon Keyspaces offre un accès aux données à faible latence, ce qui permet aux applications de récupérer et de traiter les données à une vitesse exceptionnelle. Son architecture distribuée garantit que les opérations de lecture et d'écriture sont réparties sur plusieurs nœuds, offrant des temps de réponse constants à un chiffre en millisecondes, même à des taux de requêtes élevés.
- Entièrement géré — Amazon Keyspaces est un service entièrement géré fourni par. AWS Cela signifie qu'il AWS gère les aspects opérationnels de la gestion des bases de données, notamment le provisionnement, la configuration, les correctifs, les sauvegardes et le dimensionnement. Cela vous permet de vous concentrer davantage sur le développement de vos applications et moins sur les tâches d'administration de base de données.
- Architecture sans serveur — Amazon Keyspaces fonctionne sans serveur. Vous ne payez que pour la capacité consommée, sans qu'aucun provisionnement de capacité initial ne soit requis. Vous n'avez pas de serveurs à gérer ni d'instances à choisir. Ce pay-per-request modèle offre une rentabilité et une charge opérationnelle minimale, car vous ne payez que pour les ressources que vous consommez sans avoir à provisionner et à surveiller les capacités.
- Flexibilité NoSQL avec schéma — Amazon Keyspaces suit un modèle de données NoSQL, offrant ainsi de la flexibilité dans la conception des schémas. Avec Amazon Keyspaces, vous pouvez stocker des données structurées, semi-structurées et non structurées, ce qui les rend parfaitement adaptées à la gestion de types de données divers et évolutifs. En outre, Amazon Keyspaces valide le schéma en écriture, ce qui permet une évolution centralisée du modèle de données. Cette flexibilité permet d'accélérer les cycles de développement et de s'adapter plus facilement à l'évolution des besoins de l'entreprise.

- Haute disponibilité et durabilité : Amazon Keyspaces réplique les données dans plusieurs zones de disponibilité au sein d'un même Région AWS, garantissant ainsi une haute disponibilité et une durabilité des données. Il gère automatiquement la réplication, le basculement et la restauration, minimisant ainsi le risque de perte de données ou d'interruption de service. Amazon Keyspaces fournit un SLA de disponibilité allant jusqu'à 99,999 %. Pour encore plus de résilience et des lectures locales à faible latence, Amazon Keyspaces propose une réplication multirégionale.
- Sécurité et conformité : Amazon Keyspaces s'intègre AWS Identity and Access Management pour un contrôle d'accès précis. Il fournit un chiffrement au repos et en transit, ce qui contribue à améliorer la sécurité de vos données. Amazon Keyspaces est également conforme aux normes de sécurité et aux lois sur la confidentialité, notamment les lois HIPAA, PCI DSS et RGPD, ce qui vous permet de répondre aux exigences réglementaires.
- Intégration à AWS l'écosystème — Dans le cadre de l' AWS écosystème, Amazon Keyspaces s'intègre parfaitement à d'autres Services AWS, par exemple AWS CloudFormation Amazon CloudWatch et. AWS CloudTrail Cette intégration vous permet de créer des architectures sans serveur, de tirer parti de l'infrastructure sous forme de code et de créer des applications pilotées par les données en temps réel.

Considérations générales relatives aux migrations vers Amazon Keyspaces

- Décomposez la migration en composants plus petits.

  Tenez compte des unités de migration suivantes et de leur empreinte potentielle en termes de taille des données brutes. La migration de petites quantités de données en une ou plusieurs phases peut contribuer à simplifier votre migration.

  - Par cluster : migrez toutes vos données Cassandra en une seule fois. Cette approche peut convenir aux petits clusters.
  - Par espace de touches ou par table : divisez votre migration en groupes d'espaces de touches ou de tables. Cette approche peut vous aider à migrer les données par étapes en fonction de vos besoins pour chaque charge de travail.
  - Par données — Envisagez de migrer les données pour un groupe spécifique d'utilisateurs ou de produits, afin de réduire encore davantage la taille des données.
- Priorisez les données à migrer en premier en fonction de la simplicité.

  Déterminez si vous avez des données qui pourraient d'abord être migrées plus facilement, par exemple des données qui ne changent pas à des moments précis, des données provenant de

traitements par lots effectués la nuit, des données non utilisées pendant les heures hors ligne ou des données provenant d'applications internes.

Rubriques

Conseils pour la migration des données depuis Apache Cassandra

Pour une migration réussie d'Apache Cassandra vers Amazon Keyspaces, nous vous recommandons de planifier et de comparer soigneusement les options disponibles. Cette rubrique décrit le fonctionnement du processus de migration, les outils disponibles et la manière dont vous pouvez évaluer les différentes stratégies de migration afin de sélectionner celle qui répond le mieux à vos besoins.

Rubriques

Compatibilité fonctionnelle

Examinez attentivement les différences fonctionnelles entre Apache Cassandra et Amazon Keyspaces avant la migration. Amazon Keyspaces prend en charge toutes les opérations courantes du plan de données Cassandra, telles que la création d'espaces de touches et de tables, la lecture et l'écriture de données. Cependant, certaines API Cassandra ne sont pas prises en charge par Amazon Keyspaces. Pour plus d'informations sur les API prises en charge, consultezthe section called “API, opérations, fonctions et types de données Cassandra pris en charge”. Pour un aperçu de toutes les différences fonctionnelles entre Amazon Keyspaces et Apache Cassandra, consultez. the section called “Différences fonctionnelles avec Apache Cassandra”

Pour comparer les API et le schéma Cassandra que vous utilisez aux fonctionnalités prises en charge dans Amazon Keyspaces, vous pouvez exécuter un script de compatibilité disponible dans le kit d'outils Amazon Keyspaces sur. GitHub

Comment utiliser le script de compatibilité

1. Téléchargez le script Python de compatibilité depuis GitHubet déplacez-le vers un emplacement ayant accès à votre cluster Apache Cassandra existant.
2. Le script de compatibilité utilise des paramètres similaires à`CQLSH`. `--port`Entrez l'adresse IP et le port que vous utilisez pour vous connecter et exécuter des requêtes sur l'un des nœuds Cassandra de votre cluster. `--host` Si votre cluster Cassandra utilise l'authentification, vous devez également fournir `-username` et`-password`. Pour exécuter le script de compatibilité, vous pouvez utiliser la commande suivante.

```
python toolkit-compat-tool.py --host hostname or IP -u "username" -p "password" --
port native transport port
```

Estimer le prix d'Amazon Keyspaces

Cette section fournit un aperçu des informations que vous devez collecter à partir de vos tables Apache Cassandra pour calculer le coût estimé d'Amazon Keyspaces. Chacune de vos tables nécessite des types de données différents, doit prendre en charge différentes requêtes CQL et gère un trafic de lecture/écriture distinct. La prise en compte de vos besoins sur la base de tableaux correspond aux modes d'isolation des ressources au niveau des tables d'Amazon Keyspaces et de capacité de débit de lecture/écriture. Avec Amazon Keyspaces, vous pouvez définir la capacité de lecture/écriture et les politiques de dimensionnement automatique pour les tables de manière indépendante. Comprendre les exigences relatives aux tables vous permet de hiérarchiser les tables à migrer en fonction des fonctionnalités, du coût et de l'effort de migration.

Collectez les métriques de la table Cassandra suivantes avant une migration. Ces informations permettent d'estimer le coût de votre charge de travail sur Amazon Keyspaces.

- Nom de la table : nom du keyspace complet et du nom de la table.
- Description : description de la table, par exemple de son utilisation ou du type de données qui y est stocké.
- Nombre moyen de lectures par seconde : nombre moyen de lectures au niveau des coordonnées par rapport au tableau sur un long intervalle de temps.
- Nombre moyen d'écritures par seconde : nombre moyen d'écritures au niveau des coordonnées par rapport à la table sur un long intervalle de temps.
- Taille de ligne moyenne en octets : taille de ligne moyenne en octets.

- Taille de stockage en Go : taille de stockage brute d'une table.
- Répartition de la cohérence des lectures : pourcentage de lectures utilisant une cohérence éventuelle (`LOCAL_ONE`ou`ONE`) par rapport à une cohérence forte (`LOCAL_QUORUM`).

Ce tableau présente un exemple des informations relatives à vos tables que vous devez rassembler lors de la planification d'une migration.

Nom de la table	Description	Nombre moyen de lectures par seconde	Nombre moyen d'écritures par seconde	Taille moyenne des lignes en octets	Taille de stockage en Go	Lire le tableau de cohérence
mykeyspace.mytable	Utilisé pour enregistrer l'historique du panier	10 000	5 000	2 200	2 000	100 % `LOCAL_ONE`
mykeyspace.mytable2	Utilisé pour stocker les dernières informations de profil	20 000	1 000	850	1 000	25 % `LOCAL_QUORUM` 75 % `LOCAL_ONE`

Comment collecter les métriques d'un tableau

Cette section fournit des instructions étape par étape sur la façon de collecter les métriques de table nécessaires à partir de votre cluster Cassandra existant. Ces mesures incluent la taille des lignes, la taille des tables et les demandes de lecture/écriture par seconde (RPS). Ils vous permettent d'évaluer les besoins en capacité de débit pour une table Amazon Keyspaces et d'estimer les prix.

Comment collecter les métriques d'une table sur la table source de Cassandra

1. Déterminer la taille des lignes

La taille des lignes est importante pour déterminer la capacité de lecture et l'utilisation de la capacité d'écriture dans Amazon Keyspaces. Le schéma suivant montre la distribution typique des données sur une plage de jetons Cassandra.

Vous pouvez utiliser un script d'échantillonnage de taille de ligne disponible sur GitHubpour collecter des métriques de taille de ligne pour chaque table de votre cluster Cassandra. Le script exporte les données de table depuis Apache Cassandra en utilisant `cqlsh` et `awk` pour calculer le minimum, le maximum, la moyenne et l'écart type de la taille des lignes sur un ensemble d'échantillons configurable de données de table. L'échantillonneur de taille de ligne transmet les arguments à`cqlsh`, de sorte que les mêmes paramètres peuvent être utilisés pour se connecter et lire à partir de votre cluster Cassandra.

La déclaration suivante en est un exemple.

```
./row-size-sampler.sh 10.22.33.44 9142 \\
   -u "username" -p "password" --ssl
```

Pour plus d'informations sur le calcul de la taille des lignes dans Amazon Keyspaces, consultez. the section called “Calcul de la taille des lignes”

2. Déterminer la taille de la table

Avec Amazon Keyspaces, vous n'avez pas besoin de provisionner le stockage à l'avance. Amazon Keyspaces surveille en permanence la taille facturable de vos tables afin de déterminer vos frais de stockage. Le stockage est facturé par Go par mois. La taille du tableau Amazon Keyspaces est basée sur la taille brute (non compressée) d'une seule réplique. Pour surveiller la taille du tableau dans Amazon Keyspaces, vous pouvez utiliser la

métrique`BillableTableSizeInBytes`, qui est affichée pour chaque tableau dans le. AWS Management Console

Pour estimer la taille facturable de votre tableau Amazon Keyspaces, vous pouvez utiliser l'une des deux méthodes suivantes :

- Utilisez la taille moyenne des lignes et multipliez-la par le nombre de lignes.

  Vous pouvez estimer la taille de la table Amazon Keyspaces en multipliant la taille moyenne des lignes par le nombre de lignes de votre table source Cassandra. Utilisez l'exemple de script de taille de ligne de la section précédente pour capturer la taille de ligne moyenne. Pour capturer le nombre de lignes, vous pouvez utiliser des outils tels que `dsbulk count` la détermination du nombre total de lignes dans votre table source.

- Utilisez le `nodetool` pour collecter les métadonnées des tables.

  `Nodetool`est un outil administratif fourni dans la distribution Apache Cassandra qui fournit un aperçu de l'état du processus Cassandra et renvoie les métadonnées des tables. Vous pouvez les utiliser `nodetool` pour échantillonner les métadonnées relatives à la taille des tables et ainsi extrapoler la taille des tables dans Amazon Keyspaces. La commande à utiliser est`nodetool tablestats`. Tablestats renvoie la taille et le taux de compression de la table. La taille du tableau est enregistrée comme celle `tablelivespace` du tableau et vous pouvez la diviser par le`compression ratio`. Multipliez ensuite cette valeur de taille par le nombre de nœuds. Enfin, divisez par le facteur de réplication (généralement trois). Il s'agit de la formule complète de calcul que vous pouvez utiliser pour évaluer la taille de la table.

  ```
  ((tablelivespace / compression ratio) * (total number of nodes))/ (replication
   factor)
  ```

  Supposons que votre cluster Cassandra comporte 12 nœuds. L'exécution de la `nodetool tablestats` commande renvoie une valeur `tablelivespace` de 200 Go et une valeur `compression ratio` de 0,5. Le keyspace possède un facteur de réplication de trois. Voici à quoi ressemble le calcul de cet exemple.

  ```
  (200 GB / 0.5) * (12 nodes)/ (replication factor of 3)
                         = 4,800 GB / 3
                         = 1,600 GB is the table size estimate for Amazon
   Keyspaces
  ```

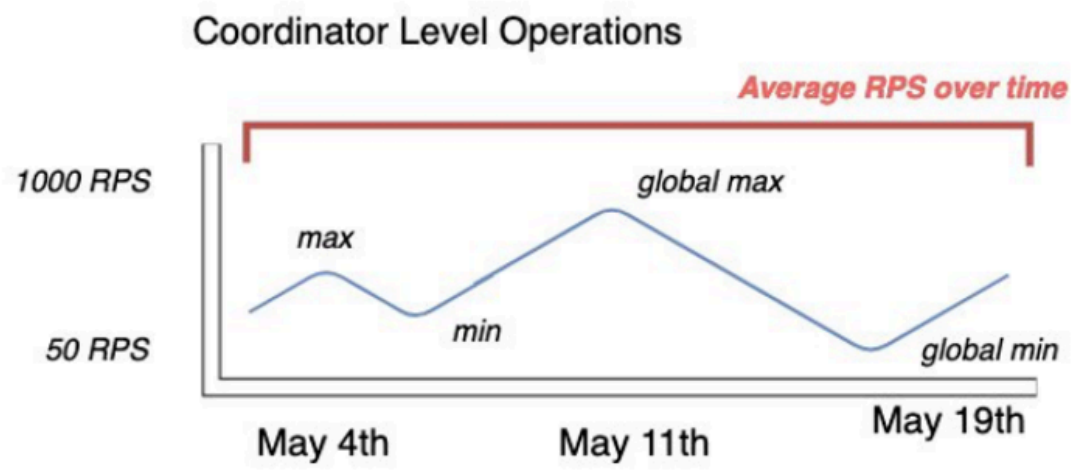
3. Capturez le nombre de lectures et d'écritures

   Pour déterminer les exigences de capacité et de mise à l'échelle de vos tables Amazon Keyspaces, capturez le taux de demandes de lecture et d'écriture de vos tables Cassandra avant la migration.

   Amazon Keyspaces fonctionne sans serveur et vous ne payez que pour ce que vous utilisez. En général, le prix du débit de lecture/écriture dans Amazon Keyspaces est basé sur le nombre et la taille des demandes. Amazon Keyspaces propose deux modes de capacité : le mode à la demande et le mode capacité provisionnée. Le mode capacité à la demande est une option de facturation flexible capable de répondre à des milliers de demandes par seconde sans qu'il soit nécessaire de planifier la capacité. Cette option propose une pay-per-request tarification pour les demandes de lecture et d'écriture afin que vous ne payiez que pour ce que vous utilisez. Si vous choisissez le mode de capacité de débit alloué vous spécifiez le nombre de lectures et d'écritures par seconde requis pour votre application. Cela vous permet de gérer votre utilisation d'Amazon Keyspaces afin de rester au niveau ou en dessous d'un taux de demandes défini afin d'optimiser le prix et de garantir la prévisibilité. Le mode provisionné offre une mise à l'échelle automatique pour ajuster automatiquement votre taux provisionné à la hausse ou à la baisse afin d'améliorer l'efficacité opérationnelle. Pour plus d'informations sur la gestion des ressources sans serveur, consultezGestion des ressources sans serveur.

   Étant donné que vous allouez séparément la capacité de débit de lecture et d'écriture dans Amazon Keyspaces, vous devez mesurer le taux de demandes de lecture et d'écriture dans vos tables existantes de manière indépendante.

   Pour recueillir les indicateurs d'utilisation les plus précis de votre cluster Cassandra existant, capturez le nombre moyen de demandes par seconde (RPS) pour les opérations de lecture et d'écriture au niveau du coordinateur sur une période prolongée pour une table agrégée sur tous les nœuds d'un seul centre de données. La capture du RPS moyen sur une période d'au moins plusieurs semaines permet de saisir les pics et les creux de vos modèles de trafic, comme le montre le schéma suivant.



Deux options s'offrent à vous pour déterminer le taux de requêtes en lecture et en écriture de votre table Cassandra.

- Utiliser le système de surveillance Cassandra existant

  Vous pouvez utiliser les métriques présentées dans le tableau suivant pour observer les demandes de lecture et d'écriture. Notez que les noms des métriques peuvent changer en fonction de l'outil de surveillance que vous utilisez.

Dimension	Métrique Cassandra JMX
écrit	`org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count`
lit	`org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count`

- Utilisation de la `nodetool`

  Utilisez `nodetool tablestats` et `nodetool info` pour capturer les opérations de lecture et d'écriture moyennes à partir de la table. `tablestats`renvoie le nombre total de lectures et d'écritures depuis le moment où le nœud a été initié. `nodetool info`fournit le temps de disponibilité d'un nœud en secondes. Pour obtenir la moyenne des lectures

et des écritures par seconde, divisez le nombre de lectures et d'écritures par le temps de disponibilité du nœud en secondes. Ensuite, pour les lectures, vous divisez par le niveau de cohérence et pour les écritures, vous divisez par le facteur de réplication. Ces calculs sont exprimés dans les formules suivantes.

Formule pour le nombre moyen de lectures par seconde :

```
((number of reads * number of nodes in cluster) / read consistency quorum
 (2)) / uptime
```

Formule pour le nombre moyen d'écritures par seconde :

```
((number of writes * number of nodes in cluster) / replication factor of 3) /
 uptime
```

Supposons que nous ayons un cluster de 12 nœuds actif depuis 4 semaines. `nodetool info`renvoie 2 419 200 secondes de disponibilité et `nodetool tablestats` renvoie 1 milliard d'écritures et 2 milliards de lectures. Cet exemple entraînerait le calcul suivant.

```
((2 billion reads * 12 in cluster) / read consistency quorum (2)) / 2,419,200
 seconds
=  12 billion reads / 2,419,200 seconds
=  4,960 read request per second
                       ((1 billion writes * 12 in cluster) / replication
 factor of 3) / 2,419,200 seconds
=  4 billion writes / 2,419,200 seconds
=  1,653 write request per second
```

4. Déterminer le taux d'utilisation de la capacité de la table

   Pour estimer l'utilisation moyenne de la capacité, commencez par les taux de demandes moyens et la taille moyenne des lignes de votre table source Cassandra.

   Amazon Keyspaces utilise des unités de capacité de lecture (RCU) et des unités de capacité d'écriture (WCU) pour mesurer la capacité de débit allouée pour les lectures et les écritures pour les tables. Pour cette estimation, nous utilisons ces unités pour calculer les besoins en capacité de lecture et d'écriture de la nouvelle table Amazon Keyspaces après la migration. Plus loin dans cette rubrique, nous verrons comment le choix entre le mode de capacité provisionnée et le mode de capacité à la demande affecte la facturation. Mais pour l'estimation de l'utilisation des capacités, nous supposons que la table est en mode provisionné.

Une RCU représente une `LOCAL_QUORUM` ou deux demandes de `LOCAL_ONE` lecture pour une ligne d'une taille maximale de 4 Ko. Si vous devez lire une ligne de plus de 4 Ko, l'opération de lecture utilise des RCU supplémentaires. Le nombre total de RCU requis dépend de la taille de la ligne et de la cohérence que vous souhaitez utiliser `LOCAL_QUORUM` ou `LOCAL_ONE` lire. Par exemple, la lecture d'une ligne de 8 Ko nécessite 2 RCU utilisant la cohérence de `LOCAL_QUORUM` lecture, et 1 RCU si vous choisissez la cohérence de `LOCAL_ONE` lecture.

Une WCU représente une écriture pour une ligne d'une taille maximale de 1 Ko. Toutes les écritures utilisent la cohérence `LOCAL_QUORUM` et il n'y a pas de frais supplémentaires pour l'utilisation de transactions légères (LWT). Si vous devez écrire une ligne supérieure à 1 Ko, l'opération d'écriture utilise des WCU supplémentaires. Le nombre total de WCU requis dépend de la taille de la ligne. Par exemple, si la taille de votre ligne est de 2 Ko, vous avez besoin de 2 WCU pour exécuter une demande d'écriture.

La formule suivante peut être utilisée pour estimer les RCU et WCU nécessaires. La capacité de lecture dans les RCU peut être déterminée en multipliant les lectures par seconde par le nombre de lignes lues par lecture multiplié par la taille moyenne des lignes divisée par 4 Ko et arrondie au nombre entier le plus proche.

La capacité d'écriture dans les WCU peut être déterminée en multipliant le nombre de demandes par la taille moyenne des lignes divisée par 1 Ko et arrondie au nombre entier le plus proche. Cela s'exprime dans les formules suivantes.

```
Read requests per second * ROUNDUP((Average Row Size)/4096 per unit) =  RCUs per
 second

Write requests per second * ROUNDUP(Average Row Size/1024 per unit) = WCUs per
 second
```

Par exemple, si vous effectuez 4 960 demandes de lecture avec une taille de ligne de 2,5 Ko sur votre table Cassandra, vous avez besoin de 4 960 RCU dans Amazon Keyspaces. Si vous effectuez actuellement 1 653 demandes d'écriture par seconde avec une taille de ligne de 2,5 Ko sur votre table Cassandra, vous avez besoin de 4 959 WCU par seconde dans Amazon Keyspaces. Cet exemple est exprimé dans les formules suivantes.

```
4,960 read requests per second * ROUNDUP( 2.5KB /4KB bytes per unit)
= 4,960 read requests per second * 1 RCU
= 4,960 RCUs
```

```
1,653 write requests per second * ROUNDUP(2.5KB/1KB per unit)
= 1,653 requests per second * 3 WCUs
= 4,959 WCUs
```

L'utilisation vous `eventual consistency` permet d'économiser jusqu'à la moitié de la capacité de débit à chaque demande de lecture. Chaque lecture finalement cohérente peut consommer jusqu'à 8 Ko. Vous pouvez calculer les lectures cohérentes éventuelles en multipliant le calcul précédent par 0,5, comme indiqué dans la formule suivante.

```
4,960 read requests per second * ROUNDUP( 2.5KB /4KB per unit) * .5
= 2,480 read request per second * 1 RCU
= 2,480 RCUs
```

5. Calculez l'estimation du prix mensuel pour Amazon Keyspaces

   Pour estimer la facturation mensuelle de la table en fonction du débit de capacité de lecture/écriture, vous pouvez calculer le prix pour le mode à la demande et pour le mode provisionné à l'aide de différentes formules et comparer les options pour votre table.

   Mode provisionné — La consommation de capacité de lecture et d'écriture est facturée selon un taux horaire basé sur les unités de capacité par seconde. Divisez d'abord ce taux par 0,7 pour représenter l'utilisation cible de 70 % par défaut de l'autoscaling. Multipliez ensuite par 30 jours calendaires, 24 heures par jour, et par le tarif régional. Ce calcul est résumé dans les formules suivantes.

   ```
   (read capacity per second / .7) * 24 hours * 30 days * regional rate
                   (write capacity per second / .7) * 24 hours * 30 days * regional
    rate
   ```

   Mode à la demande — La capacité de lecture et d'écriture est facturée au tarif par demande. Tout d'abord, multipliez le taux de demandes par 30 jours civils et 24 heures par jour. Divisez ensuite par un million d'unités de demande. Enfin, multipliez par le taux régional. Ce calcul est résumé dans les formules suivantes.

   ```
   ((read capacity per second * 30 * 24 * 60 * 60) / 1 Million read request units) *
    regional rate
                   ((write capacity per second * 30 * 24 * 60 * 60) / 1 Million write
    request units) * regional rate
   ```

Choisissez une stratégie de migration

En général, vous pouvez choisir entre trois stratégies de migration différentes lorsque vous migrez d'Apache Cassandra vers Amazon Keyspaces :

- Hors ligne — Cette migration implique la copie d'un ensemble de données de Cassandra vers Amazon Keyspaces dans le cadre d'un déploiement de migration d'applications de style bleu/vert. Si votre application peut tolérer certains temps d'arrêt pendant la migration, cette option peut simplifier le processus de migration. Pour plus d'informations sur la migration hors ligne, voir

  the section called “Migration hors ligne”.
- En ligne — Il s'agit d'un déploiement de style Canary qui inclut généralement des écritures doubles écrites directement dans la logique de l'application. Les applications qui ne nécessitent aucun temps d'arrêt pendant la migration nécessitent que les données soient copiées pendant que les lectures et les écritures en direct passent d'une source de données à l'autre.
- Hybride : cette approche permet de répliquer les modifications en temps quasi réel, mais l'application est chargée de passer des lectures aux écritures.

Après avoir examiné plus en détail les stratégies de migration disponibles, vous pouvez placer les options dans un arbre de décision afin de simplifier le processus en fonction de vos besoins et des ressources disponibles.

Migration hors ligne vers Amazon Keyspaces

Les migrations hors ligne conviennent lorsque vous pouvez vous permettre une interruption de service pour effectuer la migration. Il est courant dans les entreprises de disposer de fenêtres de maintenance pour les correctifs, les versions importantes ou de temps d'arrêt pour les mises à niveau matérielles ou les mises à niveau majeures. La migration hors ligne peut utiliser cette fenêtre pour copier des données et transférer le trafic de l'application d'Apache Cassandra vers Amazon Keyspaces. La migration hors ligne réduit les modifications apportées à l'application car elle ne nécessite pas de communication simultanée avec Cassandra et Amazon Keyspaces. De plus, lorsque le flux de données est suspendu, l'état exact peut être copié sans conserver les mutations.

Dans cet exemple, vous utilisez Amazon Simple Storage Service (Amazon S3) comme zone intermédiaire pour les données lors de la migration hors ligne afin de minimiser les temps d'arrêt. Vous pouvez importer automatiquement les données que vous avez stockées au format Parquet dans Amazon S3 dans une table Amazon Keyspaces à l'aide du connecteur Spark Cassandra et.

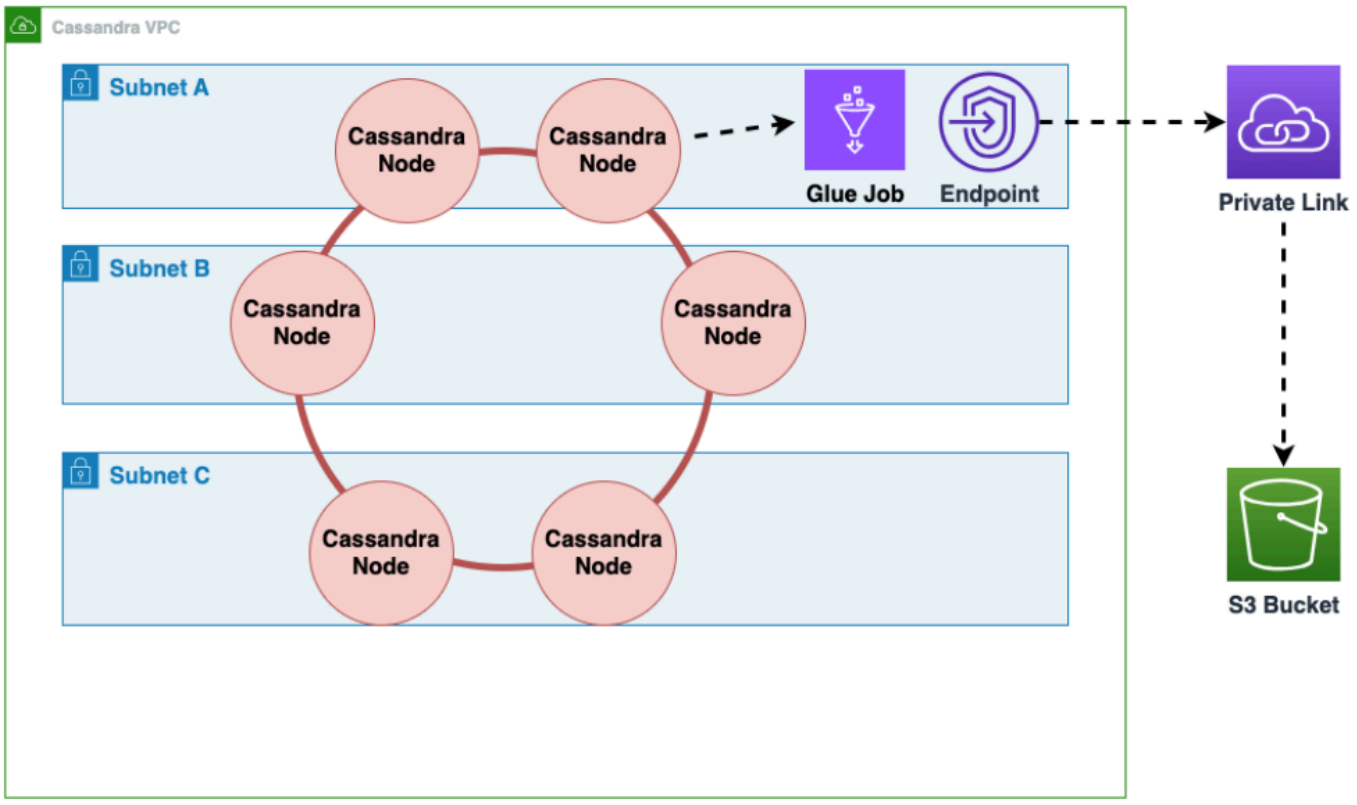
AWS Glue La section suivante va présenter un aperçu général du processus. Vous pouvez trouver des exemples de code pour ce processus sur Github.

Le processus de migration hors ligne d'Apache Cassandra vers Amazon Keyspaces à l'aide d'Amazon S3 nécessite AWS Glue les AWS Glue tâches suivantes.

1. Une tâche ETL qui extrait et transforme les données CQL et les stocke dans un compartiment Amazon S3.
2. Une deuxième tâche qui importe les données du bucket vers Amazon Keyspaces.
3. Une troisième tâche pour importer des données incrémentielles.

Comment effectuer une migration hors ligne vers Amazon Keyspaces depuis Cassandra exécutée sur Amazon EC2 dans un Amazon Virtual Private Cloud

1. Vous devez d'abord AWS Glue exporter les données de table de Cassandra au format Parquet et les enregistrer dans un compartiment Amazon S3. Vous devez exécuter une AWS Glue tâche à l'aide d'un AWS Glue connecteur vers un VPC où réside l'instance Amazon EC2 exécutant Cassandra. Ensuite, à l'aide du point de terminaison privé Amazon S3, vous pouvez enregistrer des données dans le compartiment Amazon S3. Le schéma suivant illustre ces étapes.



2. Répartissez les données dans le compartiment Amazon S3 pour améliorer la randomisation des données. Les données importées de manière uniforme permettent de répartir davantage le trafic dans la table cible. Cette étape est requise lors de l'exportation de données depuis Cassandra avec de grandes partitions (partitions de plus de 1 000 lignes) afin d'éviter les raccourcis clavier lors de l'insertion des données dans Amazon Keyspaces. Les principaux problèmes affectent `WriteThrottleEvents` Amazon Keyspaces et augmentent le temps de chargement.

3. Utilisez une autre AWS Glue tâche pour importer des données depuis le compartiment Amazon S3 vers Amazon Keyspaces. Les données mélangées dans le compartiment Amazon S3 sont stockées au format Parquet.

Outils de migration de données vers Amazon Keyspaces

Différents outils sont disponibles pour migrer les données vers Amazon Keyspaces

- Outils de migration
  - Pour les migrations de grande envergure, pensez à utiliser un outil d'extraction, de transformation et de chargement (ETL). Vous pouvez l'utiliser AWS Glue pour effectuer rapidement et efficacement des migrations de transformation de données.
  - Pour savoir comment utiliser le connecteur Apache Cassandra Spark pour écrire des données sur Amazon Keyspaces, consultez. Intégration à Apache Spark
  - Commencez rapidement à charger des données dans Amazon Keyspaces à l'aide de la `COPY FROM` commande cqlsh. cqlsh est inclus dans Apache Cassandra et convient parfaitement au

chargement de petits ensembles de données ou de données de test. Pour step-by-step obtenir des instructions, voirthe section called “Chargement de données à l'aide de cqlsh”.

- Vous pouvez également utiliser le DataStax Bulk Loader pour Apache Cassandra pour charger des données dans Amazon Keyspaces à l'aide `dsbulk` de la commande. DSBulk fournit des fonctionnalités d'importation plus robustes que cqlsh et est disponible depuis le référentiel. GitHub Pour step-by-step obtenir des instructions, voirthe section called “Chargement de données à l'aide de DSBulk”.

Rubriques

- Tutoriel : Chargement de données dans Amazon Keyspaces à l'aide de cqlsh
- Tutoriel : Chargement de données dans Amazon Keyspaces à l'aide de DSBulk

Tutoriel : Chargement de données dans Amazon Keyspaces à l'aide de cqlsh

Ce step-by-step didacticiel vous explique comment migrer des données d'Apache Cassandra vers Amazon Keyspaces à l'aide de la commande. `cqlsh COPY` Dans ce didacticiel, vous effectuez les opérations suivantes :

Rubriques

- Prérequis
- Étape 1 : Création du fichier CSV source et de la table cible
- Étape 2 : Préparation des données
- Étape 3 : définir la capacité de débit de la table
- Étape 4 : Configuration des cqlsh COPY FROM paramètres
- Étape 5 : Exécuter la cqlsh COPY FROM commande
- Résolution des problèmes

Prérequis

Vous devez effectuer les tâches suivantes avant de pouvoir commencer ce didacticiel.

1. Si ce n'est pas déjà fait, inscrivez-vous Compte AWS en suivant les étapes indiquées surthe section called “Con AWS Identity and Access Management figuration”.

2. Créez des informations d'identification spécifiques au service en suivant les étapes décrites dans. the section called “Générez des informations d'identification spécifiques au service à l'aide de la console”
3. Configurez la connexion au shell Cassandra Query Language (cqlsh) et confirmez que vous pouvez vous connecter à Amazon Keyspaces en suivant les étapes indiquées sur. the section called “Utiliser `cqlsh`”

Étape 1 : Création du fichier CSV source et de la table cible

Pour ce didacticiel, nous utilisons un fichier de valeurs séparées par des virgules (CSV) dont le `keyspaces_sample_table.csv` nom est le fichier source pour la migration des données. Le fichier d'exemple fourni contient quelques lignes de données pour une table portant le nom`book_awards`.

1. Créez le fichier source. Vous pouvez choisir l’une des options suivantes :
   - Téléchargez l'exemple de fichier CSV (`keyspaces_sample_table.csv`) contenu dans le fichier d'archive samplemigration.zip suivant. Décompressez l'archive et notez le chemin vers`keyspaces_sample_table.csv`.
   - Pour remplir un fichier CSV avec vos propres données stockées dans une base de données Apache Cassandra, vous pouvez remplir le fichier CSV source à l'aide de l'`cqlsh` `COPY TO`instruction illustrée dans l'exemple suivant.

     ```
     cqlsh localhost 9042 -u "username" -p "password" --execute
      "COPY mykeyspace.mytable TO 'keyspaces_sample_table.csv' WITH HEADER=true"
     ```

     Assurez-vous que le fichier CSV que vous créez répond aux exigences suivantes :

     - La première ligne contient les noms des colonnes.
     - Les noms des colonnes du fichier CSV source correspondent aux noms des colonnes de la table cible.
     - Les données sont délimitées par une virgule.
     - Toutes les valeurs de données sont des types de données Amazon Keyspaces valides. veuillez consulter the section called “Types de données”.
2. Créez le keyspace et le tableau cibles dans Amazon Keyspaces.

a. Connectez-vous à Amazon Keyspaces en utilisant`cqlsh`, en remplaçant le point de terminaison du service, le nom d'utilisateur et le mot de passe dans l'exemple suivant par vos propres valeurs.

```
cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Créez un nouvel espace de touches avec le `catalog` nom indiqué dans l'exemple suivant.

```
CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

c. Lorsque le nouvel espace de touches est disponible, utilisez le code suivant pour créer la table `book_awards` cible.

```
CREATE TABLE "catalog.book_awards" (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank)
   );
```

Si Apache Cassandra est votre source de données d'origine, un moyen simple de créer la table cible Amazon Keyspaces avec les en-têtes correspondants consiste à générer `CREATE TABLE` l'instruction à partir de la table source, comme indiqué dans l'instruction suivante.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Créez ensuite la table cible dans Amazon Keyspaces avec les noms des colonnes et les types de données correspondant à la description de la table source de Cassandra.

Étape 2 : Préparation des données

La préparation des données sources pour un transfert efficace est un processus en deux étapes. Tout d'abord, vous répartissez les données de manière aléatoire. Dans la deuxième étape, vous analysez les données pour déterminer les valeurs de `cqlsh` paramètres appropriées et les paramètres de table requis.

Randomiser les données

La `cqlsh COPY FROM` commande lit et écrit les données dans l'ordre dans lequel elles apparaissent dans le fichier CSV. Si vous utilisez la `cqlsh COPY TO` commande pour créer le fichier source, les données sont écrites dans un ordre trié par clé dans le fichier CSV. En interne, Amazon Keyspaces partitionne les données à l'aide de clés de partition. Bien qu'Amazon Keyspaces intègre une logique permettant d'équilibrer la charge des demandes pour la même clé de partition, le chargement des données est plus rapide et plus efficace si vous répartissez la commande de manière aléatoire. En effet, vous pouvez tirer parti de l'équilibrage de charge intégré qui se produit lorsque Amazon Keyspaces écrit sur différentes partitions.

Pour répartir uniformément les écritures sur les partitions, vous devez répartir les données de manière aléatoire dans le fichier source. Vous pouvez écrire une application pour cela ou utiliser un outil open source tel que Shuf. Shuf est disponible gratuitement sur les distributions Linux, sur macOS (en installant coreutils dans homebrew) et sur Windows (en utilisant le sous-système Windows pour Linux (WSL)). Une étape supplémentaire est nécessaire pour éviter que la ligne d'en-tête contenant les noms des colonnes ne soit modifiée au cours de cette étape.

Pour randomiser le fichier source tout en préservant l'en-tête, entrez le code suivant.

```
tail -n +2 keyspaces_sample_table.csv | shuf -o keyspace.table.csv && (head
 -1 keyspaces_sample_table.csv && cat keyspace.table.csv ) > keyspace.table.csv1 &&
 mv keyspace.table.csv1 keyspace.table.csv
```

Shuf réécrit les données dans un nouveau fichier CSV appelé. `keyspace.table.csv` Vous pouvez désormais supprimer le `keyspaces_sample_table.csv` fichier, vous n'en avez plus besoin.

Analyser les données

Déterminez la taille moyenne et maximale des lignes en analysant les données.

Vous le faites pour les raisons suivantes :

- La taille moyenne des lignes permet d'estimer la quantité totale de données à transférer.

- Vous avez besoin de la taille de ligne moyenne pour fournir la capacité d'écriture nécessaire au téléchargement des données.
- Vous pouvez vous assurer que la taille de chaque ligne est inférieure à 1 Mo, ce qui correspond à la taille de ligne maximale dans Amazon Keyspaces.

> **Note**
>
> Ce quota fait référence à la taille des lignes et non à la taille de la partition. Contrairement aux partitions Apache Cassandra, les partitions Amazon Keyspaces peuvent être de taille pratiquement indépendante. Les clés de partition et les colonnes de clustering nécessitent un espace de stockage supplémentaire pour les métadonnées, que vous devez ajouter à la taille brute des lignes. Pour de plus amples informations, veuillez consulter the section called “Calcul de la taille des lignes”.

Le code suivant utilise AWK pour analyser un fichier CSV et imprimer la taille de ligne moyenne et maximale.

```
awk -F, 'BEGIN {samp=10000;max=-1;}{if(NR>1){len=length($0);t+=len;avg=t/
NR;max=(len>max ? len : max)}}NR==samp{exit}END{printf("{lines: %d, average: %d bytes,
 max: %d bytes}\n",NR,avg,max);}' keyspace.table.csv
```

L'exécution de ce code génère le résultat suivant.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Vous utiliserez la taille de ligne moyenne à l'étape suivante de ce didacticiel pour configurer la capacité d'écriture de la table.

Étape 3 : définir la capacité de débit de la table

Ce didacticiel explique comment régler cqlsh pour charger des données dans un intervalle de temps défini. Comme vous savez à l'avance combien de lectures et d'écritures vous devez effectuer, utilisez le mode capacité provisionnée. Une fois le transfert de données terminé, vous devez définir le mode de capacité de la table en fonction des modèles de trafic de votre application. Pour en savoir plus sur la gestion des capacités, voirGestion des ressources sans serveur.

Avec le mode capacité provisionnée, vous spécifiez à l'avance la capacité de lecture et d'écriture que vous souhaitez allouer à votre table. La capacité d'écriture est facturée à l'heure et mesurée en unités de capacité d'écriture (WCU). Chaque WCU possède une capacité d'écriture suffisante pour prendre en charge l'écriture de 1 Ko de données par seconde. Lorsque vous chargez les données, le taux d'écriture doit être inférieur au nombre maximum de WCU (paramètre :`write_capacity_units`) défini sur la table cible.

Par défaut, vous pouvez allouer jusqu'à 40 000 WCU à une table et 80 000 WCU pour toutes les tables de votre compte. Si vous avez besoin de capacités supplémentaires, vous pouvez demander une augmentation de quota dans la console Service Quotas. Pour de plus amples informations sur les quotas, veuillez consulter Quotas.

Calculez le nombre moyen de WCU nécessaires pour un insert

L'insertion de 1 Ko de données par seconde nécessite 1 WCU. Si votre fichier CSV comporte 360 000 lignes et que vous souhaitez charger toutes les données en 1 heure, vous devez écrire 100 lignes par seconde (360 000 lignes/60 minutes/60 secondes = 100 lignes par seconde). Si chaque ligne contient jusqu'à 1 Ko de données, pour insérer 100 lignes par seconde, vous devez allouer 100 WCU à votre table. Si chaque ligne contient 1,5 Ko de données, vous avez besoin de deux WCU pour insérer une ligne par seconde. Par conséquent, pour insérer 100 lignes par seconde, vous devez provisionner 200 WCU.

Pour déterminer le nombre de WCU dont vous avez besoin pour insérer une ligne par seconde, divisez la taille moyenne des lignes en octets par 1024 et arrondissez au nombre entier le plus proche.

Par exemple, si la taille moyenne des lignes est de 3 000 octets, vous avez besoin de trois WCU pour insérer une ligne par seconde.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Calculer le temps et la capacité de chargement des données

Maintenant que vous connaissez la taille moyenne et le nombre de lignes de votre fichier CSV, vous pouvez calculer le nombre de WCU nécessaires pour charger les données dans un laps de temps donné, ainsi que le temps approximatif nécessaire pour charger toutes les données de votre fichier CSV en utilisant différents paramètres WCU.

Par exemple, si chaque ligne de votre fichier fait 1 Ko et que votre fichier CSV contient 1 000 000 lignes, pour charger les données en 1 heure, vous devez allouer au moins 278 WCU à votre table pendant cette heure.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Configuration des paramètres de capacité provisionnée

Vous pouvez définir les paramètres de capacité d'écriture d'une table lorsque vous créez la table ou à l'aide de la commande `ALTER TABLE` CQL. Voici la syntaxe permettant de modifier les paramètres de capacité provisionnée d'une table à l'aide de l'instruction `ALTER TABLE` CQL.

```
ALTER TABLE mykeyspace.mytable WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100,
 'write_capacity_units': 278}} ;
```

Pour la référence linguistique complète, voirthe section called “ALTER TABLE”.

Étape 4 : Configuration des **`cqlsh COPY FROM`** paramètres

Cette section explique comment déterminer les valeurs des paramètres pour`cqlsh COPY FROM`. La `cqlsh COPY FROM` commande lit le fichier CSV que vous avez préparé précédemment et insère les données dans Amazon Keyspaces à l'aide de CQL. La commande divise les lignes et répartit les `INSERT` opérations entre un ensemble de travailleurs. Chaque collaborateur établit une connexion avec Amazon Keyspaces et envoie des `INSERT` demandes via ce canal.

La `cqlsh COPY` commande n'a pas de logique interne permettant de répartir le travail de manière égale entre ses employés. Cependant, vous pouvez le configurer manuellement pour vous assurer que le travail est réparti de manière uniforme. Commencez par passer en revue les principaux paramètres cqlsh suivants :

- DÉLIMITEUR — Si vous avez utilisé un délimiteur autre qu'une virgule, vous pouvez définir ce paramètre, dont la valeur par défaut est une virgule.
- INGESTRATE — Nombre cible de lignes à `cqlsh COPY FROM` traiter par seconde. S'il n'est pas défini, la valeur par défaut est de 100 000.
- NUMPROCESSES — Le nombre de processus enfants créés par cqlsh pour les tâches. `COPY FROM` Le maximum pour ce paramètre est de 16, la valeur par défaut `num_cores - 1` `num_cores` étant le nombre de cœurs de traitement sur l'hôte exécutant cqlsh.

- MAXBATCHSIZE — La taille du lot détermine le nombre maximal de lignes insérées dans la table de destination en un seul lot. S'il n'est pas défini, cqlsh utilise des lots de 20 lignes insérées.
- CHUNKSIZE — Taille de l'unité de travail transmise à l'enfant travailleur. Par défaut, il est défini sur 5 000.
- MAXATTEMPTS — Le nombre maximum de fois où il est possible de réessayer un worker chunk ayant échoué. Une fois le nombre maximal de tentatives atteint, les enregistrements ayant échoué sont écrits dans un nouveau fichier CSV que vous pourrez réexécuter ultérieurement après avoir examiné l'échec.

`INGESTRATE`Défini en fonction du nombre de WCU que vous avez provisionnés dans la table de destination cible. La `INGESTRATE` `cqlsh COPY FROM` commande n'est pas une limite, c'est une moyenne cible. Cela signifie qu'il peut (et c'est souvent le cas) dépasser le nombre que vous avez défini. Pour tenir compte des rafales et vous assurer que la capacité est suffisante pour traiter les demandes de chargement de données, définissez 90 % `INGESTRATE` de la capacité d'écriture de la table.

```
INGESTRATE = WCUs * .90
```

Définissez ensuite le `NUMPROCESSES` paramètre sur une valeur inférieure d'un au nombre de cœurs de votre système. Pour connaître le nombre de cœurs de votre système, vous pouvez exécuter le code suivant.

```
python -c "import multiprocessing; print(multiprocessing.cpu_count())"
```

Pour ce didacticiel, nous utilisons la valeur suivante.

```
NUMPROCESSES = 4
```

Chaque processus crée un travailleur, qui établit une connexion avec Amazon Keyspaces. Amazon Keyspaces peut prendre en charge jusqu'à 3 000 demandes CQL par seconde à chaque connexion. Cela signifie que vous devez vous assurer que chaque travailleur traite moins de 3 000 demandes par seconde.

Comme c'est le cas`INGESTRATE`, les travailleurs dépassent souvent le nombre que vous avez défini et ne sont pas limités par les secondes d'horloge. Par conséquent, pour tenir compte des rafales, définissez vos paramètres cqlsh de manière à ce que chaque travailleur traite 2 500 demandes

par seconde. Pour calculer la quantité de travail distribuée à un travailleur, suivez les directives suivantes.

- Divisez `INGESTRATE` par`NUMPROCESSES`.
- Si`INGESTRATE/NUMPROCESSES`> 2 500, abaissez le `INGESTRATE` pour que cette formule soit vraie.

```
INGESTRATE / NUMPROCESSES <= 2,500
```

Avant de configurer les paramètres pour optimiser le téléchargement de nos exemples de données, examinons les paramètres `cqlsh` par défaut et voyons comment leur utilisation influe sur le processus de téléchargement des données. Dans la `cqlsh COPY FROM` mesure où il est utilisé `CHUNKSIZE` pour créer des parties du travail (`INSERT`déclarations) à distribuer aux travailleurs, le travail n'est pas automatiquement distribué de manière uniforme. Certains travailleurs peuvent rester inactifs, selon le `INGESTRATE` réglage.

Pour répartir le travail de manière égale entre les travailleurs et maintenir chaque travailleur au taux optimal de 2 500 demandes par seconde, vous devez définir `CHUNKSIZE`MAXBATCHSIZE, et `INGESTRATE` en modifiant les paramètres d'entrée. Pour optimiser l'utilisation du trafic réseau pendant le chargement des données, choisissez une valeur proche de la valeur maximale de 30. `MAXBATCHSIZE` En passant `CHUNKSIZE` à 100 et `MAXBATCHSIZE` à 25, les 10 000 rangées sont réparties uniformément entre les quatre travailleurs (10 000/ 2500 = 4).

L'exemple de code suivant illustre cela.

```
INGESTRATE = 10,000
NUMPROCESSES = 4
CHUNKSIZE = 100
MAXBATCHSIZE. = 25
Work Distribution:
Connection 1 / Worker 1 : 2,500 Requests per second
Connection 2 / Worker 2 : 2,500 Requests per second
Connection 3 / Worker 3 : 2,500 Requests per second
Connection 4 / Worker 4 : 2,500 Requests per second
```

En résumé, utilisez les formules suivantes pour définir `cqlsh COPY FROM` les paramètres :

- INGESTRATE = write_capacity_units * .90
- NUMPROCESSES = num_cores -1 (par défaut)

- INGESTRATE/NUMPROCESSES = 2 500 (Cette déclaration doit être vraie.)
- MAXBATCHSIZE = 30 (20 par défaut). Amazon Keyspaces accepte des lots allant jusqu'à 30.)
- CHUNKSIZE = (INGESTRATE/ NUMPROCESSES)/MAXBATCHSIZE

Maintenant que vous avez calculé `NUMPROCESSESINGESTRATE`, et`CHUNKSIZE`, vous êtes prêt à charger vos données.

Étape 5 : Exécuter la **`cqlsh COPY FROM`** commande

Pour exécuter la `cqlsh COPY FROM` commande, procédez comme suit.

1. Connectez-vous à Amazon Keyspaces à l'aide de cqlsh.
2. Choisissez votre keyspace à l'aide du code suivant.

```
USE catalog;
```

3. Définissez la cohérence d'écriture sur`LOCAL_QUORUM`. Pour garantir la durabilité des données, Amazon Keyspaces n'autorise aucun autre paramètre de cohérence d'écriture. Consultez le code suivant.

```
CONSISTENCY LOCAL_QUORUM;
```

4. Préparez votre `cqlsh COPY FROM` syntaxe à l'aide de l'exemple de code suivant.

```
COPY book_awards FROM './keyspace.table.csv' WITH HEADER=true
AND INGESTRATE=calculated ingestrate
AND NUMPROCESSES=calculated numprocess
AND MAXBATCHSIZE=20
AND CHUNKSIZE=calculated chunksize;
```

5. Exécutez l'instruction préparée à l'étape précédente. cqlsh renvoie tous les paramètres que vous avez configurés.

   a. Assurez-vous que les paramètres correspondent à ce que vous avez saisi. Consultez l'exemple suivant.

```
Reading options from the command line: {'chunksize': '120', 'header': 'true',
 'ingestrate': '36000', 'numprocesses': '15', 'maxbatchsize': '20'}
Using 15 child processes
```

b. Vérifiez le nombre de lignes transférées et le taux moyen actuel, comme indiqué dans l'exemple suivant.

```
Processed: 57834 rows; Rate: 6561 rows/s; Avg. rate: 31751 rows/s
```

c. Lorsque cqlsh a fini de télécharger les données, consultez le résumé des statistiques de chargement des données (nombre de fichiers lus, temps d'exécution et lignes ignorées) comme indiqué dans l'exemple suivant.

```
15556824 rows imported from 1 files in 8 minutes and 8.321 seconds (0 skipped).
```

Dans cette dernière étape du didacticiel, vous avez chargé les données sur Amazon Keyspaces.

> **Important**
>
> Maintenant que vous avez transféré vos données, ajustez les paramètres du mode capacité de votre table cible pour qu'ils correspondent aux modèles de trafic habituels de votre application. Vous êtes facturé au taux horaire pour votre capacité provisionnée jusqu'à ce que vous la modifiiez.

Résolution des problèmes

Une fois le téléchargement des données terminé, vérifiez si des lignes ont été ignorées. Pour ce faire, accédez au répertoire source du fichier CSV source et recherchez un fichier portant le nom suivant.

```
import_yourcsvfilename.err.timestamp.csv
```

cqlsh écrit toutes les lignes de données ignorées dans un fichier portant ce nom. Si le fichier existe dans votre répertoire source et contient des données, ces lignes n'ont pas été chargées sur Amazon Keyspaces. Pour réessayer ces lignes, vérifiez d'abord si des erreurs se sont produites lors du téléchargement et ajustez les données en conséquence. Pour réessayer ces lignes, vous pouvez relancer le processus.

Erreurs courantes

Les raisons les plus courantes pour lesquelles les lignes ne sont pas chargées sont les erreurs de capacité et les erreurs d'analyse.

Erreurs de demande non valides lors du chargement de données sur Amazon Keyspaces

Dans l'exemple suivant, la table source contient une colonne de compteur, qui génère des appels par lots enregistrés à partir de la commande cqlshCOPY. Les appels groupés enregistrés ne sont pas pris en charge par Amazon Keyspaces.

```
Failed to import 10 rows: InvalidRequest - Error from server: code=2200 [Invalid query]
 message="Only UNLOGGED Batches are supported at this time.",  will retry later,
 attempt 22 of 25
```

Pour résoudre cette erreur, utilisez DSBulk pour migrer les données. Pour de plus amples informations, veuillez consulter the section called “Chargement de données à l'aide de DSBulk”.

Erreurs de l'analyseur lors du chargement de données sur Amazon Keyspaces

L'exemple suivant montre une ligne ignorée en raison d'un`ParseError`.

```
Failed to import 1 rows: ParseError - Invalid ... –
```

Pour résoudre cette erreur, vous devez vous assurer que les données à importer correspondent au schéma de table dans Amazon Keyspaces. Vérifiez le fichier d'importation pour détecter les erreurs d'analyse. Vous pouvez essayer d'utiliser une seule ligne de données à l'aide d'une `INSERT` instruction pour isoler l'erreur.

Erreurs de capacité lors du chargement de données sur Amazon Keyspaces

```
Failed to import 1 rows: WriteTimeout - Error from server: code=1100 [Coordinator node
 timed out waiting for replica nodes' responses]
 message="Operation timed out - received only 0 responses." info={'received_responses':
 0, 'required_responses': 2, 'write_type': 'SIMPLE', 'consistency':
 'LOCAL_QUORUM'}, will retry later, attempt 1 of 100
```

Amazon Keyspaces utilise les `WriteTimeout` exceptions `ReadTimeout` et pour indiquer lorsqu'une demande d'écriture échoue en raison d'une capacité de débit insuffisante. Pour aider à diagnostiquer les exceptions liées à une capacité insuffisante, Amazon Keyspaces publie `WriteThrottleEvents` des `ReadThrottledEvents` statistiques sur Amazon. CloudWatch Pour de plus amples informations, veuillez consulter the section called “Surveillance avec CloudWatch”.

erreurs cqlsh lors du chargement de données sur Amazon Keyspaces

Pour résoudre les erreurs cqlsh, réexécutez la commande défaillante avec l'indicateur. `--debug`

Lorsque vous utilisez une version incompatible de cqlsh, l'erreur suivante s'affiche.

```
AttributeError: 'NoneType' object has no attribute 'is_up'
Failed to import 3 rows: AttributeError - 'NoneType' object has no attribute 'is_up',
 given up after 1 attempts
```

Vérifiez que la version correcte de cqlsh est installée en exécutant la commande suivante.

```
cqlsh --version
```

Vous devriez voir quelque chose comme la sortie suivante.

```
cqlsh 5.0.1
```

Si vous utilisez Windows, remplacez toutes les instances de `cqlsh` par`cqlsh.bat`. Par exemple, pour vérifier la version de cqlsh sous Windows, exécutez la commande suivante.

```
cqlsh.bat --version
```

La connexion à Amazon Keyspaces échoue lorsque le client cqlsh a reçu trois erreurs consécutives de tout type de la part du serveur. Le client cqlsh échoue avec le message suivant.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Pour résoudre cette erreur, vous devez vous assurer que les données à importer correspondent au schéma de table dans Amazon Keyspaces. Vérifiez le fichier d'importation pour détecter les erreurs d'analyse. Vous pouvez essayer d'utiliser une seule ligne de données en utilisant une instruction INSERT pour isoler l'erreur.

Le client tente automatiquement de rétablir la connexion.

Tutoriel : Chargement de données dans Amazon Keyspaces à l'aide de DSBulk

Ce step-by-step didacticiel vous explique comment migrer des données d'Apache Cassandra vers Amazon Keyspaces à l'aide du DataStax Bulk Loader (DSBulk) disponible sur. GitHub Dans ce didacticiel, vous exécuterez les étapes suivantes :

Rubriques

Prérequis

Vous devez effectuer les tâches suivantes avant de pouvoir commencer ce didacticiel.

1. Si ce n'est pas déjà fait, créez un AWS compte en suivant les étapes indiquées surthe section called “Con AWS Identity and Access Management figuration”.
2. Créez des informations d'identification en suivant les étapes décrites dansthe section called “Informations d'identification IAM pour l' AWS authentification”.
3. Créez un fichier JKS Trust Store.

   a. Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

   ```
   curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
   ```

   > **Note**
   >
   > Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire aux clients utilisant des autorités de certification plus anciennes.

   b. Convertissez le certificat numérique Starfield en fichier TrustStore.

   ```
   openssl x509 -outform der -in sf-class2-root.crt -out temp_file.der
   keytool -import -alias cassandra -keystore cassandra_truststore.jks -file
    temp_file.der
   ```

Au cours de cette étape, vous devez créer un mot de passe pour le keystore et faire confiance à ce certificat. La commande interactive ressemble à ceci.

```
Enter keystore password:
Re-enter new password:
Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US
Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield
 Technologies, Inc.", C=US
Serial number: 0
Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034
Certificate fingerprints:
  MD5:  32:4A:4B:BB:C8:63:69:9B:BE:74:9A:C6:DD:1D:46:24
  SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A
  SHA256:
 14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB
Signature algorithm name: SHA1withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3
Extensions:
#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
[OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US]
SerialNumber: [    00]
]
#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]
#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
]
```

```
Trust this certificate? [no]:  y
```

4. Configurez la connexion au shell Cassandra Query Language (cqlsh) et confirmez que vous pouvez vous connecter à Amazon Keyspaces en suivant les étapes indiquées sur. the section called “Utiliser cqlsh”
5. Téléchargez et installez DSBulk.

   a. Pour télécharger DSBulk, vous pouvez utiliser le code suivant.

   ```
   curl -OL https://downloads.datastax.com/dsbulk/dsbulk-1.8.0.tar.gz
   ```

   b. Décompressez ensuite le fichier tar et ajoutez DSBulk à votre fichier, PATH comme indiqué dans l'exemple suivant.

   ```
   tar -zxvf dsbulk-1.8.0.tar.gz
   # add the DSBulk directory to the path
   export PATH=$PATH:./dsbulk-1.8.0/bin
   ```

   c. Créez un `application.conf` fichier pour stocker les paramètres à utiliser par DSBulk. Vous pouvez enregistrer l'exemple suivant sous le nom `./dsbulk_keyspaces.conf`. `localhost`Remplacez-le par le point de contact de votre cluster Cassandra local si vous n'êtes pas sur le nœud local, par exemple le nom DNS ou l'adresse IP. Prenez note du nom et du chemin du fichier, car vous devrez le spécifier ultérieurement dans la `dsbulk load` commande.

   ```
   datastax-java-driver {
     basic.contact-points = [ "localhost"]
     advanced.auth-provider {
           class = software.aws.mcs.auth.SigV4AuthProvider
           aws-region = us-east-1
     }
   }
   ```

   d. Pour activer le support SigV4, téléchargez le `jar` fichier ombré depuis GitHubet placez-le dans le `lib` dossier DSBulk comme indiqué dans l'exemple suivant.

   ```
   curl -O -L https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin/
   releases/download/4.0.6-shaded-v2/aws-sigv4-auth-cassandra-java-driver-
   plugin-4.0.6-shaded.jar
   ```

Étape 1 : Création du fichier CSV source et de la table cible

Pour ce didacticiel, nous utilisons un fichier de valeurs séparées par des virgules (CSV) dont le `keyspaces_sample_table.csv` nom est le fichier source pour la migration des données. Le fichier d'exemple fourni contient quelques lignes de données pour une table portant le `nombook_awards`.

1. Créez le fichier source. Vous pouvez choisir l'une des options suivantes :
   - Téléchargez l'exemple de fichier CSV (`keyspaces_sample_table.csv`) contenu dans le fichier d'archive samplemigration.zip suivant. Décompressez l'archive et notez le chemin vers`keyspaces_sample_table.csv`.
   - Pour remplir un fichier CSV avec vos propres données stockées dans une base de données Apache Cassandra, vous pouvez remplir le fichier CSV source en utilisant `dsbulk unload` comme indiqué dans l'exemple suivant.

     ```
     dsbulk unload -k mykeyspace -t mytable -f ./my_application.conf
      > keyspaces_sample_table.csv
     ```

     Assurez-vous que le fichier CSV que vous créez répond aux exigences suivantes :

     - La première ligne contient les noms des colonnes.
     - Les noms des colonnes du fichier CSV source correspondent aux noms des colonnes de la table cible.
     - Les données sont délimitées par une virgule.
     - Toutes les valeurs de données sont des types de données Amazon Keyspaces valides. veuillez consulter the section called “Types de données”.
2. Créez le keyspace et le tableau cibles dans Amazon Keyspaces.
   a. Connectez-vous à Amazon Keyspaces en utilisant`cqlsh`, en remplaçant le point de terminaison du service, le nom d'utilisateur et le mot de passe dans l'exemple suivant par vos propres valeurs.

      ```
      cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
      p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
      ```

   b. Créez un nouvel espace de touches avec le `catalog` nom indiqué dans l'exemple suivant.

```
CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

c. Une fois que le nouveau keyspace est devenu disponible, utilisez le code suivant pour créer la table `book_awards` cible. Pour en savoir plus sur la création de ressources asynchrones et sur la façon de vérifier si une ressource est disponible, consultez. the section called "Création d'espaces clés"

```
CREATE TABLE catalog.book_awards (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank)
   );
```

Si Apache Cassandra est votre source de données d'origine, un moyen simple de créer la table cible Amazon Keyspaces avec les en-têtes correspondants consiste à générer `CREATE TABLE` l'instruction à partir de la table source, comme indiqué dans l'instruction suivante.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Créez ensuite la table cible dans Amazon Keyspaces avec les noms des colonnes et les types de données correspondant à la description de la table source de Cassandra.

Étape 2 : Préparation des données

La préparation des données sources pour un transfert efficace est un processus en deux étapes. Tout d'abord, vous répartissez les données de manière aléatoire. Dans la deuxième étape, vous analysez les données pour déterminer les valeurs de `dsbulk` paramètres appropriées et les paramètres de table requis.

Randomiser les données

La `dsbulk` commande lit et écrit les données dans l'ordre dans lequel elles apparaissent dans le fichier CSV. Si vous utilisez la `dsbulk` commande pour créer le fichier source, les données sont écrites dans un ordre trié par clé dans le fichier CSV. En interne, Amazon Keyspaces partitionne les données à l'aide de clés de partition. Bien qu'Amazon Keyspaces intègre une logique permettant d'équilibrer la charge des demandes pour la même clé de partition, le chargement des données est plus rapide et plus efficace si vous répartissez la commande de manière aléatoire. En effet, vous pouvez tirer parti de l'équilibrage de charge intégré qui se produit lorsque Amazon Keyspaces écrit sur différentes partitions.

Pour répartir uniformément les écritures sur les partitions, vous devez répartir les données de manière aléatoire dans le fichier source. Vous pouvez écrire une application pour cela ou utiliser un outil open source tel que Shuf. Shuf est disponible gratuitement sur les distributions Linux, sur macOS (en installant coreutils dans homebrew) et sur Windows (en utilisant le sous-système Windows pour Linux (WSL)). Une étape supplémentaire est nécessaire pour éviter que la ligne d'en-tête contenant les noms des colonnes ne soit modifiée au cours de cette étape.

Pour randomiser le fichier source tout en préservant l'en-tête, entrez le code suivant.

```
tail -n +2 keyspaces_sample_table.csv | shuf -o keyspace.table.csv && (head
 -1 keyspaces_sample_table.csv && cat keyspace.table.csv ) > keyspace.table.csv1 &&
 mv keyspace.table.csv1 keyspace.table.csv
```

Shuf réécrit les données dans un nouveau fichier CSV appelé. `keyspace.table.csv` Vous pouvez désormais supprimer le `keyspaces_sample_table.csv` fichier, vous n'en avez plus besoin.

Analyser les données

Déterminez la taille moyenne et maximale des lignes en analysant les données.

Vous le faites pour les raisons suivantes :

- La taille moyenne des lignes permet d'estimer la quantité totale de données à transférer.
- Vous avez besoin de la taille de ligne moyenne pour fournir la capacité d'écriture nécessaire au téléchargement des données.
- Vous pouvez vous assurer que la taille de chaque ligne est inférieure à 1 Mo, ce qui correspond à la taille de ligne maximale dans Amazon Keyspaces.

> **Note**
>
> Ce quota fait référence à la taille des lignes et non à la taille de la partition. Contrairement aux partitions Apache Cassandra, les partitions Amazon Keyspaces peuvent être de taille pratiquement indépendante. Les clés de partition et les colonnes de clustering nécessitent un espace de stockage supplémentaire pour les métadonnées, que vous devez ajouter à la taille brute des lignes. Pour plus d'informations, consultez the section called "Calcul de la taille des lignes".

Le code suivant utilise AWK pour analyser un fichier CSV et imprimer la taille de ligne moyenne et maximale.

```
awk -F, 'BEGIN {samp=10000;max=-1;}{if(NR>1){len=length($0);t+=len;avg=t/
NR;max=(len>max ? len : max)}}NR==samp{exit}END{printf("{lines: %d, average: %d bytes,
 max: %d bytes}\n",NR,avg,max);}' keyspace.table.csv
```

L'exécution de ce code génère le résultat suivant.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Assurez-vous que la taille maximale de votre ligne ne dépasse pas 1 Mo. Si c'est le cas, vous devez diviser la ligne ou compresser les données pour ramener la taille de la ligne en dessous de 1 Mo. Dans l'étape suivante de ce didacticiel, vous allez utiliser la taille de ligne moyenne pour allouer la capacité d'écriture de la table.

Étape 3 : définir la capacité de débit de la table

Ce didacticiel explique comment régler DSBulk pour charger des données dans un intervalle de temps défini. Comme vous savez à l'avance combien de lectures et d'écritures vous devez effectuer, utilisez le mode capacité provisionnée. Une fois le transfert de données terminé, vous devez définir le mode de capacité de la table en fonction des modèles de trafic de votre application. Pour en savoir plus sur la gestion des capacités, voirGestion des ressources sans serveur.

Avec le mode capacité provisionnée, vous spécifiez à l'avance la capacité de lecture et d'écriture que vous souhaitez allouer à votre table. La capacité d'écriture est facturée à l'heure et mesurée en unités de capacité d'écriture (WCU). Chaque WCU possède une capacité d'écriture suffisante pour prendre en charge l'écriture de 1 Ko de données par seconde. Lorsque vous chargez les données, le taux

d'écriture doit être inférieur au nombre maximum de WCU (paramètre :`write_capacity_units`) défini sur la table cible.

Par défaut, vous pouvez allouer jusqu'à 40 000 WCU à une table et 80 000 WCU pour toutes les tables de votre compte. Si vous avez besoin de capacités supplémentaires, vous pouvez demander une augmentation de quota dans la console Service Quotas. Pour de plus amples informations sur les quotas, veuillez consulter Quotas.

Calculez le nombre moyen de WCU nécessaires pour un insert

L'insertion de 1 Ko de données par seconde nécessite 1 WCU. Si votre fichier CSV comporte 360 000 lignes et que vous souhaitez charger toutes les données en 1 heure, vous devez écrire 100 lignes par seconde (360 000 lignes/60 minutes/60 secondes = 100 lignes par seconde). Si chaque ligne contient jusqu'à 1 Ko de données, pour insérer 100 lignes par seconde, vous devez allouer 100 WCU à votre table. Si chaque ligne contient 1,5 Ko de données, vous avez besoin de deux WCU pour insérer une ligne par seconde. Par conséquent, pour insérer 100 lignes par seconde, vous devez provisionner 200 WCU.

Pour déterminer le nombre de WCU dont vous avez besoin pour insérer une ligne par seconde, divisez la taille moyenne des lignes en octets par 1024 et arrondissez au nombre entier le plus proche.

Par exemple, si la taille moyenne des lignes est de 3 000 octets, vous avez besoin de trois WCU pour insérer une ligne par seconde.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Calculer le temps et la capacité de chargement des données

Maintenant que vous connaissez la taille moyenne et le nombre de lignes de votre fichier CSV, vous pouvez calculer le nombre de WCU nécessaires pour charger les données dans un laps de temps donné, ainsi que le temps approximatif nécessaire pour charger toutes les données de votre fichier CSV en utilisant différents paramètres WCU.

Par exemple, si chaque ligne de votre fichier fait 1 Ko et que votre fichier CSV contient 1 000 000 lignes, pour charger les données en 1 heure, vous devez allouer au moins 278 WCU à votre table pendant cette heure.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Configuration des paramètres de capacité provisionnée

Vous pouvez définir les paramètres de capacité d'écriture d'une table lorsque vous créez la table ou à l'aide de la `ALTER TABLE` commande. Voici la syntaxe permettant de modifier les paramètres de capacité provisionnée d'une table à l'aide de la `ALTER TABLE` commande.

```
ALTER TABLE catalog.book_awards WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100, 'write_capacity_units':
 278}} ;
```

Pour la référence linguistique complète, voir the section called “CREATE TABLE” etthe section called “ALTER TABLE”.

Étape 4 : Configuration des **`DSBulk`** paramètres

Cette section décrit les étapes requises pour configurer DSBulk pour le téléchargement de données vers Amazon Keyspaces. Vous configurez DSBulk à l'aide d'un fichier de configuration. Vous spécifiez le fichier de configuration directement depuis la ligne de commande.

1. Créez un fichier de configuration DSBulk pour la migration vers Amazon Keyspaces. Dans cet exemple, nous utilisons le nom du fichier. `dsbulk_keyspaces.conf` Spécifiez les paramètres suivants dans le fichier de configuration DSBulk.

   a. *`PlainTextAuthProvider`*— Créez le fournisseur d'authentification avec la `PlainTextAuthProvider` classe. `ServiceUserName`et `ServicePassword` doivent correspondre au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service en suivant les étapes décrites dans. the section called “Création d'informations d'identification”

   b. *`local-datacenter`*— Définissez la valeur de Région AWS celle `local-datacenter` à laquelle vous vous connectez. Par exemple, si l'application se connecte à`cassandra.us-east-2.amazonaws.com`, définissez le centre de données local sur`us-east-2`. Pour toutes les options disponibles Régions AWS, voirthe section called “Points de terminaison de service”. Pour éviter les répliques, définissez sur`slow-replica-avoidance`. `false`

   c. *`SSLEngineFactory`*— Pour configurer SSL/TLS, initialisez-le `SSLEngineFactory` en ajoutant une section dans le fichier de configuration avec une seule ligne qui spécifie la classe avec. `class = DefaultSslEngineFactory` Indiquez le chemin d'accès `cassandra_truststore.jks` et le mot de passe que vous avez créés précédemment.

d. *consistency*— Réglez le niveau de cohérence sur`LOCAL_QUORUM`. Les autres niveaux de cohérence d'écriture ne sont pas pris en charge. Pour plus d'informations, consultezthe section called “Niveaux de cohérence Cassandra pris en charge”.

e. Le nombre de connexions par pool est configurable dans le pilote Java. Pour cet exemple, définissez `advanced.connection.pool.local.size` sur 3.

L'exemple de fichier de configuration complet est le suivant.

```
datastax-java-driver {
basic.contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
advanced.auth-provider {
    class = PlainTextAuthProvider
    username = "ServiceUserName"
    password = "ServicePassword"
}

basic.load-balancing-policy {
    local-datacenter = "us-east-2"
    slow-replica-avoidance = false
}

basic.request {
    consistency = LOCAL_QUORUM
    default-idempotence = true
}
advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "./cassandra_truststore.jks"
    truststore-password = "my_password"
    hostname-validation = false
  }
advanced.connection.pool.local.size = 3
}
```

2. Vérifiez les paramètres de la `load` commande DSBulk.

a. *executor.maxPerSecond*— Le nombre maximum de lignes que la commande de chargement tente de traiter simultanément par seconde. S'il n'est pas défini, ce paramètre est désactivé avec -1.

`executor.maxPerSecond`Défini en fonction du nombre de WCU que vous avez provisionnés dans la table de destination cible. La `executor.maxPerSecond` `load` commande n'est pas une limite, c'est une moyenne cible. Cela signifie qu'il peut (et c'est souvent le cas) dépasser le nombre que vous avez défini. Pour tenir compte des rafales et vous assurer que la capacité est suffisante pour traiter les demandes de chargement de données, définissez 90 % `executor.maxPerSecond` de la capacité d'écriture de la table.

```
executor.maxPerSecond = WCUs * .90
```

Dans ce didacticiel, nous avons défini `executor.maxPerSecond` la valeur 5.

> **Note**
>
> Si vous utilisez DSBulk 1.6.0 ou une version ultérieure, vous pouvez utiliser `dsbulk.engine.maxConcurrentQueries` à la place.

b. Configurez ces paramètres supplémentaires pour la `load` commande DSBulk.

- *`batch-mode`*— Ce paramètre indique au système de regrouper les opérations par clé de partition. Nous vous recommandons de désactiver le mode batch, car cela peut entraîner des scénarios de raccourcis clavier et des causes`WriteThrottleEvents`.
- *`driver.advanced.retry-policy-max-retries`*— Cela détermine le nombre de tentatives à nouveau pour une requête qui a échoué. S'il n'est pas défini, la valeur par défaut est 10. Vous pouvez ajuster cette valeur selon vos besoins.
- *`driver.basic.request.timeout`*— Durée en minutes pendant laquelle le système attend le retour d'une requête. S'il n'est pas défini, la valeur par défaut est « 5 minutes ». Vous pouvez ajuster cette valeur selon vos besoins.

Étape 5 : Exécuter la commande DSBulk **`load`**

Dans la dernière étape de ce didacticiel, vous téléchargerez les données dans Amazon Keyspaces.

Pour exécuter la `load` commande DSBulk, procédez comme suit.

1. Exécutez le code suivant pour télécharger les données de votre fichier csv dans votre tableau Amazon Keyspaces. Assurez-vous de mettre à jour le chemin du fichier de configuration de l'application que vous avez créé précédemment.

```
dsbulk load -f ./dsbulk_keyspaces.conf  --connector.csv.url keyspace.table.csv
 -header true --batch.mode DISABLED --executor.maxPerSecond 5 --
driver.basic.request.timeout "5 minutes" --driver.advanced.retry-policy.max-
retries 10 -k catalog -t book_awards
```

2. La sortie inclut l'emplacement d'un fichier journal détaillant les opérations réussies et infructueuses. Le fichier est stocké dans le répertoire suivant.

```
Operation directory: /home/user_name/logs/UNLOAD_20210308-202317-801911
```

3. Les entrées du fichier journal incluront des métriques, comme dans l'exemple suivant. Vérifiez que le nombre de lignes correspond au nombre de lignes de votre fichier csv.

```
total | failed | rows/s | p50ms | p99ms | p999ms
   200 |      0 |    200 | 21.63 | 21.89 |  21.89
```

> **Important**
>
> Maintenant que vous avez transféré vos données, ajustez les paramètres du mode capacité de votre table cible pour qu'ils correspondent aux modèles de trafic habituels de votre application. Vous êtes facturé au taux horaire pour votre capacité provisionnée jusqu'à ce que vous la modifiiez. Pour plus d'informations, voir the section called “Modes de capacité de lecture/écriture”.

Exemples de code pour Amazon Keyspaces à l'aide de SDK AWS

Les exemples de code suivants montrent comment utiliser Amazon Keyspaces avec un kit de développement AWS logiciel (SDK).

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés et dans des exemples interservices.

Les Scénarios sont des exemples de code qui vous montrent comment accomplir une tâche spécifique en appelant plusieurs fonctions au sein d'un même service.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Mise en route

Bonjour Amazon Keyspaces

Les exemples de code suivants montrent comment commencer à utiliser Amazon Keyspaces.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace KeyspacesActions;

public class HelloKeyspaces
```

```
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Keyspaces (for Apache
 Cassandra).
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonKeyspaces>()
                .AddTransient<KeyspacesWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloKeyspaces>();

        var keyspacesClient =
 host.Services.GetRequiredService<IAmazonKeyspaces>();
        var keyspacesWrapper = new KeyspacesWrapper(keyspacesClient);

        Console.WriteLine("Hello, Amazon Keyspaces! Let's list your keyspaces:");
        await keyspacesWrapper.ListKeyspaces();
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.KeyspaceSummary;
import software.amazon.awssdk.services.keyspaces.model.KeyspacesException;
import software.amazon.awssdk.services.keyspaces.model.ListKeyspacesRequest;
import software.amazon.awssdk.services.keyspaces.model.ListKeyspacesResponse;
import java.util.List;

/**
 * Before running this Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloKeyspaces {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        KeyspacesClient keyClient = KeyspacesClient.builder()
                .region(region)
                .build();

        listKeyspaces(keyClient);
    }

    public static void listKeyspaces(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
 ListKeyspacesRequest.builder()
                    .maxResults(10)
```

```
                    .build();

            ListKeyspacesResponse response =
 keyClient.listKeyspaces(keyspacesRequest);
            List<KeyspaceSummary> keyspaces = response.keyspaces();
            for (KeyspaceSummary keyspace : keyspaces) {
                System.out.println("The name of the keyspace is " +
 keyspace.keyspaceName());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/**
Before running this Kotlin code example, set up your development environment,
 including your credentials.

For more information, see the following documentation topic:

https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
*/
```

```
suspend fun main() {
    listKeyspaces()
}

suspend fun listKeyspaces() {
    val keyspacesRequest =
        ListKeyspacesRequest {
            maxResults = 10
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.listKeyspaces(keyspacesRequest)
        response.keyspaces?.forEach { keyspace ->
            println("The name of the keyspace is ${keyspace.keyspaceName}")
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import boto3


def hello_keyspaces(keyspaces_client):
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Keyspaces (for Apache
 Cassandra)
    client and list the keyspaces in your account.
    This example uses the default settings specified in your shared credentials
```

```
    and config files.

    :param keyspaces_client: A Boto3 Amazon Keyspaces Client object. This object
 wraps
                             the low-level Amazon Keyspaces service API.
    """
    print("Hello, Amazon Keyspaces! Let's list some of your keyspaces:\n")
    for ks in keyspaces_client.list_keyspaces(maxResults=5).get("keyspaces", []):
        print(ks["keyspaceName"])
        print(f"\t{ks['resourceArn']}")


if __name__ == "__main__":
    hello_keyspaces(boto3.client("keyspaces"))
```

- Pour plus de détails sur l'API, consultez ListKeyspacesle AWS manuel de référence de l'API SDK for Python (Boto3).

Exemples de code

- Actions pour Amazon Keyspaces à l'aide de kits SDK AWS
  - Utilisation CreateKeyspace avec un AWS SDK ou une CLI
  - Utilisation CreateTable avec un AWS SDK ou une CLI
  - Utilisation DeleteKeyspace avec un AWS SDK ou une CLI
  - Utilisation DeleteTable avec un AWS SDK ou une CLI
  - Utilisation GetKeyspace avec un AWS SDK ou une CLI
  - Utilisation GetTable avec un AWS SDK ou une CLI
  - Utilisation ListKeyspaces avec un AWS SDK ou une CLI
  - Utilisation ListTables avec un AWS SDK ou une CLI
  - Utilisation RestoreTable avec un AWS SDK ou une CLI
  - Utilisation UpdateTable avec un AWS SDK ou une CLI
- Scénarios pour Amazon Keyspaces utilisant des SDK AWS
  - Commencez à utiliser les espaces de touches et les tables Amazon Keyspaces à l'aide d'un SDK AWS

Actions pour Amazon Keyspaces à l'aide de kits SDK AWS

Les exemples de code suivants montrent comment effectuer des actions Amazon Keyspaces individuelles à l'aide AWS des SDK. Ces extraits appellent l'API Amazon Keyspaces et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour une liste complète, consultez le manuel Amazon Keyspaces (pour Apache Cassandra) API Reference.

Exemples

- Utilisation CreateKeyspace avec un AWS SDK ou une CLI
- Utilisation CreateTable avec un AWS SDK ou une CLI
- Utilisation DeleteKeyspace avec un AWS SDK ou une CLI
- Utilisation DeleteTable avec un AWS SDK ou une CLI
- Utilisation GetKeyspace avec un AWS SDK ou une CLI
- Utilisation GetTable avec un AWS SDK ou une CLI
- Utilisation ListKeyspaces avec un AWS SDK ou une CLI
- Utilisation ListTables avec un AWS SDK ou une CLI
- Utilisation RestoreTable avec un AWS SDK ou une CLI
- Utilisation UpdateTable avec un AWS SDK ou une CLI

Utilisation **`CreateKeyspace`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateKeyspace`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create a new keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name for the new keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new keyspace.</returns>
    public async Task<string> CreateKeyspace(string keyspaceName)
    {
        var response =
            await _amazonKeyspaces.CreateKeyspaceAsync(
                new CreateKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyspaceà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public static void createKeySpace(KeyspacesClient keyClient, String
keyspaceName) {
```

```
        try {
            CreateKeyspaceRequest keyspaceRequest =
 CreateKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            CreateKeyspaceResponse response =
 keyClient.createKeyspace(keyspaceRequest);
            System.out.println("The ARN of the KeySpace is " +
 response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyspaceà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyspaceà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_keyspace(self, name):
        """
        Creates a keyspace.

        :param name: The name to give the keyspace.
        :return: The Amazon Resource Name (ARN) of the new keyspace.
```

```
        """
        try:
            response = self.keyspaces_client.create_keyspace(keyspaceName=name)
            self.ks_name = name
            self.ks_arn = response["resourceArn"]
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.ks_arn
```

- Pour plus de détails sur l'API, consultez CreateKeyspacele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateTable`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateTable`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create a new Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace where the table will be
created.</param>
    /// <param name="schema">The schema for the new table.</param>
    /// <param name="tableName">The name of the new table.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new table.</returns>
    public async Task<string> CreateTable(string keyspaceName, SchemaDefinition
schema, string tableName)
    {
        var request = new CreateTableRequest
        {
            KeyspaceName = keyspaceName,
            SchemaDefinition = schema,
            TableName = tableName,
            PointInTimeRecovery = new PointInTimeRecovery { Status =
PointInTimeRecoveryStatus.ENABLED }
        };

        var response = await _amazonKeyspaces.CreateTableAsync(request);
        return response.ResourceArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateTableà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public static void createTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
       try {
           // Set the columns.
           ColumnDefinition defTitle = ColumnDefinition.builder()
                   .name("title")
                   .type("text")
                   .build();

           ColumnDefinition defYear = ColumnDefinition.builder()
                   .name("year")
                   .type("int")
                   .build();

           ColumnDefinition defReleaseDate = ColumnDefinition.builder()
                   .name("release_date")
                   .type("timestamp")
                   .build();

           ColumnDefinition defPlot = ColumnDefinition.builder()
                   .name("plot")
                   .type("text")
                   .build();

           List<ColumnDefinition> colList = new ArrayList<>();
           colList.add(defTitle);
           colList.add(defYear);
           colList.add(defReleaseDate);
           colList.add(defPlot);

           // Set the keys.
           PartitionKey yearKey = PartitionKey.builder()
```

```
                .name("year")
                .build();

        PartitionKey titleKey = PartitionKey.builder()
                .name("title")
                .build();

        List<PartitionKey> keyList = new ArrayList<>();
        keyList.add(yearKey);
        keyList.add(titleKey);

        SchemaDefinition schemaDefinition = SchemaDefinition.builder()
                .partitionKeys(keyList)
                .allColumns(colList)
                .build();

        PointInTimeRecovery timeRecovery = PointInTimeRecovery.builder()
                .status(PointInTimeRecoveryStatus.ENABLED)
                .build();

        CreateTableRequest tableRequest = CreateTableRequest.builder()
                .keyspaceName(keySpace)
                .tableName(tableName)
                .schemaDefinition(schemaDefinition)
                .pointInTimeRecovery(timeRecovery)
                .build();

        CreateTableResponse response = keyClient.createTable(tableRequest);
        System.out.println("The table ARN is " + response.resourceArn());

    } catch (KeyspacesException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateTableà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createTable(
    keySpaceVal: String?,
    tableNameVal: String?,
) {
    // Set the columns.
    val defTitle =
        ColumnDefinition {
            name = "title"
            type = "text"
        }

    val defYear =
        ColumnDefinition {
            name = "year"
            type = "int"
        }

    val defReleaseDate =
        ColumnDefinition {
            name = "release_date"
            type = "timestamp"
        }

    val defPlot =
        ColumnDefinition {
            name = "plot"
            type = "text"
        }

    val colList = ArrayList<ColumnDefinition>()
    colList.add(defTitle)
    colList.add(defYear)
```

```
    colList.add(defReleaseDate)
    colList.add(defPlot)

    // Set the keys.
    val yearKey =
        PartitionKey {
            name = "year"
        }

    val titleKey =
        PartitionKey {
            name = "title"
        }

    val keyList = ArrayList<PartitionKey>()
    keyList.add(yearKey)
    keyList.add(titleKey)

    val schemaDefinitionOb =
        SchemaDefinition {
            partitionKeys = keyList
            allColumns = colList
        }

    val timeRecovery =
        PointInTimeRecovery {
            status = PointInTimeRecoveryStatus.Enabled
        }

    val tableRequest =
        CreateTableRequest {
            keyspaceName = keySpaceVal
            tableName = tableNameVal
            schemaDefinition = schemaDefinitionOb
            pointInTimeRecovery = timeRecovery
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateTableà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_table(self, table_name):
        """
        Creates a table in the  keyspace.
        The table is created with a schema for storing movie data
        and has point-in-time recovery enabled.

        :param table_name: The name to give the table.
        :return: The ARN of the new table.
        """
```

```
        try:
            response = self.keyspaces_client.create_table(
                keyspaceName=self.ks_name,
                tableName=table_name,
                schemaDefinition={
                    "allColumns": [
                        {"name": "title", "type": "text"},
                        {"name": "year", "type": "int"},
                        {"name": "release_date", "type": "timestamp"},
                        {"name": "plot", "type": "text"},
                    ],
                    "partitionKeys": [{"name": "year"}, {"name": "title"}],
                },
                pointInTimeRecovery={"status": "ENABLED"},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["resourceArn"]
```

- Pour plus de détails sur l'API, consultez CreateTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteKeyspace`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteKeyspace`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Delete an existing keyspace.
/// </summary>
/// <param name="keyspaceName"></param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteKeyspace(string keyspaceName)
{
    var response = await _amazonKeyspaces.DeleteKeyspaceAsync(
        new DeleteKeyspaceRequest { KeyspaceName = keyspaceName });
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteKeyspaceà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public static void deleteKeyspace(KeyspacesClient keyClient, String
 keyspaceName) {
        try {
            DeleteKeyspaceRequest deleteKeyspaceRequest =
 DeleteKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            keyClient.deleteKeyspace(deleteKeyspaceRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteKeyspaceà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteKeyspaceà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def delete_keyspace(self):
        """
        Deletes the keyspace.
        """
        try:
            self.keyspaces_client.delete_keyspace(keyspaceName=self.ks_name)
            self.ks_name = None
```

```
        except ClientError as err:
            logger.error(
                "Couldn't delete keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteKeyspacele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteTable`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteTable`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an Amazon Keyspaces table.
```

```
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTable(string keyspaceName, string tableName)
    {
        var response = await _amazonKeyspaces.DeleteTableAsync(
            new DeleteTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public static void deleteTable(KeyspacesClient keyClient, String
keyspaceName, String tableName) {
        try {
            DeleteTableRequest tableRequest = DeleteTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            keyClient.deleteTable(tableRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def delete_table(self):
        """
        Deletes the table from the keyspace.
        """
        try:
            self.keyspaces_client.delete_table(
                keyspaceName=self.ks_name, tableName=self.table_name
            )
            self.table_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                self.table_name,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetKeyspace`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetKeyspace`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get data about a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the keyspace.</returns>
```

```
    public async Task<string> GetKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.GetKeyspaceAsync(
            new GetKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetKeyspaceà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public static void checkKeyspaceExistence(KeyspacesClient keyClient, String
keyspaceName) {
       try {
           GetKeyspaceRequest keyspaceRequest = GetKeyspaceRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           GetKeyspaceResponse response =
keyClient.getKeyspace(keyspaceRequest);
           String name = response.keyspaceName();
           System.out.println("The " + name + " KeySpace is ready");

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous GetKeyspaceà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response: GetKeyspaceResponse =
 keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous GetKeyspaceà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def exists_keyspace(self, name):
        """
        Checks whether a keyspace exists.

        :param name: The name of the keyspace to look up.
        :return: True when the keyspace exists. Otherwise, False.
        """
        try:
            response = self.keyspaces_client.get_keyspace(keyspaceName=name)
            self.ks_name = response["keyspaceName"]
            self.ks_arn = response["resourceArn"]
            exists = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Keyspace %s does not exist.", name)
                exists = False
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
```

```
        return exists
```

- Pour plus de détails sur l'API, consultez GetKeyspacele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetTable`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetTable`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the Amazon Keyspaces table.</param>
    /// <returns>The response containing data about the table.</returns>
    public async Task<GetTableResponse> GetTable(string keyspaceName, string
tableName)
    {
```

```
        var response = await _amazonKeyspaces.GetTableAsync(
            new GetTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTableà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public static void checkTable(KeyspacesClient keyClient, String keyspaceName,
String tableName)
            throws InterruptedException {
        try {
            boolean tableStatus = false;
            String status;
            GetTableResponse response = null;
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            while (!tableStatus) {
                response = keyClient.getTable(tableRequest);
                status = response.statusAsString();
                System.out.println(". The table status is " + status);

                if (status.compareTo("ACTIVE") == 0) {
                    tableStatus = true;
                }
```

```
                Thread.sleep(500);
            }

            List<ColumnDefinition> cols =
 response.schemaDefinition().allColumns();
            for (ColumnDefinition def : cols) {
                System.out.println("The column name is " + def.name());
                System.out.println("The column type is " + def.type());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTableà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun checkTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
```

```
            tableName = tableNameVal
        }
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
        val cols: List<ColumnDefinition>? =
 response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous GetTableà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""
```

```
    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def get_table(self, table_name):
        """
        Gets data about a table in the keyspace.

        :param table_name: The name of the table to look up.
        :return: Data about the table.
        """
        try:
            response = self.keyspaces_client.get_table(
                keyspaceName=self.ks_name, tableName=table_name
            )
            self.table_name = table_name
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Table %s does not exist.", table_name)
                self.table_name = None
                response = None
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    table_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return response
```

- Pour plus de détails sur l'API, consultez GetTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListKeyspaces`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListKeyspaces`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Lists all keyspaces for the account.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task ListKeyspaces()
    {
        var paginator = _amazonKeyspaces.Paginators.ListKeyspaces(new
ListKeyspacesRequest());

        Console.WriteLine("{0, -30}\t{1}", "Keyspace name", "Keyspace ARN");
        Console.WriteLine(new string('-', Console.WindowWidth));
        await foreach (var keyspace in paginator.Keyspaces)
```

```
        {

Console.WriteLine($"{keyspace.KeyspaceName,-30}\t{keyspace.ResourceArn}");
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public static void listKeyspacesPaginator(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
ListKeyspacesRequest.builder()
                    .maxResults(10)
                    .build();

            ListKeyspacesIterable listRes =
keyClient.listKeyspacesPaginator(keyspacesRequest);
            listRes.stream()
                    .flatMap(r -> r.keyspaces().stream())
                    .forEach(content -> System.out.println(" Name: " +
content.keyspaceName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listKeyspacesPaginator() {
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeyspacesà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
```

```
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def list_keyspaces(self, limit):
        """
        Lists the keyspaces in your account.

        :param limit: The maximum number of keyspaces to list.
        """
        try:
            ks_paginator = self.keyspaces_client.get_paginator("list_keyspaces")
            for page in ks_paginator.paginate(PaginationConfig={"MaxItems":
limit}):
                for ks in page["keyspaces"]:
                    print(ks["keyspaceName"])
                    print(f"\t{ks['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list keyspaces. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez ListKeyspacesle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListTables`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListTables`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Lists the Amazon Keyspaces tables in a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>A list of TableSummary objects.</returns>
    public async Task<List<TableSummary>> ListTables(string keyspaceName)
    {
        var response = await _amazonKeyspaces.ListTablesAsync(new
ListTablesRequest { KeyspaceName = keyspaceName });
        response.Tables.ForEach(table =>
        {

Console.WriteLine($"{table.KeyspaceName}\t{table.TableName}\t{table.ResourceArn}");
        });

        return response.Tables;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListTablesà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public static void listTables(KeyspacesClient keyClient, String keyspaceName)
{
       try {
           ListTablesRequest tablesRequest = ListTablesRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           ListTablesIterable listRes =
keyClient.listTablesPaginator(tablesRequest);
           listRes.stream()
                   .flatMap(r -> r.tables().stream())
                   .forEach(content -> System.out.println(" ARN: " +
content.resourceArn() +
                           " Table name: " + content.tableName()));

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous ListTablesà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
            }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListTablesà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def list_tables(self):
        """
        Lists the tables in the keyspace.
        """
        try:
            table_paginator = self.keyspaces_client.get_paginator("list_tables")
            for page in table_paginator.paginate(keyspaceName=self.ks_name):
                for table in page["tables"]:
                    print(table["tableName"])
                    print(f"\t{table['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list tables in keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez ListTablesle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RestoreTable`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`RestoreTable`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Restores the specified table to the specified point in time.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to restore.</param>
    /// <param name="timestamp">The time to which the table will be restored.</
param>
    /// <returns>The Amazon Resource Name (ARN) of the restored table.</returns>
    public async Task<string> RestoreTable(string keyspaceName, string tableName,
 string restoredTableName, DateTime timestamp)
    {
        var request = new RestoreTableRequest
        {
            RestoreTimestamp = timestamp,
            SourceKeyspaceName = keyspaceName,
            SourceTableName = tableName,
            TargetKeyspaceName = keyspaceName,
```

```
            TargetTableName = restoredTableName
        };

        var response = await _amazonKeyspaces.RestoreTableAsync(request);
        return response.RestoredTableARN;
    }
```

- Pour plus de détails sur l'API, reportez-vous RestoreTableà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public static void restoreTable(KeyspacesClient keyClient, String
keyspaceName, ZonedDateTime utc) {
        try {
            Instant myTime = utc.toInstant();
            RestoreTableRequest restoreTableRequest =
RestoreTableRequest.builder()
                    .restoreTimestamp(myTime)
                    .sourceTableName("Movie")
                    .targetKeyspaceName(keyspaceName)
                    .targetTableName("MovieRestore")
                    .sourceKeyspaceName(keyspaceName)
                    .build();

            RestoreTableResponse response =
keyClient.restoreTable(restoreTableRequest);
            System.out.println("The ARN of the restored table is " +
response.restoredTableARN());

        } catch (KeyspacesException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous RestoreTableà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun restoreTable(
    keyspaceName: String?,
    utc: ZonedDateTime,
) {
    // Create an aws.smithy.kotlin.runtime.time.Instant value.
    val timeStamp =
        aws.smithy.kotlin.runtime.time
            .Instant(utc.toInstant())
    val restoreTableRequest =
        RestoreTableRequest {
            restoreTimestamp = timeStamp
            sourceTableName = "MovieKotlin"
            targetKeyspaceName = keyspaceName
            targetTableName = "MovieRestore"
            sourceKeyspaceName = keyspaceName
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous RestoreTableà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def restore_table(self, restore_timestamp):
        """
        Restores the table to a previous point in time. The table is restored
        to a new table in the same keyspace.

        :param restore_timestamp: The point in time to restore the table. This
 time
```

```
                                must be in UTC format.
        :return: The name of the restored table.
        """
        try:
            restored_table_name = f"{self.table_name}_restored"
            self.keyspaces_client.restore_table(
                sourceKeyspaceName=self.ks_name,
                sourceTableName=self.table_name,
                targetKeyspaceName=self.ks_name,
                targetTableName=restored_table_name,
                restoreTimestamp=restore_timestamp,
            )
        except ClientError as err:
            logger.error(
                "Couldn't restore table %s. Here's why: %s: %s",
                restore_timestamp,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return restored_table_name
```

- Pour plus de détails sur l'API, consultez RestoreTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateTable`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateTable`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Commencez avec les espaces de touches et les tableaux

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Updates the movie table to add a boolean column named watched.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to change.</param>
    /// <returns>The Amazon Resource Name (ARN) of the updated table.</returns>
    public async Task<string> UpdateTable(string keyspaceName, string tableName)
    {
        var newColumn = new ColumnDefinition { Name = "watched", Type =
"boolean" };
        var request = new UpdateTableRequest
        {
            KeyspaceName = keyspaceName,
            TableName = tableName,
            AddColumns = new List<ColumnDefinition> { newColumn }
        };
        var response = await _amazonKeyspaces.UpdateTableAsync(request);
        return response.ResourceArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous UpdateTableà la section Référence des AWS SDK for .NET API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public static void updateTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
       try {
           ColumnDefinition def = ColumnDefinition.builder()
                   .name("watched")
                   .type("boolean")
                   .build();

           UpdateTableRequest tableRequest = UpdateTableRequest.builder()
                   .keyspaceName(keySpace)
                   .tableName(tableName)
                   .addColumns(def)
                   .build();

           keyClient.updateTable(tableRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous UpdateTableà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun updateTable(
    keySpace: String?,
    tableNameVal: String?,
) {
    val def =
        ColumnDefinition {
            name = "watched"
            type = "boolean"
        }

    val tableRequest =
        UpdateTableRequest {
            keyspaceName = keySpace
            tableName = tableNameVal
            addColumns = listOf(def)
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateTableà la section AWS SDK pour la référence de l'API Kotlin.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def update_table(self):
        """
        Updates the schema of the table.

        This example updates a table of movie data by adding a new column
        that tracks whether the movie has been watched.
        """
        try:
            self.keyspaces_client.update_table(
                keyspaceName=self.ks_name,
                tableName=self.table_name,
                addColumns=[{"name": "watched", "type": "boolean"}],
            )
```

```
        except ClientError as err:
            logger.error(
                "Couldn't update table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez UpdateTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Scénarios pour Amazon Keyspaces utilisant des SDK AWS

Les exemples de code suivants vous montrent comment implémenter des scénarios courants dans Amazon Keyspaces à l'aide AWS de SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'Amazon Keyspaces. Chaque scénario inclut un lien vers GitHub, où vous pouvez trouver des instructions sur la façon de configurer et d'exécuter le code.

Exemples

- Commencez à utiliser les espaces de touches et les tables Amazon Keyspaces à l'aide d'un SDK AWS

Commencez à utiliser les espaces de touches et les tables Amazon Keyspaces à l'aide d'un SDK AWS

Les exemples de code suivants montrent comment :

- Créez un espace de touches et un tableau. Le schéma de table contient les données vidéo et la point-in-time restauration est activée.
- Connectez-vous au keyspace à l'aide d'une connexion TLS sécurisée avec authentification SigV4.

- Interrogez la table. Ajoutez, récupérez et mettez à jour les données des films.
- Mettez à jour le tableau. Ajoutez une colonne pour suivre les films visionnés.
- Restaurez l'état précédent de la table et nettoyez les ressources.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using System.Security.Cryptography.X509Certificates;
global using Amazon.Keyspaces;
global using Amazon.Keyspaces.Model;
global using KeyspacesActions;
global using KeyspacesScenario;
global using Microsoft.Extensions.Configuration;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;
global using Newtonsoft.Json;


namespace KeyspacesBasics;

/// <summary>
/// Amazon Keyspaces (for Apache Cassandra) scenario. Shows some of the basic
/// actions performed with Amazon Keyspaces.
/// </summary>
public class KeyspacesBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
```

```
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonKeyspaces>()
            .AddTransient<KeyspacesWrapper>()
            .AddTransient<CassandraWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<KeyspacesBasics>();

        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var keyspacesWrapper =
host.Services.GetRequiredService<KeyspacesWrapper>();
        var uiMethods = new UiMethods();

        var keyspaceName = configuration["KeyspaceName"];
        var tableName = configuration["TableName"];

        bool success; // Used to track the results of some operations.

        uiMethods.DisplayOverview();
        uiMethods.PressEnter();

        // Create the keyspace.
        var keyspaceArn = await keyspacesWrapper.CreateKeyspace(keyspaceName);

        // Wait for the keyspace to be available. GetKeyspace results in a
        // resource not found error until it is ready for use.
        try
        {
            var getKeyspaceArn = "";
```

```
            Console.Write($"Created {keyspaceName}. Waiting for it to become
available. ");
            do
            {
                getKeyspaceArn = await
keyspacesWrapper.GetKeyspace(keyspaceName);
                Console.Write(". ");
            } while (getKeyspaceArn != keyspaceArn);
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine("Waiting for keyspace to be created.");
        }

        Console.WriteLine($"\nThe keyspace {keyspaceName} is ready for use.");

        uiMethods.PressEnter();

        // Create the table.
        // First define the schema.
        var allColumns = new List<ColumnDefinition>
        {
            new ColumnDefinition { Name = "title", Type = "text" },
            new ColumnDefinition { Name = "year", Type = "int" },
            new ColumnDefinition { Name = "release_date", Type = "timestamp" },
            new ColumnDefinition { Name = "plot", Type = "text" },
        };

        var partitionKeys = new List<PartitionKey>
        {
            new PartitionKey { Name = "year", },
            new PartitionKey { Name = "title" },
        };

        var tableSchema = new SchemaDefinition
        {
            AllColumns = allColumns,
            PartitionKeys = partitionKeys,
        };

        var tableArn = await keyspacesWrapper.CreateTable(keyspaceName,
tableSchema, tableName);

        // Wait for the table to be active.
```

```
        try
        {
            var resp = new GetTableResponse();
            Console.Write("Waiting for the new table to be active. ");
            do
            {
                try
                {
                    resp = await keyspacesWrapper.GetTable(keyspaceName,
tableName);
                    Console.Write(".");
                }
                catch (ResourceNotFoundException)
                {
                    Console.Write(".");
                }
            } while (resp.Status != TableStatus.ACTIVE);

            // Display the table's schema.
            Console.WriteLine($"\nTable {tableName} has been created in
{keyspaceName}");
            Console.WriteLine("Let's take a look at the schema.");
            uiMethods.DisplayTitle("All columns");
            resp.SchemaDefinition.AllColumns.ForEach(column =>
            {
                Console.WriteLine($"{column.Name,-40}\t{column.Type,-20}");
            });

            uiMethods.DisplayTitle("Cluster keys");
            resp.SchemaDefinition.ClusteringKeys.ForEach(clusterKey =>
            {

Console.WriteLine($"{clusterKey.Name,-40}\t{clusterKey.OrderBy,-20}");
            });

            uiMethods.DisplayTitle("Partition keys");
            resp.SchemaDefinition.PartitionKeys.ForEach(partitionKey =>
            {
                Console.WriteLine($"{partitionKey.Name}");
            });

            uiMethods.PressEnter();
        }
        catch (ResourceNotFoundException ex)
```

```
        {
            Console.WriteLine($"Error: {ex.Message}");
        }

        // Access Apache Cassandra using the Cassandra drive for C#.
        var cassandraWrapper =
host.Services.GetRequiredService<CassandraWrapper>();
        var movieFilePath = configuration["MovieFile"];

        Console.WriteLine("Let's add some movies to the table we created.");
        var inserted = await cassandraWrapper.InsertIntoMovieTable(keyspaceName,
tableName, movieFilePath);

        uiMethods.PressEnter();

        Console.WriteLine("Added the following movies to the table:");
        var rows = await cassandraWrapper.GetMovies(keyspaceName, tableName);
        uiMethods.DisplayTitle("All Movies");

        foreach (var row in rows)
        {
            var title = row.GetValue<string>("title");
            var year = row.GetValue<int>("year");
            var plot = row.GetValue<string>("plot");
            var release_date = row.GetValue<DateTime>("release_date");
            Console.WriteLine($"{release_date}\t{title}\t{year}\n{plot}");
            Console.WriteLine(uiMethods.SepBar);
        }

        // Update the table schema
        uiMethods.DisplayTitle("Update table schema");
        Console.WriteLine("Now we will update the table to add a boolean field
called watched.");

        // First save the current time as a UTC Date so the original
        // table can be restored later.
        var timeChanged = DateTime.UtcNow;

        // Now update the schema.
        var resourceArn = await keyspacesWrapper.UpdateTable(keyspaceName,
tableName);
        uiMethods.PressEnter();

        Console.WriteLine("Now let's mark some of the movies as watched.");
```

```
        // Pick some files to mark as watched.
        var movieToWatch = rows[2].GetValue<string>("title");
        var watchedMovieYear = rows[2].GetValue<int>("year");
        var changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[6].GetValue<string>("title");
        watchedMovieYear = rows[6].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[9].GetValue<string>("title");
        watchedMovieYear = rows[9].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[10].GetValue<string>("title");
        watchedMovieYear = rows[10].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[13].GetValue<string>("title");
        watchedMovieYear = rows[13].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        uiMethods.DisplayTitle("Watched movies");
        Console.WriteLine("These movies have been marked as watched:");
        rows = await cassandraWrapper.GetWatchedMovies(keyspaceName, tableName);
        foreach (var row in rows)
        {
            var title = row.GetValue<string>("title");
            var year = row.GetValue<int>("year");
            Console.WriteLine($"{title,-40}\t{year,8}");
        }
        uiMethods.PressEnter();

        Console.WriteLine("We can restore the table to its previous state but
that can take up to 20 minutes to complete.");
        string answer;
        do
        {
            Console.WriteLine("Do you want to restore the table? (y/n)");
```

```
            answer = Console.ReadLine();
        } while (answer.ToLower() != "y" && answer.ToLower() != "n");

        if (answer == "y")
        {
            var restoredTableName = $"{tableName}_restored";
            var restoredTableArn = await keyspacesWrapper.RestoreTable(
                keyspaceName,
                tableName,
                restoredTableName,
                timeChanged);
            // Loop and call GetTable until the table is gone. Once it has been
            // deleted completely, GetTable will raise a
ResourceNotFoundException.
            bool wasRestored = false;

            try
            {
                do
                {
                    var resp = await keyspacesWrapper.GetTable(keyspaceName,
restoredTableName);
                    wasRestored = (resp.Status == TableStatus.ACTIVE);
                } while (!wasRestored);
            }
            catch (ResourceNotFoundException)
            {
                // If the restored table raised an error, it isn't
                // ready yet.
                Console.Write(".");
            }
        }

        uiMethods.DisplayTitle("Clean up resources.");

        // Delete the table.
        success = await keyspacesWrapper.DeleteTable(keyspaceName, tableName);

        Console.WriteLine($"Table {tableName} successfully deleted from
{keyspaceName}.");
        Console.WriteLine("Waiting for the table to be removed completely. ");

        // Loop and call GetTable until the table is gone. Once it has been
        // deleted completely, GetTable will raise a ResourceNotFoundException.
```

```
        bool wasDeleted = false;

        try
        {
            do
            {
                var resp = await keyspacesWrapper.GetTable(keyspaceName,
 tableName);
            } while (!wasDeleted);
        }
        catch (ResourceNotFoundException ex)
        {
            wasDeleted = true;
            Console.WriteLine($"{ex.Message} indicates that the table has been
 deleted.");
        }

        // Delete the keyspace.
        success = await keyspacesWrapper.DeleteKeyspace(keyspaceName);
        Console.WriteLine("The keyspace has been deleted and the demo is now
 complete.");
    }
}
```

```
namespace KeyspacesActions;

/// <summary>
/// Performs Amazon Keyspaces (for Apache Cassandra) actions.
/// </summary>
public class KeyspacesWrapper
{
    private readonly IAmazonKeyspaces _amazonKeyspaces;

    /// <summary>
    /// Constructor for the KeyspaceWrapper.
    /// </summary>
    /// <param name="amazonKeyspaces">An Amazon Keyspaces client object.</param>
    public KeyspacesWrapper(IAmazonKeyspaces amazonKeyspaces)
    {
        _amazonKeyspaces = amazonKeyspaces;
    }
```

```
    /// <summary>
    /// Create a new keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name for the new keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new keyspace.</returns>
    public async Task<string> CreateKeyspace(string keyspaceName)
    {
        var response =
            await _amazonKeyspaces.CreateKeyspaceAsync(
                new CreateKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }


    /// <summary>
    /// Create a new Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace where the table will be
created.</param>
    /// <param name="schema">The schema for the new table.</param>
    /// <param name="tableName">The name of the new table.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new table.</returns>
    public async Task<string> CreateTable(string keyspaceName, SchemaDefinition
schema, string tableName)
    {
        var request = new CreateTableRequest
        {
            KeyspaceName = keyspaceName,
            SchemaDefinition = schema,
            TableName = tableName,
            PointInTimeRecovery = new PointInTimeRecovery { Status =
PointInTimeRecoveryStatus.ENABLED }
        };

        var response = await _amazonKeyspaces.CreateTableAsync(request);
        return response.ResourceArn;
    }


    /// <summary>
    /// Delete an existing keyspace.
    /// </summary>
    /// <param name="keyspaceName"></param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.DeleteKeyspaceAsync(
            new DeleteKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTable(string keyspaceName, string tableName)
    {
        var response = await _amazonKeyspaces.DeleteTableAsync(
            new DeleteTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Get data about a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the keyspace.</returns>
    public async Task<string> GetKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.GetKeyspaceAsync(
            new GetKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }


    /// <summary>
    /// Get information about an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the Amazon Keyspaces table.</param>
    /// <returns>The response containing data about the table.</returns>
```

```
   public async Task<GetTableResponse> GetTable(string keyspaceName, string
tableName)
   {
       var response = await _amazonKeyspaces.GetTableAsync(
           new GetTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
       return response;
   }


   /// <summary>
   /// Lists all keyspaces for the account.
   /// </summary>
   /// <returns>Async task.</returns>
   public async Task ListKeyspaces()
   {
       var paginator = _amazonKeyspaces.Paginators.ListKeyspaces(new
ListKeyspacesRequest());

       Console.WriteLine("{0, -30}\t{1}", "Keyspace name", "Keyspace ARN");
       Console.WriteLine(new string('-', Console.WindowWidth));
       await foreach (var keyspace in paginator.Keyspaces)
       {

Console.WriteLine($"{keyspace.KeyspaceName,-30}\t{keyspace.ResourceArn}");
       }
   }


   /// <summary>
   /// Lists the Amazon Keyspaces tables in a keyspace.
   /// </summary>
   /// <param name="keyspaceName">The name of the keyspace.</param>
   /// <returns>A list of TableSummary objects.</returns>
   public async Task<List<TableSummary>> ListTables(string keyspaceName)
   {
       var response = await _amazonKeyspaces.ListTablesAsync(new
ListTablesRequest { KeyspaceName = keyspaceName });
       response.Tables.ForEach(table =>
       {

Console.WriteLine($"{table.KeyspaceName}\t{table.TableName}\t{table.ResourceArn}");
       });
```

```
        return response.Tables;
    }


    /// <summary>
    /// Restores the specified table to the specified point in time.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to restore.</param>
    /// <param name="timestamp">The time to which the table will be restored.</
param>
    /// <returns>The Amazon Resource Name (ARN) of the restored table.</returns>
    public async Task<string> RestoreTable(string keyspaceName, string tableName,
 string restoredTableName, DateTime timestamp)
    {
        var request = new RestoreTableRequest
        {
            RestoreTimestamp = timestamp,
            SourceKeyspaceName = keyspaceName,
            SourceTableName = tableName,
            TargetKeyspaceName = keyspaceName,
            TargetTableName = restoredTableName
        };

        var response = await _amazonKeyspaces.RestoreTableAsync(request);
        return response.RestoredTableARN;
    }


    /// <summary>
    /// Updates the movie table to add a boolean column named watched.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to change.</param>
    /// <returns>The Amazon Resource Name (ARN) of the updated table.</returns>
    public async Task<string> UpdateTable(string keyspaceName, string tableName)
    {
        var newColumn = new ColumnDefinition { Name = "watched", Type =
 "boolean" };
        var request = new UpdateTableRequest
        {
            KeyspaceName = keyspaceName,
            TableName = tableName,
            AddColumns = new List<ColumnDefinition> { newColumn }
```

```
        };
        var response = await _amazonKeyspaces.UpdateTableAsync(request);
        return response.ResourceArn;
    }

}
```

```
using System.Net;
using Cassandra;

namespace KeyspacesScenario;

/// <summary>
/// Class to perform CRUD methods on an Amazon Keyspaces (for Apache Cassandra)
 database.
///
/// NOTE: This sample uses a plain text authenticator for example purposes only.
/// Recommended best practice is to use a SigV4 authentication plugin, if
 available.
/// </summary>
public class CassandraWrapper
{
    private readonly IConfiguration _configuration;
    private readonly string _localPathToFile;
    private const string _certLocation = "https://certs.secureserver.net/
repository/sf-class2-root.crt";
    private const string _certFileName = "sf-class2-root.crt";
    private readonly X509Certificate2Collection _certCollection;
    private X509Certificate2 _amazoncert;
    private Cluster _cluster;

    // User name and password for the service.
    private string _userName = null!;
    private string _pwd = null!;

    public CassandraWrapper()
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
```

```
                true) // Optionally load local settings.
            .Build();

        _localPathToFile = Path.GetTempPath();

        // Get the Starfield digital certificate and save it locally.
        var client = new WebClient();
        client.DownloadFile(_certLocation, $"{_localPathToFile}/
{_certFileName}");

        //var httpClient = new HttpClient();
        //var httpResult = httpClient.Get(fileUrl);
        //using var resultStream = await httpResult.Content.ReadAsStreamAsync();
        //using var fileStream = File.Create(pathToSave);
        //resultStream.CopyTo(fileStream);

        _certCollection = new X509Certificate2Collection();
        _amazoncert = new X509Certificate2($"{_localPathToFile}/
{_certFileName}");

        // Get the user name and password stored in the configuration file.
        _userName = _configuration["UserName"]!;
        _pwd = _configuration["Password"]!;

        // For a list of Service Endpoints for Amazon Keyspaces, see:
        // https://docs.aws.amazon.com/keyspaces/latest/devguide/
programmatic.endpoints.html
        var awsEndpoint = _configuration["ServiceEndpoint"];

        _cluster = Cluster.Builder()
            .AddContactPoints(awsEndpoint)
            .WithPort(9142)
            .WithAuthProvider(new PlainTextAuthProvider(_userName, _pwd))
            .WithSSL(new SSLOptions().SetCertificateCollection(_certCollection))
            .WithQueryOptions(
                new QueryOptions()
                    .SetConsistencyLevel(ConsistencyLevel.LocalQuorum)
                    .SetSerialConsistencyLevel(ConsistencyLevel.LocalSerial))
            .Build();
    }

    /// <summary>
    /// Loads the contents of a JSON file into a list of movies to be
    /// added to the Apache Cassandra table.
```

```
    /// </summary>
    /// <param name="movieFileName">The full path to the JSON file.</param>
    /// <returns>A list of movie objects.</returns>
    public List<Movie> ImportMoviesFromJson(string movieFileName, int numToImport
= 0)
    {
        if (!File.Exists(movieFileName))
        {
            return null!;
        }

        using var sr = new StreamReader(movieFileName);
        string json = sr.ReadToEnd();

        var allMovies = JsonConvert.DeserializeObject<List<Movie>>(json);

        // If numToImport = 0, return all movies in the collection.
        if (numToImport == 0)
        {
            // Now return the entire list of movies.
            return allMovies;
        }
        else
        {
            // Now return the first numToImport entries.
            return allMovies.GetRange(0, numToImport);
        }
    }

    /// <summary>
    /// Insert movies into the movie table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="movieTableName">The Amazon Keyspaces table.</param>
    /// <param name="movieFilePath">The path to the resource file containing
    /// movie data to insert into the table.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> InsertIntoMovieTable(string keyspaceName, string
movieTableName, string movieFilePath, int numToImport = 20)
    {
        // Get some movie data from the movies.json file
        var movies = ImportMoviesFromJson(movieFilePath, numToImport);

        var session = _cluster.Connect(keyspaceName);
```

```
        string insertCql;

        RowSet rs;

        // Now we insert the numToImport movies into the table.
        foreach (var movie in movies)
        {
            // Escape single quote characters in the plot.
            insertCql = $"INSERT INTO {keyspaceName}.{movieTableName}
(title, year, release_date, plot) values($${movie.Title}$$, {movie.Year},
'{movie.Info.Release_Date.ToString("yyyy-MM-dd")}', $${movie.Info.Plot}$$)";
            rs = await session.ExecuteAsync(new SimpleStatement(insertCql));
        }

        return true;
    }

    /// <summary>
    /// Gets all of the movies in the movies table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table.</param>
    /// <returns>A list of row objects containing movie data.</returns>
    public async Task<List<Row>> GetMovies(string keyspaceName, string tableName)
    {
        var session = _cluster.Connect();
        RowSet rs;
        try
        {
            rs = await session.ExecuteAsync(new SimpleStatement($"SELECT * FROM
{keyspaceName}.{tableName}"));

            // Extract the row data from the returned RowSet.
            var rows = rs.GetRows().ToList();
            return rows;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
            return null!;
        }
    }
```

```
   /// <summary>
   /// Mark a movie in the movie table as watched.
   /// </summary>
   /// <param name="keyspaceName">The keyspace containing the table.</param>
   /// <param name="tableName">The name of the table.</param>
   /// <param name="title">The title of the movie to mark as watched.</param>
   /// <param name="year">The year the movie was released.</param>
   /// <returns>A set of rows containing the changed data.</returns>
   public async Task<List<Row>> MarkMovieAsWatched(string keyspaceName, string
tableName, string title, int year)
   {
       var session = _cluster.Connect();
       string updateCql = $"UPDATE {keyspaceName}.{tableName} SET watched=true
WHERE title = $${title}$$ AND year = {year};";
       var rs = await session.ExecuteAsync(new SimpleStatement(updateCql));
       var rows = rs.GetRows().ToList();
       return rows;
   }

   /// <summary>
   /// Retrieve the movies in the movies table where watched is true.
   /// </summary>
   /// <param name="keyspaceName">The keyspace containing the table.</param>
   /// <param name="tableName">The name of the table.</param>
   /// <returns>A list of row objects containing information about movies
   /// where watched is true.</returns>
   public async Task<List<Row>> GetWatchedMovies(string keyspaceName, string
tableName)
   {
       var session = _cluster.Connect();
       RowSet rs;
       try
       {
           rs = await session.ExecuteAsync(new SimpleStatement($"SELECT
title, year, plot FROM {keyspaceName}.{tableName} WHERE watched = true ALLOW
FILTERING"));

           // Extract the row data from the returned RowSet.
           var rows = rs.GetRows().ToList();
           return rows;
       }
       catch (Exception ex)
       {
           Console.WriteLine(ex.Message);
```

```
            return null!;
        }
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for .NET .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/**
 * Before running this Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * Before running this Java code example, you must create a
 * Java keystore (JKS) file and place it in your project's resources folder.
 *
 * This file is a secure file format used to hold certificate information for
 * Java applications. This is required to make a connection to Amazon Keyspaces.
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/keyspaces/latest/devguide/using_java_driver.html
 *
 * This Java example performs the following tasks:
 *
 * 1. Create a keyspace.
 * 2. Check for keyspace existence.
 * 3. List keyspaces using a paginator.
 * 4. Create a table with a simple movie data schema and enable point-in-time
 * recovery.
 * 5. Check for the table to be in an Active state.
 * 6. List all tables in the keyspace.
 * 7. Use a Cassandra driver to insert some records into the Movie table.
 * 8. Get all records from the Movie table.
 * 9. Get a specific Movie.
 * 10. Get a UTC timestamp for the current time.
 * 11. Update the table schema to add a ‘watched’ Boolean column.
 * 12. Update an item as watched.
 * 13. Query for items with watched = True.
 * 14. Restore the table back to the previous state using the timestamp.
 * 15. Check for completion of the restore action.
 * 16. Delete the table.
 * 17. Confirm that both tables are deleted.
 * 18. Delete the keyspace.
 */

public class ScenarioKeyspaces {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    /*
     * Usage:
     * fileName - The name of the JSON file that contains movie data. (Get this
 file
     * from the GitHub repo at resources/sample_file.)
```

```
     * keyspaceName - The name of the keyspace to create.
     */
    public static void main(String[] args) throws InterruptedException,
IOException {
        String fileName = "<Replace with the JSON file that contains movie
data>";
        String keyspaceName = "<Replace with the name of the keyspace to
create>";
        String titleUpdate = "The Family";
        int yearUpdate = 2013;
        String tableName = "Movie";
        String tableNameRestore = "MovieRestore";
        Region region = Region.US_EAST_1;
        KeyspacesClient keyClient = KeyspacesClient.builder()
                .region(region)
                .build();

        DriverConfigLoader loader =
DriverConfigLoader.fromClasspath("application.conf");
        CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Keyspaces example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a keyspace.");
        createKeySpace(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        Thread.sleep(5000);
        System.out.println("2. Check for keyspace existence.");
        checkKeyspaceExistence(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. List keyspaces using a paginator.");
        listKeyspacesPaginator(keyClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("4. Create a table with a simple movie data schema and
enable point-in-time recovery.");
        createTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Check for the table to be in an Active state.");
        Thread.sleep(6000);
        checkTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. List all tables in the keyspace.");
        listTables(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Use a Cassandra driver to insert some records into
the Movie table.");
        Thread.sleep(6000);
        loadData(session, fileName, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Get all records from the Movie table.");
        getMovieData(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Get a specific Movie.");
        getSpecificMovie(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Get a UTC timestamp for the current time.");
        ZonedDateTime utc = ZonedDateTime.now(ZoneOffset.UTC);
        System.out.println("DATETIME = " + Date.from(utc.toInstant()));
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Update the table schema to add a watched Boolean
column.");
        updateTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("12. Update an item as watched.");
        Thread.sleep(10000); // Wait 10 secs for the update.
        updateRecord(session, keyspaceName, titleUpdate, yearUpdate);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Query for items with watched = True.");
        getWatchedData(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Restore the table back to the previous state
using the timestamp.");
        System.out.println("Note that the restore operation can take up to 20
minutes.");
        restoreTable(keyClient, keyspaceName, utc);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("15. Check for completion of the restore action.");
        Thread.sleep(5000);
        checkRestoredTable(keyClient, keyspaceName, "MovieRestore");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("16. Delete both tables.");
        deleteTable(keyClient, keyspaceName, tableName);
        deleteTable(keyClient, keyspaceName, tableNameRestore);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Confirm that both tables are deleted.");
        checkTableDelete(keyClient, keyspaceName, tableName);
        checkTableDelete(keyClient, keyspaceName, tableNameRestore);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("18. Delete the keyspace.");
        deleteKeyspace(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("The scenario has completed successfully.");
        System.out.println(DASHES);
    }

    public static void deleteKeyspace(KeyspacesClient keyClient, String
keyspaceName) {
        try {
            DeleteKeyspaceRequest deleteKeyspaceRequest =
DeleteKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            keyClient.deleteKeyspace(deleteKeyspaceRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkTableDelete(KeyspacesClient keyClient, String
keyspaceName, String tableName)
            throws InterruptedException {
        try {
            String status;
            GetTableResponse response;
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            // Keep looping until table cannot be found and a
ResourceNotFoundException is
            // thrown.
            while (true) {
                response = keyClient.getTable(tableRequest);
                status = response.statusAsString();
                System.out.println(". The table status is " + status);
                Thread.sleep(500);
            }

        } catch (ResourceNotFoundException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
```

```
        System.out.println("The table is deleted");
    }

    public static void deleteTable(KeyspacesClient keyClient, String
keyspaceName, String tableName) {
        try {
            DeleteTableRequest tableRequest = DeleteTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            keyClient.deleteTable(tableRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkRestoredTable(KeyspacesClient keyClient, String
keyspaceName, String tableName)
            throws InterruptedException {
        try {
            boolean tableStatus = false;
            String status;
            GetTableResponse response = null;
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            while (!tableStatus) {
                response = keyClient.getTable(tableRequest);
                status = response.statusAsString();
                System.out.println("The table status is " + status);

                if (status.compareTo("ACTIVE") == 0) {
                    tableStatus = true;
                }
                Thread.sleep(500);
            }

            List<ColumnDefinition> cols =
response.schemaDefinition().allColumns();
```

```
            for (ColumnDefinition def : cols) {
                System.out.println("The column name is " + def.name());
                System.out.println("The column type is " + def.type());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void restoreTable(KeyspacesClient keyClient, String
keyspaceName, ZonedDateTime utc) {
        try {
            Instant myTime = utc.toInstant();
            RestoreTableRequest restoreTableRequest =
RestoreTableRequest.builder()
                    .restoreTimestamp(myTime)
                    .sourceTableName("Movie")
                    .targetKeyspaceName(keyspaceName)
                    .targetTableName("MovieRestore")
                    .sourceKeyspaceName(keyspaceName)
                    .build();

            RestoreTableResponse response =
keyClient.restoreTable(restoreTableRequest);
            System.out.println("The ARN of the restored table is " +
response.restoredTableARN());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getWatchedData(CqlSession session, String keyspaceName) {
        ResultSet resultSet = session
                .execute("SELECT * FROM \"" + keyspaceName + "\".\"Movie\" WHERE
watched = true ALLOW FILTERING;");
        resultSet.forEach(item -> {
            System.out.println("The Movie title is " + item.getString("title"));
            System.out.println("The Movie year is " + item.getInt("year"));
            System.out.println("The plot is " + item.getString("plot"));
        });
```

```
   }

   public static void updateRecord(CqlSession session, String keySpace, String
titleUpdate, int yearUpdate) {
       String sqlStatement = "UPDATE \"" + keySpace
               + "\".\"Movie\" SET watched=true WHERE title = :k0 AND year
= :k1;";
       BatchStatementBuilder builder =
BatchStatement.builder(DefaultBatchType.UNLOGGED);
       builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM);
       PreparedStatement preparedStatement = session.prepare(sqlStatement);
       builder.addStatement(preparedStatement.boundStatementBuilder()
               .setString("k0", titleUpdate)
               .setInt("k1", yearUpdate)
               .build());

       BatchStatement batchStatement = builder.build();
       session.execute(batchStatement);
   }

   public static void updateTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
       try {
           ColumnDefinition def = ColumnDefinition.builder()
                   .name("watched")
                   .type("boolean")
                   .build();

           UpdateTableRequest tableRequest = UpdateTableRequest.builder()
                   .keyspaceName(keySpace)
                   .tableName(tableName)
                   .addColumns(def)
                   .build();

           keyClient.updateTable(tableRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void getSpecificMovie(CqlSession session, String keyspaceName)
{
```

```
       ResultSet resultSet = session.execute(
               "SELECT * FROM \"" + keyspaceName + "\".\"Movie\" WHERE title =
'The Family' ALLOW FILTERING ;");
       resultSet.forEach(item -> {
           System.out.println("The Movie title is " + item.getString("title"));
           System.out.println("The Movie year is " + item.getInt("year"));
           System.out.println("The plot is " + item.getString("plot"));
       });
   }

   // Get records from the Movie table.
   public static void getMovieData(CqlSession session, String keyspaceName) {
       ResultSet resultSet = session.execute("SELECT * FROM \"" + keyspaceName +
"\".\"Movie\";");
       resultSet.forEach(item -> {
           System.out.println("The Movie title is " + item.getString("title"));
           System.out.println("The Movie year is " + item.getInt("year"));
           System.out.println("The plot is " + item.getString("plot"));
       });
   }

   // Load data into the table.
   public static void loadData(CqlSession session, String fileName, String
keySpace) throws IOException {
       String sqlStatement = "INSERT INTO \"" + keySpace + "\".\"Movie\" (title,
year, plot) values (:k0, :k1, :k2)";
       JsonParser parser = new JsonFactory().createParser(new File(fileName));
       com.fasterxml.jackson.databind.JsonNode rootNode = new
ObjectMapper().readTree(parser);
       Iterator<JsonNode> iter = rootNode.iterator();
       ObjectNode currentNode;
       int t = 0;
       while (iter.hasNext()) {

           // Add 20 movies to the table.
           if (t == 20)
               break;
           currentNode = (ObjectNode) iter.next();

           int year = currentNode.path("year").asInt();
           String title = currentNode.path("title").asText();
           String plot = currentNode.path("info").path("plot").toString();

           // Insert the data into the Amazon Keyspaces table.
```

```
            BatchStatementBuilder builder =
BatchStatement.builder(DefaultBatchType.UNLOGGED);
            builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM);
            PreparedStatement preparedStatement = session.prepare(sqlStatement);
            builder.addStatement(preparedStatement.boundStatementBuilder()
                    .setString("k0", title)
                    .setInt("k1", year)
                    .setString("k2", plot)
                    .build());

            BatchStatement batchStatement = builder.build();
            session.execute(batchStatement);
            t++;
        }

        System.out.println("You have added " + t + " records successfully!");
    }

    public static void listTables(KeyspacesClient keyClient, String keyspaceName)
{
        try {
            ListTablesRequest tablesRequest = ListTablesRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            ListTablesIterable listRes =
keyClient.listTablesPaginator(tablesRequest);
            listRes.stream()
                    .flatMap(r -> r.tables().stream())
                    .forEach(content -> System.out.println(" ARN: " +
content.resourceArn() +
                            " Table name: " + content.tableName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkTable(KeyspacesClient keyClient, String keyspaceName,
String tableName)
            throws InterruptedException {
        try {
            boolean tableStatus = false;
```

```
            String status;
            GetTableResponse response = null;
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            while (!tableStatus) {
                response = keyClient.getTable(tableRequest);
                status = response.statusAsString();
                System.out.println(". The table status is " + status);

                if (status.compareTo("ACTIVE") == 0) {
                    tableStatus = true;
                }
                Thread.sleep(500);
            }

            List<ColumnDefinition> cols =
response.schemaDefinition().allColumns();
            for (ColumnDefinition def : cols) {
                System.out.println("The column name is " + def.name());
                System.out.println("The column type is " + def.type());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
        try {
            // Set the columns.
            ColumnDefinition defTitle = ColumnDefinition.builder()
                    .name("title")
                    .type("text")
                    .build();

            ColumnDefinition defYear = ColumnDefinition.builder()
                    .name("year")
                    .type("int")
                    .build();
```

```
            ColumnDefinition defReleaseDate = ColumnDefinition.builder()
                    .name("release_date")
                    .type("timestamp")
                    .build();

            ColumnDefinition defPlot = ColumnDefinition.builder()
                    .name("plot")
                    .type("text")
                    .build();

            List<ColumnDefinition> colList = new ArrayList<>();
            colList.add(defTitle);
            colList.add(defYear);
            colList.add(defReleaseDate);
            colList.add(defPlot);

            // Set the keys.
            PartitionKey yearKey = PartitionKey.builder()
                    .name("year")
                    .build();

            PartitionKey titleKey = PartitionKey.builder()
                    .name("title")
                    .build();

            List<PartitionKey> keyList = new ArrayList<>();
            keyList.add(yearKey);
            keyList.add(titleKey);

            SchemaDefinition schemaDefinition = SchemaDefinition.builder()
                    .partitionKeys(keyList)
                    .allColumns(colList)
                    .build();

            PointInTimeRecovery timeRecovery = PointInTimeRecovery.builder()
                    .status(PointInTimeRecoveryStatus.ENABLED)
                    .build();

            CreateTableRequest tableRequest = CreateTableRequest.builder()
                    .keyspaceName(keySpace)
                    .tableName(tableName)
                    .schemaDefinition(schemaDefinition)
                    .pointInTimeRecovery(timeRecovery)
```

```
                    .build();

            CreateTableResponse response = keyClient.createTable(tableRequest);
            System.out.println("The table ARN is " + response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void listKeyspacesPaginator(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
ListKeyspacesRequest.builder()
                    .maxResults(10)
                    .build();

            ListKeyspacesIterable listRes =
keyClient.listKeyspacesPaginator(keyspacesRequest);
            listRes.stream()
                    .flatMap(r -> r.keyspaces().stream())
                    .forEach(content -> System.out.println(" Name: " +
content.keyspaceName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkKeyspaceExistence(KeyspacesClient keyClient, String
keyspaceName) {
        try {
            GetKeyspaceRequest keyspaceRequest = GetKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            GetKeyspaceResponse response =
keyClient.getKeyspace(keyspaceRequest);
            String name = response.keyspaceName();
            System.out.println("The " + name + " KeySpace is ready");

        } catch (KeyspacesException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createKeySpace(KeyspacesClient keyClient, String
 keyspaceName) {
        try {
            CreateKeyspaceRequest keyspaceRequest =
 CreateKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            CreateKeyspaceResponse response =
 keyClient.createKeyspace(keyspaceRequest);
            System.out.println("The ARN of the KeySpace is " +
 response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/**
 Before running this Kotlin code example, set up your development environment,
 including your credentials.

 For more information, see the following documentation topic:

 https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

 This example uses a secure file format to hold certificate information for
 Kotlin applications. This is required to make a connection to Amazon Keyspaces.
 For more information, see the following documentation topic:

 https://docs.aws.amazon.com/keyspaces/latest/devguide/using_java_driver.html

 This Kotlin example performs the following tasks:

 1. Create a keyspace.
 2. Check for keyspace existence.
 3. List keyspaces using a paginator.
 4. Create a table with a simple movie data schema and enable point-in-time
 recovery.
 5. Check for the table to be in an Active state.
 6. List all tables in the keyspace.
 7. Use a Cassandra driver to insert some records into the Movie table.
 8. Get all records from the Movie table.
 9. Get a specific Movie.
 10. Get a UTC timestamp for the current time.
 11. Update the table schema to add a 'watched' Boolean column.
 12. Update an item as watched.
 13. Query for items with watched = True.
 14. Restore the table back to the previous state using the timestamp.
```

```
 15. Check for completion of the restore action.
 16. Delete the table.
 17. Confirm that both tables are deleted.
 18. Delete the keyspace.
 */

/*
   Usage:
     fileName - The name of the JSON file that contains movie data. (Get this
 file from the GitHub repo at resources/sample_file.)
     keyspaceName - The name of the keyspace to create.
 */
val DASHES: String = String(CharArray(80)).replace("\u0000", "-")

suspend fun main() {
    val fileName = "<Replace with the JSON file that contains movie data>"
    val keyspaceName = "<Replace with the name of the keyspace to create>"
    val titleUpdate = "The Family"
    val yearUpdate = 2013
    val tableName = "MovieKotlin"
    val tableNameRestore = "MovieRestore"

    val loader = DriverConfigLoader.fromClasspath("application.conf")
    val session =
        CqlSession
            .builder()
            .withConfigLoader(loader)
            .build()

    println(DASHES)
    println("Welcome to the Amazon Keyspaces example scenario.")
    println(DASHES)

    println(DASHES)
    println("1. Create a keyspace.")
    createKeySpace(keyspaceName)
    println(DASHES)

    println(DASHES)
    delay(5000)
    println("2. Check for keyspace existence.")
    checkKeyspaceExistence(keyspaceName)
    println(DASHES)
```

```
    println(DASHES)
    println("3. List keyspaces using a paginator.")
    listKeyspacesPaginator()
    println(DASHES)

    println(DASHES)
    println("4. Create a table with a simple movie data schema and enable point-
in-time recovery.")
    createTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("5. Check for the table to be in an Active state.")
    delay(6000)
    checkTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("6. List all tables in the keyspace.")
    listTables(keyspaceName)
    println(DASHES)

    println(DASHES)
    println("7. Use a Cassandra driver to insert some records into the Movie
 table.")
    delay(6000)
    loadData(session, fileName, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("8. Get all records from the Movie table.")
    getMovieData(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("9. Get a specific Movie.")
    getSpecificMovie(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("10. Get a UTC timestamp for the current time.")
    val utc = ZonedDateTime.now(ZoneOffset.UTC)
    println("DATETIME = ${Date.from(utc.toInstant())}")
    println(DASHES)
```

```
    println(DASHES)
    println("11. Update the table schema to add a watched Boolean column.")
    updateTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("12. Update an item as watched.")
    delay(10000) // Wait 10 seconds for the update.
    updateRecord(session, keyspaceName, titleUpdate, yearUpdate)
    println(DASHES)

    println(DASHES)
    println("13. Query for items with watched = True.")
    getWatchedData(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("14. Restore the table back to the previous state using the
timestamp.")
    println("Note that the restore operation can take up to 20 minutes.")
    restoreTable(keyspaceName, utc)
    println(DASHES)

    println(DASHES)
    println("15. Check for completion of the restore action.")
    delay(5000)
    checkRestoredTable(keyspaceName, "MovieRestore")
    println(DASHES)

    println(DASHES)
    println("16. Delete both tables.")
    deleteTable(keyspaceName, tableName)
    deleteTable(keyspaceName, tableNameRestore)
    println(DASHES)

    println(DASHES)
    println("17. Confirm that both tables are deleted.")
    checkTableDelete(keyspaceName, tableName)
    checkTableDelete(keyspaceName, tableNameRestore)
    println(DASHES)

    println(DASHES)
    println("18. Delete the keyspace.")
```

```
    deleteKeyspace(keyspaceName)
    println(DASHES)

    println(DASHES)
    println("The scenario has completed successfully.")
    println(DASHES)
}

suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}

suspend fun checkTableDelete(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var status: String
    var response: GetTableResponse
    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    try {
        KeyspacesClient { region = "us-east-1" }.use { keyClient ->
            // Keep looping until the table cannot be found and a
 ResourceNotFoundException is thrown.
            while (true) {
                response = keyClient.getTable(tableRequest)
                status = response.status.toString()
                println(". The table status is $status")
                delay(500)
            }
        }
    } catch (e: ResourceNotFoundException) {
        println(e.message)
```

```
    }
    println("The table is deleted")
}

suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}

suspend fun checkRestoredTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println("The table status is $status")

            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
```

```
        val cols = response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

suspend fun restoreTable(
    keyspaceName: String?,
    utc: ZonedDateTime,
) {
    // Create an aws.smithy.kotlin.runtime.time.Instant value.
    val timeStamp =
        aws.smithy.kotlin.runtime.time
            .Instant(utc.toInstant())
    val restoreTableRequest =
        RestoreTableRequest {
            restoreTimestamp = timeStamp
            sourceTableName = "MovieKotlin"
            targetKeyspaceName = keyspaceName
            targetTableName = "MovieRestore"
            sourceKeyspaceName = keyspaceName
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
    }
}

fun getWatchedData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"$keyspaceName\".
\"MovieKotlin\" WHERE watched = true ALLOW FILTERING;")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
```

```
    }
}

fun updateRecord(
    session: CqlSession,
    keySpace: String,
    titleUpdate: String?,
    yearUpdate: Int,
) {
    val sqlStatement =
        "UPDATE \"$keySpace\".\"MovieKotlin\" SET watched=true WHERE title = :k0
 AND year = :k1;"
    val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
    builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
    val preparedStatement = session.prepare(sqlStatement)
    builder.addStatement(
        preparedStatement
            .boundStatementBuilder()
            .setString("k0", titleUpdate)
            .setInt("k1", yearUpdate)
            .build(),
    )
    val batchStatement = builder.build()
    session.execute(batchStatement)
}

suspend fun updateTable(
    keySpace: String?,
    tableNameVal: String?,
) {
    val def =
        ColumnDefinition {
            name = "watched"
            type = "boolean"
        }

    val tableRequest =
        UpdateTableRequest {
            keyspaceName = keySpace
            tableName = tableNameVal
            addColumns = listOf(def)
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
```

```
        keyClient.updateTable(tableRequest)
    }
}

fun getSpecificMovie(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet =
        session.execute("SELECT * FROM \"$keyspaceName\".\"MovieKotlin\" WHERE
 title = 'The Family' ALLOW FILTERING ;")

    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Get records from the Movie table.
fun getMovieData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"$keyspaceName\".
\"MovieKotlin\";")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Load data into the table.
fun loadData(
    session: CqlSession,
    fileName: String,
    keySpace: String,
) {
    val sqlStatement =
        "INSERT INTO \"$keySpace\".\"MovieKotlin\" (title, year, plot) values
 (:k0, :k1, :k2)"
    val parser = JsonFactory().createParser(File(fileName))
    val rootNode = ObjectMapper().readTree<JsonNode>(parser)
```

```
    val iter: Iterator<JsonNode> = rootNode.iterator()
    var currentNode: ObjectNode

    var t = 0
    while (iter.hasNext()) {
        if (t == 50) {
            break
        }

        currentNode = iter.next() as ObjectNode
        val year = currentNode.path("year").asInt()
        val title = currentNode.path("title").asText()
        val info = currentNode.path("info").toString()

        // Insert the data into the Amazon Keyspaces table.
        val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
        builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
        val preparedStatement: PreparedStatement = session.prepare(sqlStatement)
        builder.addStatement(
            preparedStatement
                .boundStatementBuilder()
                .setString("k0", title)
                .setInt("k1", year)
                .setString("k2", info)
                .build(),
        )

        val batchStatement = builder.build()
        session.execute(batchStatement)
        t++
    }
}

suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
```

```
                println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
            }
    }
}

suspend fun checkTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
        val cols: List<ColumnDefinition>? =
 response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

suspend fun createTable(
    keySpaceVal: String?,
    tableNameVal: String?,
) {
    // Set the columns.
```

```
    val defTitle =
        ColumnDefinition {
            name = "title"
            type = "text"
        }

    val defYear =
        ColumnDefinition {
            name = "year"
            type = "int"
        }

    val defReleaseDate =
        ColumnDefinition {
            name = "release_date"
            type = "timestamp"
        }

    val defPlot =
        ColumnDefinition {
            name = "plot"
            type = "text"
        }

    val colList = ArrayList<ColumnDefinition>()
    colList.add(defTitle)
    colList.add(defYear)
    colList.add(defReleaseDate)
    colList.add(defPlot)

    // Set the keys.
    val yearKey =
        PartitionKey {
            name = "year"
        }

    val titleKey =
        PartitionKey {
            name = "title"
        }

    val keyList = ArrayList<PartitionKey>()
    keyList.add(yearKey)
    keyList.add(titleKey)
```

```
    val schemaDefinitionOb =
        SchemaDefinition {
            partitionKeys = keyList
            allColumns = colList
        }

    val timeRecovery =
        PointInTimeRecovery {
            status = PointInTimeRecoveryStatus.Enabled
        }

    val tableRequest =
        CreateTableRequest {
            keyspaceName = keySpaceVal
            tableName = tableNameVal
            schemaDefinition = schemaDefinitionOb
            pointInTimeRecovery = timeRecovery
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}

suspend fun listKeyspacesPaginator() {
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}

suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
```

```
        val response: GetKeyspaceResponse =
 keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}

suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Kotlin API reference.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
class KeyspaceScenario:
    """Runs an interactive scenario that shows how to get started using Amazon
 Keyspaces."""

    def __init__(self, ks_wrapper):
        """
        :param ks_wrapper: An object that wraps Amazon Keyspace actions.
        """
        self.ks_wrapper = ks_wrapper

    @demo_func
    def create_keyspace(self):
        """
        1. Creates a keyspace.
        2. Lists up to 10 keyspaces in your account.
        """
        print("Let's create a keyspace.")
        ks_name = q.ask(
            "Enter a name for your new keyspace.\nThe name can contain only
 letters, "
            "numbers and underscores: ",
            q.non_empty,
        )
        if self.ks_wrapper.exists_keyspace(ks_name):
            print(f"A keyspace named {ks_name} exists.")
        else:
            ks_arn = self.ks_wrapper.create_keyspace(ks_name)
            ks_exists = False
            while not ks_exists:
                wait(3)
                ks_exists = self.ks_wrapper.exists_keyspace(ks_name)
```

```
            print(f"Created a new keyspace.\n\t{ks_arn}.")
        print("The first 10 keyspaces in your account are:\n")
        self.ks_wrapper.list_keyspaces(10)

    @demo_func
    def create_table(self):
        """
        1. Creates a table in the keyspace. The table is configured with a schema
to hold
           movie data and has point-in-time recovery enabled.
        2. Waits for the table to be in an active state.
        3. Displays schema information for the table.
        4. Lists tables in the keyspace.
        """
        print("Let's create a table for movies in your keyspace.")
        table_name = q.ask("Enter a name for your table: ", q.non_empty)
        table = self.ks_wrapper.get_table(table_name)
        if table is not None:
            print(
                f"A table named {table_name} already exists in keyspace "
                f"{self.ks_wrapper.ks_name}."
            )
        else:
            table_arn = self.ks_wrapper.create_table(table_name)
            print(f"Created table {table_name}:\n\t{table_arn}")
            table = {"status": None}
            print("Waiting for your table to be ready...")
            while table["status"] != "ACTIVE":
                wait(5)
                table = self.ks_wrapper.get_table(table_name)
        print(f"Your table is {table['status']}. Its schema is:")
        pp(table["schemaDefinition"])
        print("\nThe tables in your keyspace are:\n")
        self.ks_wrapper.list_tables()

    @demo_func
    def ensure_tls_cert(self):
        """
        Ensures you have a TLS certificate available to use to secure the
connection
        to the keyspace. This function downloads a default certificate or lets
you
        specify your own.
        """
```

```
        print("To connect to your keyspace, you must have a TLS certificate.")
        print("Checking for TLS certificate...")
        cert_path = os.path.join(
            os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
        )
        if not os.path.exists(cert_path):
            cert_choice = q.ask(
                f"Press enter to download a certificate from
{QueryManager.CERT_URL} "
                f"or enter the full path to the certificate you want to use: "
            )
            if cert_choice:
                cert_path = cert_choice
            else:
                cert = requests.get(QueryManager.CERT_URL).text
                with open(cert_path, "w") as cert_file:
                    cert_file.write(cert)
        else:
            q.ask(f"Certificate {cert_path} found. Press Enter to continue.")
        print(
            f"Certificate {cert_path} will be used to secure the connection to
your keyspace."
        )
        return cert_path

    @demo_func
    def query_table(self, qm, movie_file):
        """
        1. Adds movies to the table from a sample movie data file.
        2. Gets a list of movies from the table and lets you select one.
        3. Displays more information about the selected movie.
        """
        qm.add_movies(self.ks_wrapper.table_name, movie_file)
        movies = qm.get_movies(self.ks_wrapper.table_name)
        print(f"Added {len(movies)} movies to the table:")
        sel = q.choose("Pick one to learn more about it: ", [m.title for m in
movies])
        movie_choice = qm.get_movie(
            self.ks_wrapper.table_name, movies[sel].title, movies[sel].year
        )
        print(movie_choice.title)
        print(f"\tReleased: {movie_choice.release_date}")
        print(f"\tPlot: {movie_choice.plot}")
```

```
    @demo_func
    def update_and_restore_table(self, qm):
        """
        1. Updates the table by adding a column to track watched movies.
        2. Marks some of the movies as watched.
        3. Gets the list of watched movies from the table.
        4. Restores to a movies_restored table at a previous point in time.
        5. Gets the list of movies from the restored table.
        """
        print("Let's add a column to record which movies you've watched.")
        pre_update_timestamp = datetime.utcnow()
        print(
            f"Recorded the current UTC time of {pre_update_timestamp} so we can
restore the table later."
        )
        self.ks_wrapper.update_table()
        print("Waiting for your table to update...")
        table = {"status": "UPDATING"}
        while table["status"] != "ACTIVE":
            wait(5)
            table = self.ks_wrapper.get_table(self.ks_wrapper.table_name)
        print("Column 'watched' added to table.")
        q.ask(
            "Let's mark some of the movies as watched. Press Enter when you're
ready.\n"
        )
        movies = qm.get_movies(self.ks_wrapper.table_name)
        for movie in movies[:10]:
            qm.watched_movie(self.ks_wrapper.table_name, movie.title, movie.year)
            print(f"Marked {movie.title} as watched.")
        movies = qm.get_movies(self.ks_wrapper.table_name, watched=True)
        print("-" * 88)
        print("The watched movies in our table are:\n")
        for movie in movies:
            print(movie.title)
        print("-" * 88)
        if q.ask(
            "Do you want to restore the table to the way it was before all of
these\n"
            "updates? Keep in mind, this can take up to 20 minutes. (y/n) ",
            q.is_yesno,
        ):
            starting_table_name = self.ks_wrapper.table_name
```

```
            table_name_restored =
self.ks_wrapper.restore_table(pre_update_timestamp)
            table = {"status": "RESTORING"}
            while table["status"] != "ACTIVE":
                wait(10)
                table = self.ks_wrapper.get_table(table_name_restored)
            print(
                f"Restored {starting_table_name} to {table_name_restored} "
                f"at a point in time of {pre_update_timestamp}."
            )
            movies = qm.get_movies(table_name_restored)
            print("Now the movies in our table are:")
            for movie in movies:
                print(movie.title)

    def cleanup(self, cert_path):
        """
        1. Deletes the table and waits for it to be removed.
        2. Deletes the keyspace.

        :param cert_path: The path of the TLS certificate used in the demo. If
the
                          certificate was downloaded during the demo, it is
removed.
        """
        if q.ask(
            f"Do you want to delete your {self.ks_wrapper.table_name} table and "
            f"{self.ks_wrapper.ks_name} keyspace? (y/n) ",
            q.is_yesno,
        ):
            table_name = self.ks_wrapper.table_name
            self.ks_wrapper.delete_table()
            table = self.ks_wrapper.get_table(table_name)
            print("Waiting for the table to be deleted.")
            while table is not None:
                wait(5)
                table = self.ks_wrapper.get_table(table_name)
            print("Table deleted.")
            self.ks_wrapper.delete_keyspace()
            print(
                "Keyspace deleted. If you chose to restore your table during the
"
                "demo, the original table is also deleted."
            )
```

```
            if cert_path == os.path.join(
                os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
            ) and os.path.exists(cert_path):
                os.remove(cert_path)
                print("Removed certificate that was downloaded for this demo.")

    def run_scenario(self):
        logging.basicConfig(level=logging.INFO, format="%(levelname)s:
 %(message)s")

        print("-" * 88)
        print("Welcome to the Amazon Keyspaces (for Apache Cassandra) demo.")
        print("-" * 88)

        self.create_keyspace()
        self.create_table()
        cert_file_path = self.ensure_tls_cert()
        # Use a context manager to ensure the connection to the keyspace is
 closed.
        with QueryManager(
            cert_file_path, boto3.DEFAULT_SESSION, self.ks_wrapper.ks_name
        ) as qm:
            self.query_table(qm, "../../../resources/sample_files/movies.json")
            self.update_and_restore_table(qm)
        self.cleanup(cert_file_path)

        print("\nThanks for watching!")
        print("-" * 88)


if __name__ == "__main__":
    try:
        scenario = KeyspaceScenario(KeyspaceWrapper.from_client())
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

Définissez une classe qui englobe les actions du keyspace et du tableau.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""
```

```
    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_keyspace(self, name):
        """
        Creates a keyspace.

        :param name: The name to give the keyspace.
        :return: The Amazon Resource Name (ARN) of the new keyspace.
        """
        try:
            response = self.keyspaces_client.create_keyspace(keyspaceName=name)
            self.ks_name = name
            self.ks_arn = response["resourceArn"]
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.ks_arn


    def exists_keyspace(self, name):
        """
        Checks whether a keyspace exists.

        :param name: The name of the keyspace to look up.
```

```
        :return: True when the keyspace exists. Otherwise, False.
        """
        try:
            response = self.keyspaces_client.get_keyspace(keyspaceName=name)
            self.ks_name = response["keyspaceName"]
            self.ks_arn = response["resourceArn"]
            exists = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Keyspace %s does not exist.", name)
                exists = False
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return exists


    def list_keyspaces(self, limit):
        """
        Lists the keyspaces in your account.

        :param limit: The maximum number of keyspaces to list.
        """
        try:
            ks_paginator = self.keyspaces_client.get_paginator("list_keyspaces")
            for page in ks_paginator.paginate(PaginationConfig={"MaxItems":
limit}):
                for ks in page["keyspaces"]:
                    print(ks["keyspaceName"])
                    print(f"\t{ks['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list keyspaces. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
    def create_table(self, table_name):
        """
        Creates a table in the  keyspace.
        The table is created with a schema for storing movie data
        and has point-in-time recovery enabled.

        :param table_name: The name to give the table.
        :return: The ARN of the new table.
        """
        try:
            response = self.keyspaces_client.create_table(
                keyspaceName=self.ks_name,
                tableName=table_name,
                schemaDefinition={
                    "allColumns": [
                        {"name": "title", "type": "text"},
                        {"name": "year", "type": "int"},
                        {"name": "release_date", "type": "timestamp"},
                        {"name": "plot", "type": "text"},
                    ],
                    "partitionKeys": [{"name": "year"}, {"name": "title"}],
                },
                pointInTimeRecovery={"status": "ENABLED"},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["resourceArn"]


    def get_table(self, table_name):
        """
        Gets data about a table in the keyspace.

        :param table_name: The name of the table to look up.
        :return: Data about the table.
        """
        try:
```

```
            response = self.keyspaces_client.get_table(
                keyspaceName=self.ks_name, tableName=table_name
            )
            self.table_name = table_name
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Table %s does not exist.", table_name)
                self.table_name = None
                response = None
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    table_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return response


    def list_tables(self):
        """
        Lists the tables in the keyspace.
        """
        try:
            table_paginator = self.keyspaces_client.get_paginator("list_tables")
            for page in table_paginator.paginate(keyspaceName=self.ks_name):
                for table in page["tables"]:
                    print(table["tableName"])
                    print(f"\t{table['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list tables in keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def update_table(self):
        """
        Updates the schema of the table.
```

```
        This example updates a table of movie data by adding a new column
        that tracks whether the movie has been watched.
        """
        try:
            self.keyspaces_client.update_table(
                keyspaceName=self.ks_name,
                tableName=self.table_name,
                addColumns=[{"name": "watched", "type": "boolean"}],
            )
        except ClientError as err:
            logger.error(
                "Couldn't update table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def restore_table(self, restore_timestamp):
        """
        Restores the table to a previous point in time. The table is restored
        to a new table in the same keyspace.

        :param restore_timestamp: The point in time to restore the table. This
time
                                  must be in UTC format.
        :return: The name of the restored table.
        """
        try:
            restored_table_name = f"{self.table_name}_restored"
            self.keyspaces_client.restore_table(
                sourceKeyspaceName=self.ks_name,
                sourceTableName=self.table_name,
                targetKeyspaceName=self.ks_name,
                targetTableName=restored_table_name,
                restoreTimestamp=restore_timestamp,
            )
        except ClientError as err:
            logger.error(
                "Couldn't restore table %s. Here's why: %s: %s",
                restore_timestamp,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
```

```
            )
            raise
        else:
            return restored_table_name


    def delete_table(self):
        """
        Deletes the table from the keyspace.
        """
        try:
            self.keyspaces_client.delete_table(
                keyspaceName=self.ks_name, tableName=self.table_name
            )
            self.table_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_keyspace(self):
        """
        Deletes the keyspace.
        """
        try:
            self.keyspaces_client.delete_keyspace(keyspaceName=self.ks_name)
            self.ks_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

Définissez une classe qui crée une connexion TLS à un espace de touches, s'authentifie avec SigV4 et envoie des requêtes CQL à une table de l'espace de touches.

```
class QueryManager:
    """
    Manages queries to an Amazon Keyspaces (for Apache Cassandra) keyspace.
    Queries are secured by TLS and authenticated by using the Signature V4
 (SigV4)
    AWS signing protocol. This is more secure than sending username and password
    with a plain-text authentication provider.

    This example downloads a default certificate to secure TLS, or lets you
 specify
    your own.

    This example uses a table of movie data to demonstrate basic queries.
    """

    DEFAULT_CERT_FILE = "sf-class2-root.crt"
    CERT_URL = f"https://certs.secureserver.net/repository/sf-class2-root.crt"

    def __init__(self, cert_file_path, boto_session, keyspace_name):
        """
        :param cert_file_path: The path and file name of the certificate used for
 TLS.
        :param boto_session: A Boto3 session. This is used to acquire your AWS
 credentials.
        :param keyspace_name: The name of the keyspace to connect.
        """
        self.cert_file_path = cert_file_path
        self.boto_session = boto_session
        self.ks_name = keyspace_name
        self.cluster = None
        self.session = None

    def __enter__(self):
        """
        Creates a session connection to the keyspace that is secured by TLS and
        authenticated by SigV4.
        """
        ssl_context = SSLContext(PROTOCOL_TLSv1_2)
```

```
        ssl_context.load_verify_locations(self.cert_file_path)
        ssl_context.verify_mode = CERT_REQUIRED
        auth_provider = SigV4AuthProvider(self.boto_session)
        contact_point = f"cassandra.
{self.boto_session.region_name}.amazonaws.com"
        exec_profile = ExecutionProfile(
            consistency_level=ConsistencyLevel.LOCAL_QUORUM,
            load_balancing_policy=DCAwareRoundRobinPolicy(),
        )
        self.cluster = Cluster(
            [contact_point],
            ssl_context=ssl_context,
            auth_provider=auth_provider,
            port=9142,
            execution_profiles={EXEC_PROFILE_DEFAULT: exec_profile},
            protocol_version=4,
        )
        self.cluster.__enter__()
        self.session = self.cluster.connect(self.ks_name)
        return self

    def __exit__(self, *args):
        """
        Exits the cluster. This shuts down all existing session connections.
        """
        self.cluster.__exit__(*args)

    def add_movies(self, table_name, movie_file_path):
        """
        Gets movies from a JSON file and adds them to a table in the keyspace.

        :param table_name: The name of the table.
        :param movie_file_path: The path and file name of a JSON file that
 contains movie data.
        """
        with open(movie_file_path, "r") as movie_file:
            movies = json.loads(movie_file.read())
        stmt = self.session.prepare(
            f"INSERT INTO {table_name} (year, title, release_date, plot) VALUES
 (?, ?, ?, ?);"
        )
        for movie in movies[:20]:
            self.session.execute(
                stmt,
```

```
                parameters=[
                    movie["year"],
                    movie["title"],
                    date.fromisoformat(movie["info"]
["release_date"].partition("T")[0]),
                    movie["info"]["plot"],
                ],
            )

    def get_movies(self, table_name, watched=None):
        """
        Gets the title and year of the full list of movies from the table.

        :param table_name: The name of the movie table.
        :param watched: When specified, the returned list of movies is filtered
 to
                        either movies that have been watched or movies that have
 not
                        been watched. Otherwise, all movies are returned.
        :return: A list of movies in the table.
        """
        if watched is None:
            stmt = SimpleStatement(f"SELECT title, year from {table_name}")
            params = None
        else:
            stmt = SimpleStatement(
                f"SELECT title, year from {table_name} WHERE watched = %s ALLOW
 FILTERING"
            )
            params = [watched]
        return self.session.execute(stmt, parameters=params).all()

    def get_movie(self, table_name, title, year):
        """
        Gets a single movie from the table, by title and year.

        :param table_name: The name of the movie table.
        :param title: The title of the movie.
        :param year: The year of the movie's release.
        :return: The requested movie.
        """
        return self.session.execute(
            SimpleStatement(
                f"SELECT * from {table_name} WHERE title = %s AND year = %s"
```

```
            ),
            parameters=[title, year],
        ).one()

    def watched_movie(self, table_name, title, year):
        """
        Updates a movie as having been watched.

        :param table_name: The name of the movie table.
        :param title: The title of the movie.
        :param year: The year of the movie's release.
        """
        self.session.execute(
            SimpleStatement(
                f"UPDATE {table_name} SET watched=true WHERE title = %s AND year
= %s"
            ),
            parameters=[title, year],
        )
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'Amazon Keyspaces avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Bibliothèques et outils Amazon Keyspaces (pour Apache Cassandra)

Cette section fournit des informations sur les bibliothèques Amazon Keyspaces (pour Apache Cassandra), des exemples de code et des outils.

Rubriques

Bibliothèques et exemples

Vous pouvez trouver les bibliothèques open source et les outils de développement d'Amazon Keyspaces GitHub dans les référentiels AWSet des AWSexemples.

Kit d'outils pour développeurs Amazon Keyspaces (pour Apache Cassandra)

Ce référentiel fournit une image Docker avec des outils de développement utiles pour Amazon Keyspaces. Par exemple, il inclut un fichier CQLSHRC contenant les meilleures pratiques, une extension d'AWSauthentification facultative pour cqlsh et des outils d'assistance permettant d'effectuer des tâches courantes. La boîte à outils est optimisée pour Amazon Keyspaces, mais fonctionne également avec les clusters Apache Cassandra.

https://github.com/aws-samples/amazon-keyspaces-toolkit.

Exemples d'Amazon Keyspaces (pour Apache Cassandra)

Ce référentiel est notre liste officielle d'exemples de code Amazon Keyspaces. Le référentiel est subdivisé en sections par langue (voir Exemples). Chaque langue possède sa propre sous-section d'exemples. Ces exemples illustrent les implémentations et les modèles courants des services Amazon Keyspaces que vous pouvez utiliser lors de la création d'applications.

https://github.com/aws-samples/amazon-keyspaces-examples/.

AWSPlug-ins d'authentification Signature Version 4 (SigV4)

Les plugins vous permettent de gérer l'accès à Amazon Keyspaces à l'aide d'utilisateurs et de rôles AWS Identity and Access Management (IAM).

Java :https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin.

Node.js :https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.

Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.

Allez :https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Exemples et référentiels d'outils de développement mis en évidence

Vous trouverez ci-dessous une sélection d'outils communautaires utiles pour Amazon Keyspaces (pour Apache Cassandra).

Tampons du protocole Amazon Keyspaces

Vous pouvez utiliser Protocol Buffers (Protobuf) avec Amazon Keyspaces pour proposer une alternative aux types définis par l'utilisateur (UDT) d'Apache Cassandra. Protobuf est un format de données multiplateforme gratuit et open source utilisé pour sérialiser des données structurées. Vous pouvez stocker des données Protobuf à l'aide du type de données CQL et `BLOB` refactoriser les UDT tout en préservant les données structurées entre les applications et les langages de programmation.

Ce référentiel fournit un exemple de code qui se connecte à Amazon Keyspaces, crée une nouvelle table et insère une ligne contenant un message Protobuf. Ensuite, la ligne est lue avec une forte cohérence.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/protobuf-user-defined-types

AWS CloudFormationmodèle pour créer un CloudWatch tableau de bord Amazon pour les métriques Amazon Keyspaces (pour Apache Cassandra)

Ce référentiel fournit des AWS CloudFormation modèles permettant de configurer rapidement CloudWatch des métriques pour Amazon Keyspaces. L'utilisation de ce modèle vous permettra de

démarrer plus facilement en fournissant des CloudWatch tableaux de bord prédéfinis déployables contenant des métriques couramment utilisées.

https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates.

Utilisation d'Amazon Keyspaces (pour Apache Cassandra) avec AWS Lambda

Le référentiel contient des exemples qui montrent comment se connecter à Amazon Keyspaces depuis Lambda. Vous trouverez ci-dessous quelques exemples.

C#.NET :. https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/dotnet/datastax-v3/connection-lambda

Java :https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/connection-lambda.

Un autre exemple de Lambda qui montre comment déployer et utiliser Amazon Keyspaces à partir d'un Python Lambda est disponible dans le référentiel suivant.

https://github.com/aws-samples/aws-keyspaces-lambda-python

Utilisation d'Amazon Keyspaces (pour Apache Cassandra) avec Spring

Voici un exemple qui montre comment utiliser Amazon Keyspaces avec Spring Boot.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

Utilisation d'Amazon Keyspaces (pour Apache Cassandra) avec Scala

Cet exemple montre comment se connecter à Amazon Keyspaces à l'aide du plugin d'authentification SigV4 avec Scala.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/connection-sigv4

Utilisation d'Amazon Keyspaces (pour Apache Cassandra) avec AWS Glue

Voici un exemple qui montre comment utiliser Amazon Keyspaces avecAWS Glue.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue

Amazon Keyspaces (pour Apache Cassandra) Langage de requête Cassandra (CQL) vers convertisseur AWS CloudFormation

Ce package implémente un outil de ligne de commande pour convertir les scripts CQL (Apache Cassandra Query Language) en modèles AWS CloudFormation (CloudFormation), ce qui permet de gérer facilement les schémas Amazon Keyspaces sous forme de piles. CloudFormation

https://github.com/aws/amazon-keyspaces-cql-to-cfn-converter.

Amazon Keyspaces (pour Apache Cassandra), aides pour le pilote Apache Cassandra pour Java

Ce référentiel contient des politiques relatives aux pilotes, des exemples et des bonnes pratiques lors de l'utilisation du pilote DataStax Java avec Amazon Keyspaces (pour Apache Cassandra).

https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

Démo de compression rapide d'Amazon Keyspaces (pour Apache Cassandra)

Ce référentiel montre comment compresser, stocker et lire/écrire des objets volumineux pour des performances plus rapides et une réduction du débit et des coûts de stockage.

https://github.com/aws-samples/amazon-keyspaces-compression-example.

Démonstration du codec Amazon Keyspaces (pour Apache Cassandra) et Amazon S3

Le codec Amazon S3 personnalisé prend en charge le mappage transparent et configurable par l'utilisateur des pointeurs UUID vers les objets Amazon S3.

https://github.com/aws-samples/amazon-keyspaces-large-object-s3-demo.

Intégration d'Amazon Keyspaces à Apache Spark

Apache Spark est un moteur open source pour l'analyse de données à grande échelle. Apache Spark vous permet d'effectuer des analyses sur les données stockées dans Amazon Keyspaces de manière plus efficace. Vous pouvez également utiliser Amazon Keyspaces pour fournir à des applications des informations cohérentes,single-digit-millisecondaccès en lecture aux données d'analyse depuis Spark. Le connecteur open source Spark Cassandra simplifie la lecture et l'écriture de données entre Amazon Keyspaces et Spark.

La prise en charge du connecteur Cassandra par Amazon Keyspaces rationalise l'exécution des charges de travail Cassandra dans les pipelines d'analyse basés sur Spark en utilisant un service de base de données entièrement géré et sans serveur. Avec Amazon Keyspaces, vous n'avez pas à craindre que Spark soit en concurrence pour les mêmes ressources d'infrastructure sous-jacentes que vos tables. Les tables Amazon Keyspaces augmentent ou diminuent automatiquement en fonction du trafic de votre application.

Le didacticiel suivant explique les étapes et les meilleures pratiques requises pour lire et écrire des données sur Amazon Keyspaces à l'aide du connecteur Spark Cassandra. Le didacticiel explique comment migrer des données vers Amazon Keyspaces en chargeant des données depuis un fichier à l'aide du connecteur Spark Cassandra et en les écrivant dans une table Amazon Keyspaces.
Le didacticiel explique ensuite comment relire les données depuis Amazon Keyspaces à l'aide du connecteur Spark Cassandra. Vous le feriez pour exécuter les charges de travail Cassandra dans des pipelines d'analyse basés sur Spark.

Rubriques

Conditions préalables à l'établissement de connexions à Amazon Keyspaces avec le connecteur Spark Cassandra

Avant de vous connecter à Amazon Keyspaces à l'aide du connecteur Spark Cassandra, vous devez vous assurer que vous avez installé les éléments suivants. La compatibilité d'Amazon Keyspaces avec le connecteur Spark Cassandra a été testée avec les versions recommandées suivantes :

- Version 8 de Java
- Scala 2.12
- Spark 3.4
- Cassandra Connector 2.5 et versions supérieures
- Pilote Cassandra 4.12

1. Pour installer Scala, suivez les instructions surhttps://www.scala-lang.org/download/scala2.html.
2. Pour installer Spark 3.4.1, suivez cet exemple.

```
curl -o spark-3.4.1-bin-hadoop3.tgz -k https://dlcdn.apache.org/spark/spark-3.4.1/
spark-3.4.1-bin-hadoop3.tgz

# now to untar
tar -zxvf spark-3.4.1-bin-hadoop3.tgz

# set this variable.
export SPARK_HOME=$PWD/spark-3.4.1-bin-hadoop3
```

Étape 1 : Configuration d'Amazon Keyspaces pour l'intégration avec le connecteur Apache Cassandra Spark

Au cours de cette étape, vous confirmez que le partitionneur de votre compte est compatible avec le connecteur Apache Spark et vous configurez les autorisations IAM requises. Les meilleures pratiques suivantes vous aident à fournir une capacité de lecture/écriture suffisante pour le tableau.

1. Confirmez que`Murmur3Partitioner`le partitionneur est le partitionneur par défaut pour votre compte. Ce partitionneur est compatible avec le connecteur Spark Cassandra. Pour

plus d'informations sur les partitionneurs et sur la façon de les modifier, voirthe section called “Utilisation de partitionneurs”.

2. Configurez vos autorisations IAM pour Amazon Keyspaces, à l'aide des points de terminaison VPC de l'interface, avec Apache Spark.

   - Attribuez un accès en lecture/écriture à la table utilisateur et un accès en lecture aux tables système comme indiqué dans l'exemple de politique IAM ci-dessous.
   - Pour les clients accédant à Amazon Keyspaces avec Spark over, vous devez renseigner le tableau system.peers avec les points de terminaison de votre interface disponible.Points de terminaison VPC.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      },
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

3. Suivez les bonnes pratiques suivantes pour configurer une capacité de débit de lecture/écriture suffisante pour que votre table Amazon Keyspaces prenne en charge le trafic provenant du connecteur Spark Cassandra.

   - Commencez à utiliser la capacité à la demande pour vous aider à tester le scénario.
   - Pour optimiser le coût du débit des tables pour les environnements de production, utilisez un limiteur de débit pour le trafic provenant du connecteur et configurez votre table pour utiliser la capacité provisionnée avec un dimensionnement automatique. Pour plus d'informations, veuillez consulter the section called “Gérez la capacité de débit grâce à la mise à l'échelle automatique”.
   - Vous pouvez utiliser un limiteur de débit fixe fourni avec le pilote Cassandra. Il y a quelqueslimiteurs de débit adaptés à Amazon Keyspacesdans leAWSéchantillonsrepo.
   - Pour plus d'informations sur la gestion des capacités, voirthe section called “Modes de capacité de lecture/écriture”.

Étape 2 : Configuration du connecteur Apache Cassandra Spark

Apache Spark est une plateforme informatique polyvalente que vous pouvez configurer de différentes manières. Pour configurer Spark et le connecteur Spark Cassandra en vue de leur intégration à Amazon Keyspaces, nous vous recommandons de commencer par les paramètres de configuration minimaux décrits dans la section suivante, puis de les augmenter ultérieurement en fonction de votre charge de travail.

- Créez des partitions Spark d'une taille inférieure à 8 Mo.

  Dans Spark,des cloisonsreprésentent un bloc atomique de données qui peut être exécuté en parallèle. Lorsque vous écrivez des données sur Amazon Keyspaces à l'aide du connecteur Spark Cassandra, plus la partition Spark est petite, plus le nombre d'enregistrements que la tâche va écrire est faible. Si une tâche Spark rencontre plusieurs erreurs, elle échoue une fois que le nombre de tentatives indiqué est épuisé. Pour éviter de rejouer des tâches volumineuses et de retraiter un grand nombre de données, limitez la taille de la partition Spark.
- Utilisez un faible nombre d'écritures simultanées par exécuteur avec un grand nombre de tentatives.

  Amazon Keyspaces renvoie les erreurs de capacité insuffisante aux pilotes Cassandra sous forme de délais d'exécution. Vous ne pouvez pas remédier aux délais d'attente causés par une capacité insuffisante en modifiant la durée du délai d'expiration configuré, car le

connecteur Spark Cassandra tente de réessayer les demandes de manière transparente à l'aide du`MultipleRetryPolicy`. Pour éviter que les nouvelles tentatives ne surchargent le pool de connexions du pilote, utilisez un faible nombre d'écritures simultanées par exécuteur avec un grand nombre de tentatives. L'extrait de code suivant en est un exemple.

```
spark.cassandra.query.retry.count = 500
spark.cassandra.output.concurrent.writes = 3
```

- Répartissez le débit total et répartissez-le sur plusieurs sessions Cassandra.
  - Le connecteur Cassandra Spark crée une session pour chaque exécuteur Spark. Considérez cette session comme l'unité d'échelle permettant de déterminer le débit requis et le nombre de connexions requises.
  - Lorsque vous définissez le nombre de cœurs par exécuteur et le nombre de cœurs par tâche, commencez par une valeur faible et augmentez selon les besoins.
  - Définissez les échecs des tâches Spark pour autoriser le traitement en cas d'erreurs transitoires. Une fois que vous vous êtes familiarisé avec les caractéristiques et les exigences de trafic de votre application, nous vous recommandons de définir`spark.task.maxFailures`à une valeur bornée.
  - Par exemple, la configuration suivante peut gérer deux tâches simultanées par exécuteur et par session :

    ```
    spark.executor.instances = configurable -> number of executors for the session.
    spark.executor.cores = 2 -> Number of cores per executor.
    spark.task.cpus = 1 -> Number of cores per task.
    spark.task.maxFailures = -1
    ```

- Désactivez le traitement par lots.
  - Nous vous recommandons de désactiver le traitement par lots pour améliorer les modèles d'accès aléatoire. L'extrait de code suivant en est un exemple.

    ```
    spark.cassandra.output.batch.size.rows = 1 (Default = None)
    spark.cassandra.output.batch.grouping.key = none (Default = Partition)
    spark.cassandra.output.batch.grouping.buffer.size = 100 (Default = 1000)
    ```

- Set**`SPARK_LOCAL_DIRS`**sur un disque local rapide disposant de suffisamment d'espace.
  - Par défaut, Spark enregistre les fichiers de sortie cartographique et les ensembles de données distribués résilients (RDD) dans un`/tmp` dossier. Selon la configuration de votre hôte Spark, cela peut entraîneril ne reste plus d'espace sur l'appareilerreurs de style.

- Pour définir leSPARK_LOCAL_DIRSvariable d'environnement vers un répertoire appelé/example/spark-dir, vous pouvez utiliser la commande suivante.

```
export SPARK_LOCAL_DIRS=/example/spark-dir
```

Étape 3 : Création du fichier de configuration de l'application

Pour utiliser le connecteur open source Spark Cassandra avec Amazon Keyspaces, vous devez fournir un fichier de configuration de l'application contenant les paramètres requis pour vous connecter auDataStaxpilote Java. Vous pouvez utiliser des informations d'identification spécifiques au service ou le plug-in SigV4 pour vous connecter.

Si ce n'est pas déjà fait, vous devez convertir le certificat numérique Starfield en fichier TrustStore. Vous pouvez suivre les étapes détaillées surthe section called “Avant de commencer”à partir du didacticiel de connexion au pilote Java. Prenez note du chemin et du mot de passe du fichier TrustStore, car vous avez besoin de ces informations lorsque vous créez le fichier de configuration de l'application.

Connectez-vous avec l'authentification SigV4

Cette section vous montre un exempleapplication.conffichier que vous pouvez utiliser lors de la connexion avecAWSles informations d'identification et le plugin SigV4. Si ce n'est pas déjà fait, vous devez générer vos clés d'accès IAM (un identifiant de clé d'accès et une clé d'accès secrète) et les enregistrer dans votreAWSfichier de configuration ou en tant que variables d'environnement. Pour obtenir des instructions complètes, veuillez consulter the section called “Informations d'identification requises pour AWS l'authentification”.

Dans l'exemple suivant, remplacez le chemin d'accès à votre fichier TrustStore et remplacez le mot de passe.

```
datastax-java-driver {
        basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
        basic.load-balancing-policy {
            class = DefaultLoadBalancingPolicy
            local-datacenter = us-east-1
            slow-replica-avoidance = false
        }
        basic.request {
              consistency = LOCAL_QUORUM
```

```
        }
        advanced {
                auth-provider = {
                   class = software.aws.mcs.auth.SigV4AuthProvider
                   aws-region = us-east-1
                 }
            ssl-engine-factory {
                class = DefaultSslEngineFactory
                truststore-path = "path_to_file/cassandra_truststore.jks"
                truststore-password = "password"
        hostname-validation=false
            }
   }
        advanced.connection.pool.local.size = 3
}
```

Mettez à jour et enregistrez ce fichier de configuration sous`/home/user1/application.conf`. Les exemples suivants utilisent ce chemin.

Connectez-vous avec des informations d'identification spécifiques au service

Cette section vous montre un exemple`application.conf`fichier que vous pouvez utiliser lors de la connexion avec des informations d'identification spécifiques au service. Si ce n'est pas déjà fait, vous devez générer des informations d'identification spécifiques au service pour Amazon Keyspaces. Pour obtenir des instructions complètes, veuillez consulter the section called “Informations d'identification spécifiques au service”.

Dans l'exemple suivant, remplacez`username`et`password`avec vos propres informations d'identification. Remplacez également le chemin d'accès à votre fichier TrustStore et remplacez le mot de passe.

```
datastax-java-driver {
        basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
        basic.load-balancing-policy {
            class = DefaultLoadBalancingPolicy
            local-datacenter = us-east-1
        }
        basic.request {
              consistency = LOCAL_QUORUM
        }
```

```
        advanced {
            auth-provider = {
            class = PlainTextAuthProvider
                    username = "username"
                    password = "password"
                    aws-region = "us-east-1"
            }
            ssl-engine-factory {
                class = DefaultSslEngineFactory
                truststore-path = "path_to_file/cassandra_truststore.jks"
                truststore-password = "password"
                hostname-validation=false
            }
            metadata = {
                schema {
                     token-map.enabled = true
                }
            }
        }
}
```

Mettez à jour et enregistrez ce fichier de configuration sous/home/user1/application.confà utiliser avec l'exemple de code.

Connectez-vous grâce à un tarif fixe

Pour imposer un taux fixe par exécuteur Spark, vous pouvez définir un limiteur de requêtes. Ce limiteur de requêtes limite le taux de requêtes par seconde. Le connecteur Spark Cassandra déploie une session Cassandra par exécuteur. L'utilisation de la formule suivante peut vous aider à obtenir un débit constant par rapport à une table.

```
max-request-per-second * numberOfExecutors = total throughput against a table
```

Vous pouvez ajouter cet exemple au fichier de configuration de l'application que vous avez créé précédemment.

```
datastax-java-driver {
  advanced.throttler {
    class = RateLimitingRequestThrottler

    max-requests-per-second = 3000
    max-queue-size = 30000
```

```
    drain-interval = 1 millisecond
  }
}
```

Étape 4 : Préparer les données source et la table cible dans Amazon Keyspaces

Au cours de cette étape, vous allez créer un fichier source avec des exemples de données et un tableau Amazon Keyspaces.

1. Créez le fichier source. Vous pouvez choisir l'une des options suivantes :
   - Pour ce didacticiel, vous allez utiliser un fichier de valeurs séparées par des virgules (CSV) portant le nom`keyspaces_sample_table.csv`comme fichier source pour la migration des données. Le fichier d'exemple fourni contient quelques lignes de données pour une table portant le nom`book_awards`.
     - Téléchargez l'exemple de fichier CSV (`keyspaces_sample_table.csv`) qui est contenu dans le fichier d'archive suivantsamplemigration.zip. Décompressez l'archive et notez le chemin vers`keyspaces_sample_table.csv`.
   - Si vous souhaitez utiliser votre propre fichier CSV pour écrire des données dans Amazon Keyspaces, assurez-vous que les données sont aléatoires. Les données qui sont lues directement à partir d'une base de données ou exportées vers des fichiers plats sont généralement classées par partition et par clé primaire. L'importation de données commandées dans Amazon Keyspaces peut entraîner leur écriture sur de plus petits segments des partitions Amazon Keyspaces, ce qui entraîne une répartition inégale du trafic. Cela peut entraîner un ralentissement des performances et des taux d'erreur plus élevés.

     En revanche, la répartition aléatoire des données permet de tirer parti des fonctionnalités d'équilibrage de charge intégrées d'Amazon Keyspaces en répartissant le trafic entre les partitions de manière plus uniforme. Il existe différents outils que vous pouvez utiliser pour la randomisation des données. Pour un exemple utilisant l'outil open sourceChouf, voirthe section called “Étape 2 : Préparation des données”dans le didacticiel sur la migration des données. L'exemple suivant montre comment mélanger des données en tant que`DataFrame`.

     ```
     import org.apache.spark.sql.functions.randval
     ```

```
shuffledDF = dataframe.orderBy(rand())
```

2. Créez l'espace clé et le tableau cibles dans Amazon Keyspaces.

   a. Connectez-vous à Amazon Keyspaces à l'aide `decqlsh`, puis remplacez le point de terminaison du service, le nom d'utilisateur et le mot de passe de l'exemple suivant par vos propres valeurs.

   ```
   cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
   p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
   ```

   b. Créez un nouvel espace de touches avec le nom`catalog`comme indiqué dans l'exemple suivant.

   ```
   CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
   ```

   c. Une fois que le nouvel espace de touches a atteint l'état Disponible, utilisez le code suivant pour créer la table cible`book_awards`. Pour en savoir plus sur la création de ressources asynchrones et sur la façon de vérifier si une ressource est disponible, voirthe section called “Création d'espaces clés”.

   ```
   CREATE TABLE catalog.book_awards (
      year int,
      award text,
      rank int,
      category text,
      book_title text,
      author text,
      publisher text,
      PRIMARY KEY ((year, award), category, rank)
      );
   ```

Étape 5 : Écrire et lire les données Amazon Keyspaces à l'aide du connecteur Apache Cassandra Spark

Au cours de cette étape, vous commencez par charger les données du fichier d'exemple dans un`DataFrame`avec le connecteur Spark Cassandra. Ensuite, vous écrivez les données à partir du`DataFrame`dans votre tableau Amazon Keyspaces. Vous pouvez également utiliser cette partie indépendamment, par exemple pour migrer des données vers une table Amazon Keyspaces. Enfin,

vous lisez les données de votre tableau dans un`DataFrame`à l'aide du connecteur Spark Cassandra. Vous pouvez également utiliser cette partie indépendamment, par exemple pour lire les données d'une table Amazon Keyspaces afin d'effectuer des analyses de données avec Apache Spark.

1. Démarrez le Spark Shell comme indiqué dans l'exemple suivant. Notez que cet exemple utilise l'authentification SigV4.

```
./spark-shell --files application.conf --conf
 spark.cassandra.connection.config.profile.path=application.conf
 --packages software.aws.mcs:aws-sigv4-auth-cassandra-java-driver-
plugin:4.0.5,com.datastax.spark:spark-cassandra-connector_2.12:3.1.0 --conf
 spark.sql.extensions=com.datastax.spark.connector.CassandraSparkExtensions
```

2. Importez le connecteur Spark Cassandra à l'aide du code suivant.

```
import org.apache.spark.sql.cassandra._
```

3. Pour lire les données du fichier CSV et les stocker dans un`DataFrame`, vous pouvez utiliser l'exemple de code suivant.

```
var df =
 spark.read.option("header","true").option("inferSchema","true").csv("keyspaces_sample_tabl
```

Vous pouvez afficher le résultat à l'aide de la commande suivante.

```
scala> df.show();
```

La sortie doit ressembler à ceci.

```
+----------------+----+-----------+----+------------------+--------------------
+-------------+
|           award|year|   category|rank|            author|          book_title|
 publisher|
+----------------+----+-----------+----+------------------+--------------------
+-------------+
|Kwesi Manu Prize|2020|    Fiction|   1|        Akua Mansa|   Where did you go?|
SomePublisher|
|Kwesi Manu Prize|2020|    Fiction|   2|       John Stiles|           Yesterday|
Example Books|
|Kwesi Manu Prize|2020|    Fiction|   3|        Nikki Wolf|Moving to the Cha...|
 AnyPublisher|
```

```
|            Wolf|2020|Non-Fiction|   1|       Wang Xiulan|    History of Ideas|
Example Books|
|            Wolf|2020|Non-Fiction|   2|Ana Carolina Silva|       Science Today|
SomePublisher|
|            Wolf|2020|Non-Fiction|   3| Shirley Rodriguez|The Future of Sea...|
 AnyPublisher|
|     Richard Roe|2020|    Fiction|   1| Alejandro Rosalez|         Long Summer|
SomePublisher|
|     Richard Roe|2020|    Fiction|   2|       Arnav Desai|             The Key|
Example Books|
|     Richard Roe|2020|    Fiction|   3|     Mateo Jackson|    Inside the Whale|
 AnyPublisher|
+----------------+----+-----------+----+------------------+--------------------
+-------------+
```

Vous pouvez confirmer le schéma des données dans le`DataFrame`comme indiqué dans l'exemple suivant.

```
scala> df.printSchema
```

La sortie doit ressembler à ceci.

```
root
|-- award: string (nullable = true)
|-- year: integer (nullable = true)
|-- category: string (nullable = true)
|-- rank: integer (nullable = true)
|-- author: string (nullable = true)
|-- book_title: string (nullable = true)
|-- publisher: string (nullable = true)
```

4. Utilisez la commande suivante pour écrire les données dans le`DataFrame`au tableau Amazon Keyspaces.

```
df.write.cassandraFormat("book_awards", "catalog").mode("APPEND").save()
```

5. Pour confirmer que les données ont été enregistrées, vous pouvez les relire dans une base de données, comme illustré dans l'exemple suivant.

```
var newDf = spark.read.cassandraFormat("book_awards", "catalog").load()
```

Vous pouvez ensuite afficher les données qui se trouvent désormais dans le cadre de données.

```
scala> newDf.show()
```

La sortie de cette commande doit ressembler à ceci.

```
+--------------------+-----------------+---------------+----------+-------------
+----+----+
|          book_title|           author|          award|  category|
 publisher|rank|year|
+--------------------+-----------------+---------------+----------+-------------
+----+----+
|         Long Summer| Alejandro Rosalez|     Richard Roe|    Fiction|
SomePublisher|   1|2020|
|    History of Ideas|       Wang Xiulan|           Wolf|Non-Fiction|Example
 Books|   1|2020|
|   Where did you go?|        Akua Mansa|Kwesi Manu Prize|    Fiction|
SomePublisher|   1|2020|
|    Inside the Whale|     Mateo Jackson|     Richard Roe|    Fiction|
 AnyPublisher|   3|2020|
|           Yesterday|       John Stiles|Kwesi Manu Prize|    Fiction|Example
 Books|   2|2020|
|Moving to the Cha...|        Nikki Wolf|Kwesi Manu Prize|    Fiction|
 AnyPublisher|   3|2020|
|The Future of Sea...| Shirley Rodriguez|           Wolf|Non-Fiction|
 AnyPublisher|   3|2020|
|       Science Today|Ana Carolina Silva|           Wolf|Non-Fiction|
SomePublisher|   2|2020|
|             The Key|       Arnav Desai|     Richard Roe|    Fiction|Example
 Books|   2|2020|
+--------------------+-----------------+---------------+----------+-------------
+----+----+
```

Résolution des erreurs courantes lors de l'utilisation du connecteur Spark Cassandra avec Amazon Keyspaces

Si vous utilisez Amazon Virtual Private Cloud et que vous vous connectez à Amazon Keyspaces, les erreurs les plus courantes lors de l'utilisation du connecteur Spark sont dues aux problèmes de configuration suivants.

- L'utilisateur ou le rôle IAM utilisé dans le VPC ne dispose pas des autorisations requises pour accéder au `system.peers` tableau dans Amazon Keyspaces. Pour plus d'informations, veuillez consulter the section called “Remplissage des entrées de `system.peers` table avec les informations de point de terminaison VPC de l'interface”.
- L'utilisateur ou le rôle IAM ne dispose pas des autorisations de lecture/écriture requises pour la table des utilisateurs ni pour l'accès en lecture aux tables système dans Amazon Keyspaces. Pour plus d'informations, veuillez consulter the section called “Étape 1 : Configuration d'Amazon Keyspaces”.
- La configuration du pilote Java ne désactive pas la vérification du nom d'hôte lors de la création de la connexion SSL/TLS. Pour obtenir des exemples, consultez the section called “Étape 2 : Configuration du pilote”.

Pour connaître les étapes détaillées de résolution des problèmes de connexion, voirthe section called “Erreurs de connexion au point de terminaison VPC”.

De plus, vous pouvez utiliser AmazonCloudWatchdes métriques pour vous aider à résoudre les problèmes liés à la configuration de votre connecteur Spark Cassandra dans Amazon Keyspaces. Pour en savoir plus sur l'utilisation d'Amazon Keyspaces avecCloudWatch, voirthe section called “Surveillance avec CloudWatch”.

La section suivante décrit les mesures les plus utiles à observer lorsque vous utilisez le connecteur Spark Cassandra.

PerConnectionRequestRateExceeded

Amazon Keyspaces dispose d'un quota de 3 000 requêtes par seconde et par connexion. Chaque exécuteur Spark établit une connexion avec Amazon Keyspaces. L'exécution de plusieurs tentatives peut épuiser votre quota de débit de demandes par connexion. Si vous dépassez ce quota, Amazon Keyspaces émet un`PerConnectionRequestRateExceeded`entrée métriqueCloudWatch.

Si tu voisPerConnectionRequestRateExceededEn cas d'événements associés à d'autres erreurs système ou utilisateur, il est probable que Spark exécute plusieurs tentatives au-delà du nombre de requêtes alloué par connexion.

Si tu vois`PerConnectionRequestRateExceeded`sans autres erreurs, vous devrez peut-être augmenter le nombre de connexions dans les paramètres de votre pilote pour augmenter le débit, ou vous devrez peut-être augmenter le nombre d'exécuteurs dans votre tâche Spark.

StoragePartitionThroughputCapacityExceeded

Amazon Keyspaces dispose d'un quota de 1 000 WCU ou WRU par seconde/3 000 RCU ou RRU par seconde, par partition. Si vous voyez`StoragePartitionThroughputCapacityExceeded` CloudWatchévénements, cela peut indiquer que les données ne sont pas randomisées lors du chargement. Pour obtenir des exemples de brassage de données, voirthe section called “Étape 4 : Préparer les données source et la table cible”.

Erreurs et avertissements courants

Si vous utilisez Amazon Virtual Private Cloud et que vous vous connectez à Amazon Keyspaces, le pilote Cassandra peut émettre un message d'avertissement concernant le nœud de contrôle lui-même dans le`system.peers`table. Pour plus d'informations, veuillez consulter the section called “Erreurs et avertissements courants”. Vous pouvez ignorer cet avertissement en toute sécurité.

Résolution des problèmes liés à Amazon Keyspaces (pour Apache Cassandra)

Les sections suivantes fournissent des informations sur la manière de résoudre les problèmes de configuration courants que vous pouvez rencontrer lors de l'utilisation d'Amazon Keyspaces (pour Apache Cassandra).

Pour obtenir des conseils de résolution des problèmes spécifiques à l'accès IAM, consultezthe section called “Résolution des problèmes”.

Pour plus d'informations sur les meilleures pratiques en matière de sécurité, consultezthe section called “Bonnes pratiques de sécurité”.

Rubriques

Résolution des erreurs générales dans Amazon Keyspaces

Vous recevez des erreurs générales ? Voici quelques problèmes courants et comment les résoudre.

Erreurs générales

Vous êtes confronté à l'une des exceptions de premier niveau suivantes, qui peut se produire pour de nombreuses raisons différentes.

- `NoNodeAvailableException`
- `NoHostAvailableException`
- `AllNodesFailedException`

Ces exceptions sont générées par le pilote client et peuvent se produire soit lorsque vous établissez la connexion de contrôle, soit lorsque vous effectuez des demandes de lecture/écriture/préparation/exécution/batch.

Lorsque l'erreur se produit alors que vous établissez la connexion de contrôle, cela signifie que tous les points de contact spécifiés dans votre application sont inaccessibles. Lorsque l'erreur se produit lors de l'exécution de requêtes de lecture/écriture/préparation/exécution, cela indique que toutes les tentatives pour cette demande ont été épuisées. Chaque nouvelle tentative est tentée sur un nœud différent lorsque vous utilisez la politique de nouvelle tentative par défaut.

Comment isoler l'erreur sous-jacente des exceptions du pilote Java de haut niveau

Ces erreurs générales peuvent être causées soit par des problèmes de connexion, soit lors d'opérations de lecture/écriture/préparation/exécution. Des défaillances transitoires sont prévisibles dans les systèmes distribués et doivent être traitées en réessayant la demande. Le pilote Java ne réessaie pas automatiquement en cas d'erreur de connexion. Il est donc recommandé de mettre en œuvre la politique de nouvelle tentative lors de l'établissement de la connexion du pilote dans votre application. Pour un aperçu détaillé des meilleures pratiques en matière de connexion, voirthe section called “Connexions”.

Par défaut, le pilote Java définit la valeur false `idempotence` pour toutes les demandes, ce qui signifie qu'il ne réessaie pas automatiquement les demandes de lecture/écriture/préparation qui ont échoué. `idempotence`Pour configurer `true` et demander au pilote de réessayer les demandes ayant échoué, vous pouvez le faire de différentes manières. Voici un exemple de la façon dont vous pouvez définir l'idempotence par programmation pour une seule requête dans votre application Java.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");
s.setIdempotent(true);
```

Vous pouvez également définir l'idempotence par défaut pour l'ensemble de votre application Java par programmation, comme indiqué dans l'exemple suivant.

```
// Make all statements idempotent by default:
cluster.getConfiguration().getQueryOptions().setDefaultIdempotence(true);
//Set the default idempotency to true in your Cassandra configuration
basic.request.default-idempotence = true
```

Une autre recommandation consiste à créer une politique de nouvelle tentative au niveau de l'application. Dans ce cas, l'application doit intercepter la demande `NoNodeAvailableException` et réessayer. Nous recommandons 10 tentatives avec un retard exponentiel commençant à 10 ms et allant jusqu'à 100 ms avec un temps total de 1 seconde pour toutes les tentatives.

Une autre option consiste à appliquer la politique de nouvelles tentatives exponentielles d'Amazon Keyspaces lors de l'établissement de la connexion au pilote Java disponible sur Github.

Vérifiez que vous avez établi des connexions à plusieurs nœuds lorsque vous utilisez la politique de nouvelle tentative par défaut. Vous pouvez le faire à l'aide de la requête suivante dans Amazon Keyspaces.

```
SELECT * FROM system.peers;
```

Si la réponse à cette requête est vide, cela indique que vous travaillez avec un seul nœud pour Amazon Keyspaces. Si vous utilisez la politique de nouvelle tentative par défaut, il n'y aura aucune nouvelle tentative car la nouvelle tentative par défaut a toujours lieu sur un autre nœud. Pour en savoir plus sur l'établissement de connexions via des points de terminaison VPC, consultez. the section called “Connexions aux points de terminaison VPC”

Pour un step-by-step didacticiel expliquant comment établir une connexion à Amazon Keyspaces à l'aide du pilote Cassandra Datastax 4.x, consultez. the section called “Plugin d'authentification pour Java 4.x”

Résolution des problèmes de connexion dans Amazon Keyspaces

Vous rencontrez des difficultés pour vous connecter ? Voici quelques problèmes courants et comment les résoudre.

Erreurs de connexion à un point de terminaison Amazon Keyspaces

Les connexions échouées et les erreurs de connexion peuvent entraîner des messages d'erreur différents. La section suivante couvre les scénarios les plus courants.

Rubriques

- Je n'arrive pas à me connecter à Amazon Keyspaces avec cqlsh
- Je ne parviens pas à me connecter à Amazon Keyspaces à l'aide d'un pilote client Cassandra

Je n'arrive pas à me connecter à Amazon Keyspaces avec cqlsh

Vous essayez de vous connecter à un point de terminaison Amazon Keyspaces à l'aide de cqlsh et la connexion échoue avec un. **`Connection error`**

Si vous essayez de vous connecter à une table Amazon Keyspaces et que cqlsh n'est pas correctement configuré, la connexion échoue. La section suivante fournit des exemples des problèmes de configuration les plus courants qui entraînent des erreurs de connexion lorsque vous essayez d'établir une connexion à l'aide de cqlsh.

> **Note**
>
> Si vous essayez de vous connecter à Amazon Keyspaces depuis un VPC, des autorisations supplémentaires sont requises. Pour configurer correctement une connexion à l'aide de points de terminaison VPC, suivez les étapes décrites dans le. the section called “Connexion aux points de terminaison VPC”

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais une erreur de connexion s'affiche. **`timed out`**

Cela peut être le cas si vous n'avez pas fourni le port correct, ce qui entraîne l'erreur suivante.

```
#  cqlsh cassandra.us-east-1.amazonaws.com 9140 -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.199': error(None,
 "Tried connecting to [('3.234.248.199', 9140)]. Last error: timed out")})
```

Pour résoudre ce problème, vérifiez que vous utilisez le port 9142 pour la connexion.

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`Name or service not known`**

Cela peut être le cas si vous avez utilisé un point de terminaison mal orthographié ou qui n'existe pas. Dans l'exemple suivant, le nom du point de terminaison est mal orthographié.

```
#  cqlsh cassandra.us-east-1.amazon.com 9142 -u "USERNAME" -p "PASSWORD" --ssl
Traceback (most recent call last):
  File "/usr/bin/cqlsh.py", line 2458, in >module>
    main(*read_options(sys.argv[1:], os.environ))
  File "/usr/bin/cqlsh.py", line 2436, in main
    encoding=options.encoding)
  File "/usr/bin/cqlsh.py", line 484, in __init__
    load_balancing_policy=WhiteListRoundRobinPolicy([self.hostname]),
  File "/usr/share/cassandra/lib/cassandra-driver-internal-only-3.11.0-bb96859b.zip/
cassandra-driver-3.11.0-bb96859b/cassandra/policies.py", line 417, in __init__
socket.gaierror: [Errno -2] Name or service not known
```

Pour résoudre ce problème lorsque vous utilisez des points de terminaison publics pour vous connecter, sélectionnez un point de the section called “Points de terminaison de service” terminaison disponible et vérifiez que le nom du point de terminaison ne contient aucune erreur. Si vous utilisez des points de terminaison VPC pour vous connecter, vérifiez que les informations du point de terminaison VPC sont correctes dans votre configuration cqlsh.

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`OperationTimedOut`**

Amazon Keyspaces exige que le protocole SSL soit activé pour les connexions afin de garantir une sécurité renforcée. Le paramètre SSL est peut-être absent si le message d'erreur suivant s'affiche.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD"
Connection error: ('Unable to connect to any servers', {'3.234.248.192':
 OperationTimedOut('errors=Timed out creating connection (5 seconds),
 last_host=None',)})
#
```

Pour résoudre ce problème, ajoutez l'indicateur suivant à la commande de connexion cqlsh.

```
--ssl
```

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh et vous recevez un message d'erreur. **`SSL transport factory requires a valid certfile to be specified`**

Dans ce cas, le chemin d'accès au certificat SSL/TLS est manquant, ce qui entraîne l'erreur suivante.

```
# cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory
#


# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Validation is enabled; SSL transport factory requires a valid certfile to be specified.
 Please provide path to the certfile in [ssl] section as 'certfile' option in /
root/.cassandra/cqlshrc (or use [certfiles] section) or set SSL_CERTFILE environment
 variable.
```

```
#
```

Pour résoudre ce problème, ajoutez le chemin du fichier de certificat sur votre ordinateur.

```
certfile =  path_to_file/sf-class2-root.crt
```

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`No such file or directory`**

Cela peut être le cas si le chemin d'accès au fichier de certificat sur votre ordinateur est incorrect, ce qui entraîne l'erreur suivante.

```
# cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile = /root/wrong_path/sf-class2-root.crt
#



# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.192': IOError(2, 'No
 such file or directory')})
#
```

Pour résoudre ce problème, vérifiez que le chemin d'accès au fichier de certificat sur votre ordinateur est correct.

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`[X509] PEM lib`**

Cela peut être le cas si le fichier de certificat SSL/TLS n'`sf-class2-root.crt`est pas valide, ce qui entraîne l'erreur suivante.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.241':
 error(185090057, u"Tried connecting to [('3.234.248.241', 9142)]. Last error: [X509]
 PEM lib (_ssl.c:3063)")})
```

```
#
```

Pour résoudre ce problème, téléchargez le certificat numérique Starfield à l'aide de la commande suivante. Enregistrez `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais vous recevez une erreur SSL. **unknown**

Cela peut être le cas si le fichier de certificat SSL/TLS `sf-class2-root.crt` est vide, ce qui entraîne l'erreur suivante.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.220': error(0,
 u"Tried connecting to [('3.234.248.220', 9142)]. Last error: unknown error
 (_ssl.c:3063)")})
#
```

Pour résoudre ce problème, téléchargez le certificat numérique Starfield à l'aide de la commande suivante. Enregistrez `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`SSL: CERTIFICATE_VERIFY_FAILED`**

Cela peut être le cas si le fichier de certificat SSL/TLS n'a pas pu être vérifié, ce qui entraîne l'erreur suivante.

```
Connection error: ('Unable to connect to any servers', {'3.234.248.223':
 error(1, u"Tried connecting to [('3.234.248.223', 9142)]. Last error: [SSL:
 CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:727)")})
```

Pour résoudre ce problème, téléchargez à nouveau le fichier de certificat à l'aide de la commande suivante. Enregistrez `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`Last error: timed out`**

Cela peut être le cas si vous n'avez pas configuré de règle sortante pour Amazon Keyspaces dans votre groupe de sécurité Amazon EC2, ce qui entraîne l'erreur suivante.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.206': error(None,
 "Tried connecting to [('3.234.248.206', 9142)]. Last error: timed out")})
#
```

Pour confirmer que ce problème est dû à la configuration de l'instance Amazon EC2 et non`cqlsh`, vous pouvez essayer de vous connecter à votre espace de touches en utilisant, par exemple AWS CLI, la commande suivante.

```
aws keyspaces list-tables --keyspace-name 'my_keyspace'
```

Si cette commande expire également, l'instance Amazon EC2 n'est pas correctement configurée.

Pour vérifier que vous disposez des autorisations suffisantes pour accéder à Amazon Keyspaces, vous pouvez utiliser le AWS CloudShell pour vous connecter à. `cqlsh` Si ces connexions sont établies, vous devez configurer l'instance Amazon EC2.

Pour résoudre ce problème, vérifiez que votre instance Amazon EC2 dispose d'une règle sortante qui autorise le trafic vers Amazon Keyspaces. Si ce n'est pas le cas, vous devez créer un nouveau groupe de sécurité pour l'instance EC2 et ajouter une règle autorisant le trafic sortant vers les ressources Amazon Keyspaces. Pour mettre à jour la règle de trafic sortant afin d'autoriser le trafic vers Amazon Keyspaces, choisissez CQLSH/CASSANDRA dans le menu déroulant Type.

Après avoir créé le nouveau groupe de sécurité avec la règle de trafic sortant, vous devez l'ajouter à l'instance. Sélectionnez l'instance, puis sélectionnez Actions, Sécurité, puis Modifier les groupes de sécurité. Ajoutez le nouveau groupe de sécurité avec la règle sortante, mais assurez-vous que le groupe par défaut reste également disponible.

Pour plus d'informations sur la façon d'afficher et de modifier les règles de sortie EC2, consultez la section Ajouter des règles à un groupe de sécurité dans le guide de l'utilisateur Amazon EC2.

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`Unauthorized`**

Cela peut être le cas si vous ne disposez pas des autorisations Amazon Keyspaces dans la politique utilisateur d'IAM, ce qui entraîne l'erreur suivante.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "testuser-at-12345678910" -p
 "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.241':
 AuthenticationFailed('Failed to authenticate to 3.234.248.241: Error from server:
 code=2100 [Unauthorized] message="User arn:aws:iam::12345678910:user/testuser has no
 permissions."',)})
#
```

Pour résoudre ce problème, assurez-vous que l'utilisateur IAM `testuser-at-12345678910` est autorisé à accéder à Amazon Keyspaces. Pour des exemples de politiques IAM qui accordent l'accès à Amazon Keyspaces, consultez. the section called “Exemples de politiques basées sur l’identité”

Pour obtenir des conseils de résolution des problèmes spécifiques à l'accès IAM, consultezthe section called “Résolution des problèmes”.

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de cqlsh, mais un message d'erreur s'affiche. **`Bad credentials`**

Cela peut être le cas si le nom d'utilisateur ou le mot de passe est incorrect, ce qui entraîne l'erreur suivante.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.248':
 AuthenticationFailed('Failed to authenticate to 3.234.248.248: Error from server:
 code=0100 [Bad credentials] message="Provided username USERNAME and/or password are
 incorrect"',)})
#
```

Pour résoudre ce problème, vérifiez que le *NOM D'UTILISATEUR* et le MOT DE *PASSE de* votre code correspondent au nom d'utilisateur et au mot de passe que vous avez obtenus lorsque vous avez généré les informations d'identification spécifiques au service.

> **Important**
>
> Si vous continuez à voir des erreurs lorsque vous essayez de vous connecter à cqlsh, réexécutez la commande avec l'`--debug`option et incluez le résultat détaillé lors de la prise de contact. AWS Support

Je ne parviens pas à me connecter à Amazon Keyspaces à l'aide d'un pilote client Cassandra

Les sections suivantes présentent les erreurs les plus courantes lors de la connexion avec un pilote client Cassandra.

Vous essayez de vous connecter à une table Amazon Keyspaces à l'aide du pilote DataStax Java, mais un message d'**`NodeUnavailableException`**erreur s'affiche.

Si la connexion sur laquelle la demande est tentée est interrompue, cela entraîne l'erreur suivante.

```
[com.datastax.oss.driver.api.core.NodeUnavailableException: No connection
 was available to Node(endPoint=vpce-22ff22f2f22222fff-aa1bb234.cassandra.us-
west-2.vpce.amazonaws.com/11.1.1111.222:9142, hostId=1a23456b-
c77d-8888-9d99-146cb22d6ef6, hashCode=123ca4567)]
```

Pour résoudre ce problème, trouvez la valeur du rythme cardiaque et abaissez-la à 30 secondes si elle est supérieure.

```
advanced.heartbeat.interval = 30 seconds
```

Recherchez ensuite le délai d'expiration associé et assurez-vous que la valeur est définie sur au moins 5 secondes.

```
advanced.connection.init-query-timeout = 5 seconds
```

Vous essayez de vous connecter à une table Amazon Keyspaces à l'aide d'un pilote et du plugin SigV4, mais un message d'erreur s'affiche. **`AttributeError`**

Si les informations d'identification ne sont pas correctement configurées, l'erreur suivante se produit.

```
cassandra.cluster.NoHostAvailable: (‘Unable to connect to any servers’,
 {‘44.234.22.154:9142’: AttributeError(“‘NoneType’ object has no attribute
 ‘access_key’“)})
```

Pour résoudre ce problème, vérifiez que vous transmettez les informations d'identification associées à votre utilisateur ou rôle IAM lorsque vous utilisez le plug-in SigV4. Le plug-in SigV4 nécessite les informations d'identification suivantes.

- `AWS_ACCESS_KEY_ID`— Spécifie une clé d' AWS accès associée à un utilisateur ou à un rôle IAM.
- `AWS_SECRET_ACCESS_KEY`— Spécifie la clé secrète associée à la clé d'accès. Il s'agit du « mot de passe » de la clé d'accès.

Pour en savoir plus sur les clés d'accès et le plug-in SigV4, consultezthe section called “Informations d'identification IAM pour l' AWS authentification”.

Vous essayez de vous connecter à une table Amazon Keyspaces à l'aide d'un pilote, mais un message d'erreur s'affiche. **`PartialCredentialsError`**

Si ce `AWS_SECRET_ACCESS_KEY` n'est pas le cas, cela peut entraîner l'erreur suivante.

```
cassandra.cluster.NoHostAvailable: (‘Unable to connect to any servers’,
 {‘44.234.22.153:9142’:
 PartialCredentialsError(‘Partial credentials found in config-file, missing:
 aws_secret_access_key’)})
```

Pour résoudre ce problème, vérifiez que vous passez à la fois le `AWS_ACCESS_KEY_ID` et le `AWS_SECRET_ACCESS_KEY` lorsque vous utilisez le plug-in SigV4. Pour en savoir plus sur les clés d'accès et le plug-in SigV4, consultezthe section called “Informations d'identification IAM pour l' AWS authentification”.

Vous essayez de vous connecter à une table Amazon Keyspaces à l'aide d'un pilote, mais un message d'**`Invalid signature`**erreur s'affiche.

Cela peut être le cas si vous avez utilisé des informations d'identification incorrectes, ce qui entraîne l'erreur suivante.

```
cassandra.cluster.NoHostAvailable: (‘Unable to connect to any servers’,
 {‘44.234.22.134:9142’:
 AuthenticationFailed(‘Failed to authenticate to 44.234.22.134:9142: Error from server:
 code=0100
 [Bad credentials] message=“Authentication failure: Invalid signature”’)})
```

Pour résoudre ce problème, vérifiez que les informations d'identification que vous transmettez sont associées à l'utilisateur ou au rôle IAM que vous avez configuré pour accéder à Amazon Keyspaces. Pour en savoir plus sur les clés d'accès et le plug-in SigV4, consultezthe section called “Informations d'identification IAM pour l' AWS authentification”.

La connexion de mon point de terminaison VPC ne fonctionne pas correctement

Vous essayez de vous connecter à Amazon Keyspaces via des points de terminaison VPC, mais vous recevez des erreurs de mappage de jetons ou vous rencontrez un faible débit.

Cela peut être le cas si la connexion du point de terminaison VPC n'est pas correctement configurée.

Pour résoudre ces problèmes, vérifiez les détails de configuration suivants. Pour suivre un step-by-step didacticiel expliquant comment configurer une connexion via des points de terminaison VPC d'interface pour Amazon Keyspaces, consultez. the section called “Connexion aux points de terminaison VPC”

1. Vérifiez que l'entité IAM utilisée pour se connecter à Amazon Keyspaces dispose d'un accès en lecture/écriture à la table utilisateur et d'un accès en lecture aux tables système, comme indiqué dans l'exemple suivant.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

2. Vérifiez que l'entité IAM utilisée pour se connecter à Amazon Keyspaces dispose des autorisations de lecture requises pour accéder aux informations du point de terminaison VPC sur votre instance Amazon EC2, comme indiqué dans l'exemple suivant.

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

```
            "Sid":"ListVPCEndpoints",
            "Effect":"Allow",
            "Action":[
               "ec2:DescribeNetworkInterfaces",
               "ec2:DescribeVpcEndpoints"
            ],
            "Resource":"*"
         }
      ]
}
```

> **Note**
>
> Les politiques gérées `AmazonKeyspacesFullAccess` incluent `AmazonKeyspacesReadOnlyAccess_v2` les autorisations requises pour permettre à Amazon Keyspaces d'accéder à l'instance Amazon EC2 afin de lire les informations sur les points de terminaison VPC d'interface disponibles.

Pour plus d'informations sur les points de terminaison VPC, voir the section called “Utilisation des points de terminaison VPC de l'interface pour Amazon Keyspaces”

3. Vérifiez que la configuration SSL du pilote Java définit la validation du nom d'hôte sur false, comme indiqué dans cet exemple.

```
hostname-validation = false
```

Pour plus d'informations sur la configuration du pilote, consultezthe section called “Étape 2 : Configuration du pilote”.

4. Pour confirmer que le point de terminaison du VPC a été correctement configuré, vous pouvez exécuter l'instruction suivante depuis votre VPC.

> **Note**
>
> Vous ne pouvez pas utiliser votre environnement de développement local ou l'éditeur CQL d'Amazon Keyspaces pour confirmer cette configuration, car ils utilisent le point de terminaison public.

```
SELECT peer FROM system.peers;
```

La sortie doit ressembler à cet exemple et renvoyer entre 2 et 6 nœuds avec des adresses IP privées, en fonction de la configuration et AWS de la région de votre VPC.

```
peer
---------------
 192.0.2.0.15
 192.0.2.0.24
 192.0.2.0.13
 192.0.2.0.7
 192.0.2.0.8

(5 rows)
```

Je ne parviens pas à me connecter en utilisant **cassandra-stress**

Vous essayez de vous connecter à Amazon Keyspaces à l'aide de la **cassandra-stress** commande, mais un **SSL context** message d'erreur s'affiche.

Cela se produit si vous essayez de vous connecter à Amazon Keyspaces, mais que le TrustStore n'est pas correctement configuré. Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients.

Dans ce cas, le message d'erreur suivant s'affiche.

```
Error creating the initializing the SSL Context
```

Pour résoudre ce problème, suivez les instructions pour configurer un TrustStore comme indiqué dans cette rubriquethe section called “Avant de commencer”.

Une fois le TrustStore configuré, vous devriez pouvoir vous connecter à l'aide de la commande suivante.

```
./cassandra-stress user profile=./profile.yaml n=100 "ops(insert=1,select=1)"
 cl=LOCAL_QUORUM -node "cassandra.eu-north-1.amazonaws.com" -port native=9142
 -transport ssl-alg="PKIX" truststore="./cassandra_truststore.jks" truststore-
password="trustStore_pw" -mode native cql3 user="user_name" password="password"
```

Je ne parviens pas à me connecter à l'aide des identités IAM

Vous essayez de vous connecter à une table Amazon Keyspaces en utilisant une identité IAM, mais vous recevez un message d'erreur. **`Unauthorized`**

Cela se produit si vous essayez de vous connecter à une table Amazon Keyspaces en utilisant une identité IAM (par exemple, un utilisateur IAM) sans mettre en œuvre la politique et sans donner au préalable à l'utilisateur les autorisations requises.

Dans ce cas, le message d'erreur suivant s'affiche.

```
Connection error: ('Unable to connect to any servers', {'3.234.248.202':
 AuthenticationFailed('Failed to authenticate to 3.234.248.202:
Error from server: code=2100 [Unauthorized] message="User
 arn:aws:iam::1234567890123:user/testuser has no permissions."',)})
```

Pour résoudre ce problème, vérifiez les autorisations de l'utilisateur IAM. Pour se connecter à un pilote standard, l'utilisateur doit au moins avoir `SELECT` accès aux tables système, car la plupart des pilotes lisent les espaces-clés/tables du système lorsqu'ils établissent la connexion.

Par exemple, les politiques IAM qui accordent l'accès au système Amazon Keyspaces et aux tables utilisateur, voir. the section called “Accès aux tables Amazon Keyspaces”

Pour consulter la section de résolution des problèmes spécifique à IAM, consultezthe section called “Résolution des problèmes”.

J'essaie d'importer des données avec cqlsh et la connexion à ma table Amazon Keyspaces est perdue

Vous essayez de télécharger des données sur Amazon Keyspaces avec cqlsh, mais vous recevez des erreurs de connexion.

La connexion à Amazon Keyspaces échoue lorsque le client cqlsh a reçu trois erreurs consécutives de tout type de la part du serveur. Le client cqlsh échoue avec le message suivant.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Pour résoudre cette erreur, vous devez vous assurer que les données à importer correspondent au schéma de table dans Amazon Keyspaces. Vérifiez le fichier d'importation pour détecter les erreurs d'analyse. Vous pouvez essayer d'utiliser une seule ligne de données en utilisant une instruction INSERT pour isoler l'erreur.

Le client tente automatiquement de rétablir la connexion.

Résolution des problèmes liés à la gestion des capacités dans Amazon Keyspaces

Vous rencontrez des problèmes avec la capacité sans serveur ? Voici quelques problèmes courants et comment les résoudre.

Erreurs de capacité sans serveur

Cette section explique comment reconnaître les erreurs liées à la gestion des capacités sans serveur et comment les résoudre. Par exemple, vous pouvez observer des événements de capacité insuffisante lorsque votre application dépasse la capacité de débit allouée.

Apache Cassandra étant un logiciel basé sur des clusters conçu pour fonctionner sur une flotte de nœuds, il ne contient aucun message d'exception lié aux fonctionnalités sans serveur telles que la capacité de débit. La plupart des pilotes ne comprennent que les codes d'erreur disponibles dans Apache Cassandra. Amazon Keyspaces utilise donc le même ensemble de codes d'erreur pour garantir la compatibilité.

Pour associer les erreurs de Cassandra aux événements de capacité sous-jacents, vous pouvez utiliser Amazon CloudWatch pour surveiller les indicateurs Amazon Keyspaces pertinents. Les événements d'insuffisance de capacité qui entraînent des erreurs côté client peuvent être classés dans les trois groupes suivants en fonction de la ressource à l'origine de l'événement :

- Tableau — Si vous choisissez le mode Capacité allouée pour une table et que votre application dépasse le débit alloué, vous pouvez observer des erreurs de capacité insuffisante. Pour plus d'informations, consultez the section called “Modes de capacité de lecture/écriture”.
- Partition — Vous pouvez rencontrer des événements de capacité insuffisante si le trafic sur une partition donnée dépasse 3 000 RCU ou 1 000 WCU. Nous recommandons de répartir le trafic de manière uniforme sur les partitions comme meilleure pratique. Pour plus d'informations, consultez the section called “Modélisation des données”.
- Connexion : le débit risque d'être insuffisant si vous dépassez le quota du nombre maximal d'opérations par seconde et par connexion. Pour augmenter le débit, vous pouvez augmenter le nombre de connexions par défaut lors de la configuration de la connexion avec le pilote.

  Pour savoir comment configurer les connexions pour Amazon Keyspaces, consultez. the section called “Comment configurer les connexions” Pour plus d'informations sur l'optimisation des

connexions via les points de terminaison VPC, consultez. the section called “Connexions aux points de terminaison VPC”

Pour déterminer quelle ressource est à l'origine de l'événement de capacité insuffisante qui renvoie l'erreur côté client, vous pouvez consulter le tableau de bord dans la console Amazon Keyspaces. Par défaut, la console fournit une vue agrégée des CloudWatch mesures relatives à la capacité et au trafic les plus courantes dans la section Capacité et mesures associées de l'onglet Capacité du tableau.

Pour créer votre propre tableau de bord à l'aide d'Amazon CloudWatch, consultez les statistiques Amazon Keyspaces suivantes.

- `PerConnectionRequestRateExceeded`— Demandes adressées à Amazon Keyspaces qui dépassent le quota du taux de demandes par connexion. Chaque connexion client à Amazon Keyspaces peut prendre en charge jusqu'à 3 000 demandes CQL par seconde. Vous pouvez effectuer plus de 3 000 demandes par seconde en créant plusieurs connexions.
- `ReadThrottleEvents`— Demandes adressées à Amazon Keyspaces qui dépassent la capacité de lecture d'une table.
- `StoragePartitionThroughputCapacityExceeded`— Demandes adressées à une partition de stockage Amazon Keyspaces qui dépassent la capacité de débit de la partition. Les partitions de stockage Amazon Keyspaces peuvent prendre en charge jusqu'à 1 000 WCU/WRU par seconde et 3 000 RCU/RRU par seconde. Pour atténuer ces exceptions, nous vous recommandons de revoir votre modèle de données afin de répartir le trafic de lecture/écriture sur un plus grand nombre de partitions.
- `WriteThrottleEvents`— Demandes adressées à Amazon Keyspaces qui dépassent la capacité d'écriture d'une table.

Pour en savoir plus CloudWatch, consultezthe section called “Surveillance avec CloudWatch”. Pour obtenir la liste de toutes les CloudWatch statistiques disponibles pour Amazon Keyspaces, consultez. the section called “Métriques et dimensions”

> **Note**
>
> Pour commencer à utiliser un tableau de bord personnalisé qui affiche toutes les statistiques fréquemment observées pour Amazon Keyspaces, vous pouvez utiliser un CloudWatch modèle prédéfini disponible GitHub dans le AWS référentiel d'exemples.

Rubriques

Je reçois des erreurs de capacité **`NoHostAvailable`** insuffisante de la part de mon pilote client

Vous voyez **`Read_Timeout`** des **`Write_Timeout`** exceptions pour une table.

Les tentatives répétées d'écriture ou de lecture dans une table Amazon Keyspaces dont la capacité est insuffisante peuvent entraîner des erreurs côté client spécifiques au pilote.

CloudWatch À utiliser pour surveiller vos indicateurs de débit provisionnés et réels, ainsi que les événements de capacité insuffisante pour le tableau. Par exemple, une demande de lecture dont la capacité de débit est insuffisante échoue avec une `Read_Timeout` exception et est publiée dans la `ReadThrottleEvents` métrique. Une demande d'écriture dont la capacité de débit est insuffisante échoue avec une `Write_Timeout` exception et est publiée dans la `WriteThrottleEvents` métrique. Pour plus d'informations sur ces métriques, consultez the section called "Métriques et dimensions".

Pour résoudre ces problèmes, envisagez l'une des options suivantes.

- Augmentez le débit provisionné pour la table, qui correspond à la capacité de débit maximale qu'une application peut consommer. Pour plus d'informations, consultez the section called "Unités de capacité en lecture et unités de capacité en écriture".
- Laissez le service gérer la capacité de débit en votre nom grâce à la mise à l'échelle automatique. Pour plus d'informations, consultez the section called "Gérez la capacité de débit grâce à la mise à l'échelle automatique".
- Choisissez le mode de capacité à la demande pour le tableau. Pour plus d'informations, consultez the section called "Mode de capacité à la demande".

Si vous devez augmenter le quota de capacité par défaut de votre compte, consultezQuotas.

Des erreurs liées au dépassement de la capacité de partition s'affichent.

Lorsque le message d'erreur s'affiche, `StoragePartitionThroughputCapacityExceeded` la capacité de la partition est temporairement dépassée. Cela peut être géré automatiquement par la

capacité adaptative ou la capacité à la demande. Nous vous recommandons de revoir votre modèle de données afin de répartir le trafic de lecture/écriture sur un plus grand nombre de partitions afin de limiter ces erreurs. Les partitions de stockage Amazon Keyspaces peuvent prendre en charge jusqu'à 1 000 WCU/WRU par seconde et 3 000 RCU/RRU par seconde. Pour en savoir plus sur la manière d'améliorer votre modèle de données afin de répartir le trafic de lecture/écriture sur un plus grand nombre de partitions, consultez. the section called “Modélisation des données”

`Write_Timeout`les exceptions peuvent également être causées par un taux élevé d'opérations d'écriture simultanées qui incluent des données statiques et non statiques dans la même partition logique. Si le trafic est censé exécuter plusieurs opérations d'écriture simultanées incluant des données statiques et non statiques au sein de la même partition logique, nous vous recommandons d'écrire les données statiques et non statiques séparément. L'écriture séparée des données permet également d'optimiser les coûts de débit.

Vous constatez des erreurs liées au dépassement du taux de demandes de connexion.

Cela est `PerConnectionRequestRateExceeded` dû à l'une des causes suivantes.

- Il se peut que le nombre de connexions configurées par session ne soit pas suffisant.
- Il se peut que vous obteniez moins de connexions que vos homologues disponibles, car les autorisations de point de terminaison VPC ne sont pas correctement configurées. Pour plus d'informations sur les politiques relatives aux points de terminaison VPC, consultez. the section called “Utilisation des points de terminaison VPC de l'interface pour Amazon Keyspaces”
- Si vous utilisez un pilote 4.x, vérifiez si la validation du nom d'hôte est activée. Le pilote active la vérification du nom d'hôte TLS par défaut. Cette configuration fait apparaître Amazon Keyspaces sous la forme d'un cluster à nœud unique pour le pilote. Nous vous recommandons de désactiver la vérification du nom d'hôte.

Nous vous recommandons de suivre les meilleures pratiques suivantes pour optimiser vos connexions et votre débit :

- Configurez le réglage du débit des requêtes CQL.

  Amazon Keyspaces prend en charge jusqu'à 3 000 requêtes CQL par connexion TCP et par seconde, mais le nombre de connexions qu'un pilote peut établir est illimité.

  La plupart des pilotes Cassandra open source établissent un pool de connexions avec Cassandra et équilibrent la charge des requêtes sur ce pool de connexions. Amazon Keyspaces expose 9

adresses IP homologues aux conducteurs. Le comportement par défaut de la plupart des pilotes consiste à établir une connexion unique à chaque adresse IP homologue. Par conséquent, le débit maximal de requêtes CQL d'un pilote utilisant les paramètres par défaut sera de 27 000 requêtes CQL par seconde.

Pour augmenter ce nombre, nous vous recommandons d'augmenter le nombre de connexions par adresse IP que votre pilote gère dans son pool de connexions. Par exemple, si vous définissez le nombre maximum de connexions par adresse IP sur 2, vous doublerez le débit maximal de votre pilote pour le porter à 54 000 requêtes CQL par seconde.

- Optimisez vos connexions à nœud unique.

  Par défaut, la plupart des pilotes Cassandra open source établissent une ou plusieurs connexions à chaque adresse IP indiquée dans le `system.peers` tableau lors de l'établissement d'une session. Cependant, certaines configurations peuvent conduire un pilote à se connecter à une seule adresse IP Amazon Keyspaces. Cela peut se produire si le pilote tente de valider le nom d'hôte SSL des nœuds homologues (par exemple, les pilotes DataStax Java) ou lorsqu'il se connecte via un point de terminaison VPC.

  Pour bénéficier de la même disponibilité et des mêmes performances qu'un pilote connecté à plusieurs adresses IP, nous vous recommandons de procéder comme suit :

  - Augmentez le nombre de connexions par IP à 9 ou plus en fonction du débit client souhaité.
  - Créez une politique de nouvelles tentatives personnalisée qui garantit que les nouvelles tentatives sont exécutées sur le même nœud.
  - Si vous utilisez des points de terminaison VPC, accordez à l'entité IAM utilisée pour se connecter à Amazon Keyspaces l'autorisation d'interroger votre VPC pour obtenir les informations relatives au point de terminaison et à l'interface réseau. Cela améliore l'équilibrage de charge et augmente le débit de lecture/écriture. Pour plus d'informations, consultez ???.

Je reçois des erreurs de délai d'écriture lors de l'importation des données

Vous recevez une erreur de temporisation lorsque vous chargez des données à l'aide de la **`cqlsh COPY`** commande.

```
Failed to import 1 rows: WriteTimeout - Error from server: code=1100 [Coordinator node
 timed out waiting for replica nodes' responses]
 message="Operation timed out - received only 0 responses." info={'received_responses':
 0, 'required_responses': 2, 'write_type': 'SIMPLE', 'consistency':
```

```
'LOCAL_QUORUM'}, will retry later, attempt 1 of 100
```

Amazon Keyspaces utilise les `WriteTimeout` exceptions `ReadTimeout` et pour indiquer lorsqu'une demande d'écriture échoue en raison d'une capacité de débit insuffisante. Pour aider à diagnostiquer les exceptions de capacité insuffisante, Amazon Keyspaces publie les statistiques suivantes sur Amazon. CloudWatch

- `WriteThrottleEvents`
- `ReadThrottledEvents`
- `StoragePartitionThroughputCapacityExceeded`

Pour résoudre les erreurs de capacité insuffisante lors du chargement de données, réduisez le taux d'écriture par travailleur ou le taux d'ingestion total, puis réessayez de télécharger les lignes. Pour plus d'informations, consultez the section called "Étape 4 : Configuration des `cqlsh COPY FROM` paramètres". Pour une option de téléchargement de données plus robuste, pensez à utiliser DSBulk, disponible dans le GitHub référentiel. Pour step-by-step obtenir des instructions, voirthe section called "Chargement de données à l'aide de DSBulk".

Je ne vois pas la taille de stockage réelle d'un keyspace ou d'une table

Vous ne pouvez pas voir la taille de stockage réelle du keyspace ou de la table.

Pour en savoir plus sur la taille de rangement de votre table, consultezthe section called "Évaluer les coûts au niveau de la table". Vous pouvez également estimer la taille de stockage en commençant par calculer la taille des lignes dans un tableau. Des instructions détaillées pour le calcul de la taille des lignes sont disponibles surthe section called "Calcul de la taille des lignes".

Résolution des problèmes liés au langage de définition des données dans Amazon Keyspaces

Vous rencontrez des difficultés pour créer des ressources ? Voici quelques problèmes courants et comment les résoudre.

Erreurs de langage de définition des données

Amazon Keyspaces exécute des opérations de langage de définition de données (DDL) de manière asynchrone, par exemple en créant et en supprimant des espaces clés et des tables. Si une application essaie d'utiliser la ressource avant qu'elle ne soit prête, l'opération échoue.

Vous pouvez surveiller l'état de création de nouveaux espaces de touches et de nouvelles tables dans le AWS Management Console, ce qui indique lorsqu'un espace de touches ou une table est en attente ou actif. Vous pouvez également surveiller l'état de création d'un nouvel espace de touches ou d'une nouvelle table par programmation en interrogeant la table du schéma du système. Un espace de touches ou une table devient visible dans le schéma du système lorsqu'il est prêt à être utilisé.

> **Note**
>
> Pour optimiser la création d'espaces clés à l'aide de cet utilitaire AWS CloudFormation, vous pouvez utiliser cet utilitaire pour convertir des scripts CQL en CloudFormation modèles. L'outil est disponible dans le GitHub référentiel.

Rubriques

J'ai créé un nouvel espace de touches, mais je ne parviens pas à le voir ou à y accéder

Vous recevez des erreurs de la part de votre application qui tente d'accéder à un nouvel espace de touches.

Si vous essayez d'accéder à un espace de touches Amazon Keyspaces récemment créé mais toujours en cours de création asynchrone, vous recevrez un message d'erreur. L'erreur suivante est un exemple.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 keyspace mykeyspace"
```

Le modèle de conception recommandé pour vérifier qu'un nouvel espace de touches est prêt à être utilisé consiste à interroger les tables de schéma du système Amazon Keyspaces (system_schema_mcs.*).

Pour plus d'informations, consultez the section called "Création d'espaces clés".

J'ai créé une nouvelle table, mais je ne parviens pas à la consulter ou à y accéder

Vous recevez des erreurs de la part de votre application qui tente d'accéder à une nouvelle table.

Si vous essayez d'accéder à une table Amazon Keyspaces récemment créée mais toujours en cours de création asynchrone, vous recevrez un message d'erreur. Par exemple, une tentative d'interrogation d'une table qui n'est pas encore disponible échoue avec une `unconfigured table` erreur.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 table mykeyspace.mytable"
```

La tentative d'affichage de la table avec `sync_table()` échoue avec un`KeyError`.

```
KeyError: 'mytable'
```

Le modèle de conception recommandé pour vérifier qu'une nouvelle table est prête à être utilisée consiste à interroger les tables du schéma du système Amazon Keyspaces (system_schema_mcs.*).

Il s'agit de l'exemple de sortie d'une table en cours de création.

```
user-at-123@cqlsh:system_schema_mcs> select table_name,status from
 system_schema_mcs.tables where keyspace_name='example_keyspace' and
 table_name='example_table';

table_name | status

------------+----------

example_table | CREATING
```

```
(1 rows)
```

Voici un exemple de sortie pour une table active.

```
user-at-123@cqlsh:system_schema_mcs> select table_name,status from
 system_schema_mcs.tables where keyspace_name='example_keyspace' and
 table_name='example_table';

table_name | status

------------+----------

example_table | ACTIVE

(1 rows)
```

Pour plus d'informations, consultez the section called “Création de tables”.

J'essaie de restaurer une table à l'aide de Amazon Keyspaces point-in-time Recovery (PITR), mais la restauration échoue

Si vous essayez de restaurer une table Amazon Keyspaces avec point-in-time restauration (PITR) et que le processus de restauration commence mais ne se termine pas correctement, vous n'avez peut-être pas configuré toutes les autorisations requises pour le processus de restauration pour cette table en particulier.

Outre les autorisations des utilisateurs, Amazon Keyspaces peut avoir besoin d'autorisations pour effectuer des actions au nom de votre principal pendant le processus de restauration. C'est le cas si la table est chiffrée à l'aide d'une clé gérée par le client ou si vous utilisez des politiques IAM qui limitent le trafic entrant.

Par exemple, si vous utilisez des clés de condition dans votre politique IAM pour restreindre le trafic source à des points de terminaison ou à des plages d'adresses IP spécifiques, l'opération de restauration échoue. Pour permettre à Amazon Keyspaces d'effectuer l'opération de restauration des tables pour le compte de votre principal, vous devez ajouter une clé de condition `aws:ViaAWSService` globale dans la politique IAM.

Pour plus d'informations sur les autorisations de restauration de tables, consultezthe section called “Restaurer les autorisations”.

J'essaie d'utiliser INSERT/UPDATE pour modifier les paramètres personnalisés de durée de vie (TTL), mais l'opération échoue

Si vous essayez d'insérer ou de mettre à jour une valeur TTL personnalisée, l'opération risque d'échouer avec l'erreur suivante.

```
TTL is not yet supported.
```

Pour spécifier des valeurs TTL personnalisées pour les lignes ou les colonnes à l'aide `UPDATE` des opérations `INSERT` OR, vous devez d'abord activer le TTL pour la table. Vous pouvez activer le TTL pour un tableau à l'aide de la propriété `ttl` personnalisée.

Pour plus d'informations sur l'activation des paramètres TTL personnalisés pour les tables, consultezthe section called “Comment activer le Time-à-live (Tlive) sur des tables existantes à l'aide de propriétés personnalisées”.

J'essaie de télécharger des données dans ma table Amazon Keyspaces et je reçois un message d'erreur indiquant que le nombre de colonnes est dépassé

Vous téléchargez des données et vous avez dépassé le nombre de colonnes pouvant être mises à jour simultanément.

Cette erreur se produit lorsque le schéma de votre table dépasse la taille maximale de 350 Ko. Pour plus d’informations, consultez Quotas.

J'essaie de supprimer des données de ma table Amazon Keyspaces et la suppression échoue pour la plage

Vous essayez de supprimer des données par clé de partition et vous recevez un message d'erreur de suppression de plage.

Cette erreur se produit lorsque vous essayez de supprimer plus de 1 000 lignes en une seule opération de suppression.

```
Range delete requests are limited by the amount of items that can be deleted in a
 single range.
```

Pour plus d’informations, consultez the section called “Supprimer une plage”.

Pour supprimer plus de 1 000 lignes au sein d'une même partition, considérez les options suivantes.

- Supprimer par partition — Si la majorité des partitions comportent moins de 1 000 lignes, vous pouvez essayer de supprimer les données par partition. Si les partitions contiennent plus de 1 000 lignes, essayez plutôt de les supprimer par la colonne de clustering.
- Supprimer par colonne de clustering : si votre modèle contient plusieurs colonnes de clustering, vous pouvez utiliser la hiérarchie des colonnes pour supprimer plusieurs lignes. Les colonnes de clustering sont une structure imbriquée et vous pouvez supprimer de nombreuses lignes en agissant sur la colonne de niveau supérieur.
- Supprimer par ligne individuelle — Vous pouvez parcourir les lignes par itération et supprimer chaque ligne à l'aide de sa clé primaire complète (colonnes de partition et colonnes de clustering).
- Il est recommandé de diviser vos lignes sur des partitions. Dans Amazon Keyspaces, nous vous recommandons de répartir votre débit sur des partitions de table. Cela répartit les données et l'accès de manière uniforme sur les ressources physiques, ce qui permet d'obtenir le meilleur débit. Pour plus d’informations, consultez the section called “Modélisation des données”.

Tenez également compte des recommandations suivantes lorsque vous planifiez des opérations de suppression pour des charges de travail importantes.

- Avec Amazon Keyspaces, les partitions peuvent contenir un nombre de lignes pratiquement illimité. Cela vous permet de redimensionner les partitions de manière « plus large » que le guide Cassandra traditionnel de 100 Mo. Il n'est pas rare que les séries chronologiques ou les registres augmentent de plus d'un gigaoctet de données au fil du temps.
- Avec Amazon Keyspaces, il n'y a aucune stratégie de compactage ni aucune pierre angulaire à prendre en compte lorsque vous devez effectuer des opérations de suppression pour des charges de travail importantes. Vous pouvez supprimer autant de données que vous le souhaitez sans affecter les performances de lecture.

Gestion des ressources sans serveur dans Amazon Keyspaces (pour Apache Cassandra)

Amazon Keyspaces (pour Apache Cassandra) fonctionne sans serveur. Au lieu de déployer, de gérer et de maintenir les ressources de stockage et de calcul pour votre charge de travail via les nœuds d'un cluster, Amazon Keyspaces alloue les ressources de stockage et de débit de lecture/écriture directement aux tables.

Ce chapitre fournit des informations sur la gestion des ressources sans serveur dans Amazon Keyspaces. Pour savoir comment surveiller les ressources sans serveur avec Amazon CloudWatch, consultezthe section called “Surveillance avec CloudWatch”.

Rubriques

Stockage sur Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) fournit automatiquement le stockage aux tables en fonction des données réellement stockées dans votre table. Il n'est pas nécessaire de provisionner le stockage sur les tables à l'avance. Amazon Keyspaces augmente ou diminue automatiquement le stockage de votre table au fur et à mesure que votre application écrit, met à jour et supprime des données. Contrairement aux clusters Apache Cassandra traditionnels, Amazon Keyspaces n'a pas besoin de stockage supplémentaire pour prendre en charge les opérations système de bas niveau telles que le compactage. Vous ne payez que pour l'espace de stockage que vous utilisez.

Amazon Keyspaces configure les espaces clés avec un facteur de réplication de trois par défaut. Vous ne pouvez pas modifier le facteur de réplication. Amazon Keyspaces réplique les données des tables trois fois automatiquement dans plusieurs zones de disponibilité pour une AWS haute disponibilité. Le prix par Go du stockage Amazon Keyspaces inclut déjà la réplication. Consultez les tarifs d'Amazon Keyspaces (pour Apache Cassandra) pour plus d'informations.

Amazon Keyspaces surveille en permanence la taille de vos tables afin de déterminer vos frais de stockage. Pour plus d'informations sur la façon dont Amazon Keyspaces calcule la taille facturable des données, consultez. the section called “Calcul de la taille des lignes”

Modes de capacité de lecture/écriture dans Amazon Keyspaces

Amazon Keyspaces propose deux modes de capacité de lecture/écriture pour traiter les lectures et les écritures sur vos tables :

- À la demande (par défaut)
- Alloué

Le mode de capacité de lecture/écriture que vous choisissez contrôle la façon dont vous êtes facturé pour le débit de lecture et d'écriture et comment la capacité de débit de table est gérée.

Rubriques

Mode de capacité à la demande

Le mode de capacité à la demande d'Amazon Keyspaces (pour Apache Cassandra) est une option de facturation flexible capable de traiter des milliers de demandes par seconde sans planification des capacités. Cette option propose une pay-per-request tarification pour les demandes de lecture et d'écriture afin que vous ne payiez que pour ce que vous utilisez.

Lorsque vous choisissez le mode à la demande, Amazon Keyspaces peut augmenter instantanément la capacité de débit de votre table jusqu'à n'importe quel niveau de trafic précédemment atteint, puis redescendre à la baisse lorsque le trafic des applications diminue. Si le niveau de trafic d'une charge globale atteint un nouveau pic, le service s'adapte rapidement pour augmenter la capacité de débit de votre table. Vous pouvez activer le mode de capacité à la demande pour les tables nouvelles et existantes.

Le mode à la demande est une bonne option si l'une des conditions suivantes est remplie :

- Vous créez des tables avec des charges de travail inconnues.
- Vous avez un trafic imprévisible au niveau de l'application.
- Vous préférez ne payer qu'à l'utilisation.

Pour démarrer avec le mode à la demande, vous pouvez créer une nouvelle table ou mettre à jour une table existante pour utiliser le mode capacité à la demande à l'aide de la console ou avec quelques lignes de code CQL (Cassandra Query Language). Pour plus d'informations, consultez the section called "Tables".

Rubriques

- Unités de demande de lecture et unités de demande d'écriture
- Trafic de pointe et propriétés de scalabilité
- Débit initial pour le mode de capacité à la demande

Unités de demande de lecture et unités de demande d'écriture

Avec les tables du mode de capacité à la demande, vous n'avez pas besoin de spécifier à l'avance le débit de lecture et d'écriture que vous comptez utiliser par votre application. Amazon Keyspaces vous facture les lectures et les écritures que vous effectuez sur vos tables en termes d'unités de demande de lecture (RRU) et d'unités de demande d'écriture (WRU).

- Une RRU représente une `LOCAL_QUORUM` ou deux demandes de `LOCAL_ONE` lecture pour une ligne d'une taille maximale de 4 Ko. Si vous devez lire une ligne de plus de 4 Ko, l'opération de lecture utilise des RRU supplémentaires. Le nombre total de RRU requis dépend de la taille de la ligne et de l'utilisation ou pas de la cohérence de lecture `LOCAL_QUORUM` ou `LOCAL_ONE`. Par exemple, la lecture d'une ligne de 8 Ko nécessite 2 RRU utilisant la cohérence de lecture `LOCAL_QUORUM` et 1 RRU si vous choisissez la cohérence de lecture `LOCAL_ONE`.
- Un WRU représente une écriture pour une ligne d'une taille maximale de 1 Ko. Toutes les écritures utilisent la cohérence `LOCAL_QUORUM` et il n'y a pas de frais supplémentaires pour l'utilisation de transactions légères (LWT). Si vous devez écrire une ligne supérieure à 1 Ko, l'opération d'écriture utilise des WRU supplémentaires. Le nombre total de WRU requis dépend de la taille de la ligne. Par exemple, si la taille de votre ligne est de 2 Ko, vous avez besoin de 2 WRU pour effectuer une requête d'écriture.

Pour de plus amples informations sur les niveaux de cohérence pris en charge, veuillez consulter the section called “Niveaux de cohérence Cassandra pris en charge”.

Trafic de pointe et propriétés de scalabilité

Les tables Amazon Keyspaces qui utilisent le mode capacité à la demande s'adaptent automatiquement au volume de trafic de votre application. Le mode de capacité à la demande peut gérer jusqu'à deux fois le trafic de pointe précédent d'une table. Par exemple, le modèle de trafic de votre application peut varier entre 5 000 et 10 000 lectures de `LOCAL_QUORUM` par seconde, 10 000 lectures par seconde étant le pic de trafic précédent.

Avec ce modèle, le mode de capacité à la demande prend instantanément en charge un trafic soutenu pouvant atteindre 20 000 lectures par seconde. Si votre application présente un trafic de 20 000 lectures par seconde, ce pic devient votre nouvelle capacité de pointe précédente, permettant ainsi au trafic futur d'atteindre jusqu'à 40 000 lectures par seconde.

Si vous avez besoin de plus du double de votre pic précédent sur une table, Amazon Keyspaces alloue automatiquement plus de capacité à mesure que votre volume de trafic augmente. Cela permet de s'assurer que votre table dispose d'une capacité de débit suffisante pour traiter les demandes supplémentaires. Toutefois, vous pouvez observer des erreurs de capacité de débit insuffisante si vous dépassez le double de votre pic précédent en 30 minutes.

Par exemple, supposons que le modèle de trafic de votre application varie entre 5 000 et 10 000 lectures fortement cohérentes par seconde, où 20 000 lectures par seconde correspondent au pic de trafic atteint précédemment. Dans ce cas, le service vous recommande d'espacer votre croissance de trafic sur au moins 30 minutes avant de conduire jusqu'à 40 000 lectures par seconde.

Pour savoir comment estimer la consommation de capacité de lecture et d'écriture d'une table, voirthe section called “Estimation de la consommation de capacité”.

Pour en savoir plus sur les quotas par défaut de votre compte et sur la façon de les augmenter, reportez-vous à la section Quotas.

Débit initial pour le mode de capacité à la demande

Si vous créez une table avec un mode de capacité à la demande ou basculez une table existante vers le mode de capacité à la demande pour la première fois, la table possède les paramètres suivants du trafic de pointe précédent, même si la table n'a pas encore opéré de trafic en mode de capacité à la demande :

- Table nouvellement créée avec mode capacité à la demande : le pic précédent était de 2 000 WRU et 6 000 RRU. Vous pouvez poursuivre jusqu'à doubler le pic précédent immédiatement. Cela permet aux tables à la demande nouvellement créées de servir jusqu'à 4 000 WRU et 12 000 RRU.
- Table existante basculée vers le mode de capacité à la demande : le trafic de pointe précédent équivaut à la moitié des unités de capacité en écriture (WCU) et des unités de capacité en lecture (RCU) allouées à la table ou aux paramètres pour une table nouvellement créée avec le mode de capacité à la demande, selon la valeur la plus élevée.

Mode de capacité de débit provisionnée

Si vous choisissez le mode de capacité de débit alloué vous spécifiez le nombre de lectures et d'écritures par seconde requis pour votre application. Cela vous permet de gérer votre utilisation d'Amazon Keyspaces afin de rester au niveau ou en dessous d'un taux de demandes défini afin d'optimiser le prix et de garantir la prévisibilité. Pour en savoir plus sur la mise à l'échelle automatique du débit alloué, reportez-vous à la section the section called “Gérez la capacité de débit grâce à la mise à l'échelle automatique”.

Le mode de capacité de débit alloué est une bonne option si l'une des conditions suivantes est vraie :

- Votre application présente un niveau de trafic prévisible.
- Vous exécutez des applications dont le trafic est constant ou évolue progressivement.
- Vous pouvez prévoir les besoins en capacité pour optimiser le prix.

Unités de capacité en lecture et unités de capacité en écriture

Pour les tables en mode alloué, vous spécifiez votre capacité de débit en termes d'unités de capacité en lecture (RCU) et d'unités de capacité en écriture (WCU) :

- Un RCU représente une lecture de `LOCAL_QUORUM` par seconde ou deux lectures de `LOCAL_ONE` par seconde, pour une ligne d'une taille maximale de 4 Ko. Si vous devez lire une ligne de plus de 4 Ko, l'opération de lecture utilise des RCU supplémentaires.

  Le nombre total de RCU requis dépend de la taille de la ligne et du nombre de lecture de `LOCAL_ONE` ou de `LOCAL_QUORUM` souhaitées. Par exemple, si la taille de votre ligne est de 8 Ko, vous avez besoin de 2 RCU pour supporter une lecture de `LOCAL_QUORUM` par seconde et d'1 RCU si vous choisissez les lectures de `LOCAL_ONE`.

- Une WCU représente une écriture par seconde pour une ligne d'une taille maximale de 1 Ko. Toutes les écritures utilisent la cohérence `LOCAL_QUORUM` et il n'y a pas de frais supplémentaires pour l'utilisation de transactions légères (LWT). Si vous devez écrire une ligne supérieure à 1 Ko, l'opération d'écriture utilise des WCU supplémentaires.

  Le nombre total de WCU requis dépend de la taille de la ligne. Par exemple, si la taille de votre ligne est de 2 Ko, vous avez besoin de 2 WCU pour supporter une demande d'écriture par seconde. Pour plus d'informations sur la façon d'estimer la consommation de capacité de lecture et d'écriture d'une table, consultezthe section called “Estimation de la consommation de capacité”.

Si votre application lit ou écrit des lignes plus grandes (jusqu'à la taille de ligne maximale d'Amazon Keyspaces de 1 Mo), elle consomme davantage d'unités de capacité. Pour en savoir plus sur l'estimation de la taille des lignes, consultezthe section called “Calcul de la taille des lignes”. Par exemple, supposons que vous créiez une table allouée avec 6 RCU et 6 WCU. Avec ces paramètres, votre application pourrait effectuer les opérations suivantes :

- Effectuez des `LOCAL_QUORUM` lectures allant jusqu'à 24 Ko par seconde (4 Ko × 6 RCU).
- Effectuez des lectures de `LOCAL_ONE` allant jusqu'à 48 Ko par seconde (deux fois plus de débit de lecture).
- Écrivez jusqu'à 6 Ko par seconde (1 Ko × 6 WCU).

Le Débit alloué est la quantité maximum de capacité qu'une application peut consommer à partir d'une table. Si votre application dépasse votre capacité de débit alloué, vous pouvez observer des erreurs de capacité insuffisante.

Par exemple, une demande de lecture dont la capacité de débit est insuffisante échoue avec une `Read_Timeout` exception et est publiée dans la `ReadThrottleEvents` métrique. Une demande d'écriture dont la capacité de débit est insuffisante échoue avec une `Write_Timeout` exception et est publiée dans la `WriteThrottleEvents` métrique.

Vous pouvez utiliser Amazon CloudWatch pour surveiller vos indicateurs de débit provisionnés et réels ainsi que les événements de capacité insuffisante. Pour plus d’informations sur ces métriques, consultez the section called “Métriques et dimensions”.

> **Note**
>
> Des erreurs répétées dues à une capacité insuffisante peuvent entraîner des exceptions spécifiques au pilote côté client, par exemple si le pilote DataStax Java échoue avec un. `NoHostAvailableException`

Pour modifier les paramètres de capacité de débit des tables, vous pouvez utiliser l'instruction AWS Management Console ou l'`ALTER TABLE`instruction à l'aide de CQL. Pour plus d'informations, voir. the section called “ALTER TABLE”

Pour en savoir plus sur les quotas par défaut de votre compte et sur la façon de les augmenter, reportez-vous à la section Quotas.

Modes de gestion et de visualisation de la capacité

Vous pouvez interroger la table système dans l'espace de touches du système Amazon Keyspaces pour consulter les informations relatives au mode capacité relatives à une table. Vous pouvez également voir si une table utilise le mode de capacité de débit à la demande ou allouée. Si la table est configurée avec le mode de capacité de débit alloué, vous pouvez voir la capacité de débit alloué pour la table.

Exemple

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'mykeyspace' and
 table_name = 'mytable';
```

Une table configurée avec le mode de capacité à la demande renvoie ce qui suit.

```
                  {
                      'capacity_mode': {
                          'last_update_to_pay_per_request_timestamp':
'1579551547603',
                          'throughput_mode': 'PAY_PER_REQUEST'
                       }
                  }
```

Une table configurée avec le mode de capacité de débit alloué renvoie ce qui suit.

```
                    {
                        'capacity_mode': {
                            'last_update_to_pay_per_request_timestamp':
'1579048006000',
                            'read_capacity_units': '5000',
                            'throughput_mode': 'PROVISIONED',
                            'write_capacity_units': '6000'
                        }
                     }
```

La valeur `last_update_to_pay_per_request_timestamp` est mesurée en millisecondes.

Pour modifier la capacité de débit alloué d'une table, utilisez the section called “ALTER TABLE”.

Considérations relatives au changement de mode de capacité

Lorsque vous passez d'une table du mode capacité allouée au mode capacité à la demande, Amazon Keyspaces apporte plusieurs modifications à la structure de votre table et de vos partitions. Ce processus peut prendre plusieurs minutes. Pendant la période de commutation, votre table fournit un débit compatible avec les montants WCU et RCU précédemment alloués.

Si vous passez du mode de capacité à la demande au mode de capacité allouée, votre table fournit un débit correspondant au trafic de pointe précédent atteint lorsque la table était en mode de capacité à la demande.

> **Note**
>
> Vous ne pouvez passer du mode de capacité provisionné au mode à la demande qu'une seule fois par période de 24 heures.

Gérez automatiquement la capacité de débit grâce au dimensionnement automatique d'Amazon Keyspaces

De nombreuses charges de travail de base de données sont cycliques par nature et difficiles à prévoir. Prenons l'exemple d'une application de réseau social où la plupart des utilisateurs sont actifs

pendant la journée. La base de données doit être capable de gérer l'activité durant cette période, mais elle n'a pas besoin des mêmes niveaux de débit pendant la nuit.

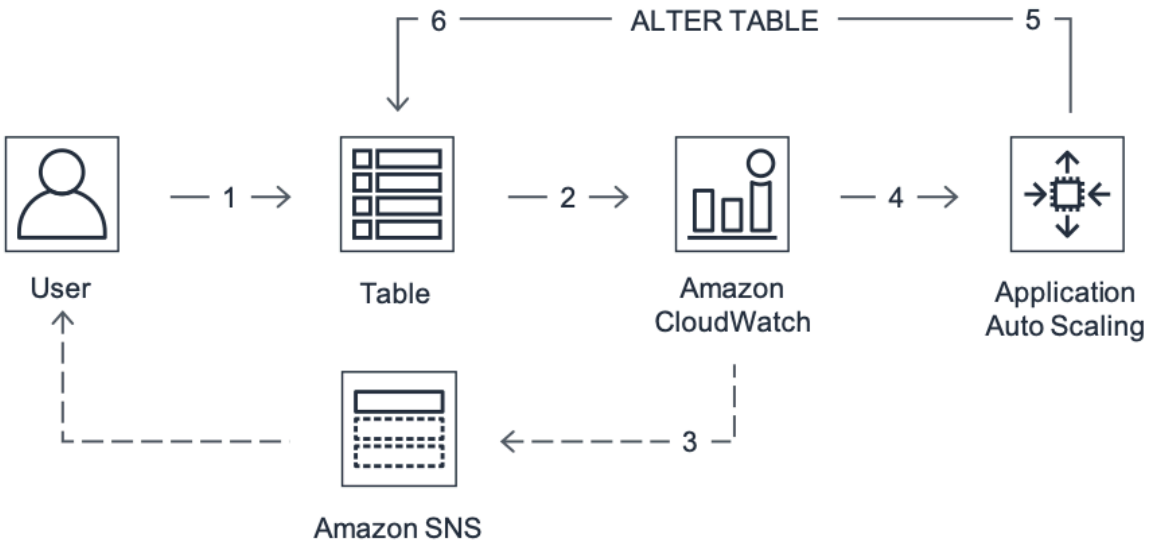
Un autre exemple : supposons que vous ayez une nouvelles application de jeux pour appareils mobiles qui est rapidement adoptée par les utilisateurs. Si le jeu devient trop populaire, les ressources de base de données disponibles risquent d'être dépassées, entraînant un ralentissement des performances et le mécontentement des clients. Ce type de charges de travail nécessite souvent une intervention manuelle pour augmenter ou diminuer les ressources de base de données en fonction de la variation des niveaux d'utilisation.

Amazon Keyspaces (pour Apache Cassandra) vous aide à fournir une capacité de débit efficace pour des charges de travail variables en ajustant automatiquement la capacité de débit en réponse au trafic réel des applications. Amazon Keyspaces utilise le service Application Auto Scaling pour augmenter et diminuer la capacité de lecture et d'écriture d'une table en votre nom. Pour plus d'informations sur Application Auto Scaling, consultez le Guide de l'utilisateur d'Application Auto Scaling.

> **Note**
>
> Pour démarrer rapidement avec le dimensionnement automatique d'Amazon Keyspaces, consultez. the section called “Utilisation de la console” Pour gérer les politiques de dimensionnement d'Amazon Keyspaces avec Cassandra Query Language (CQL), consultez. the section called “Utilisation de CQL” Pour savoir comment gérer les politiques de dimensionnement d'Amazon Keyspaces à l'aide de la CLI, consultez. the section called “Utilisation de la CLI”

Comment fonctionne le dimensionnement automatique d'Amazon Keyspaces

Le schéma suivant fournit une vue d'ensemble détaillée de la façon dont le dimensionnement automatique d'Amazon Keyspaces gère la capacité de débit d'une table.



Pour activer la mise à l'échelle automatique d'une table , vous devez créer une stratégie de mise à l'échelle. La stratégie de dimensionnement permet de spécifier si vous souhaitez dimensionner la capacité en lecture ou en écriture (ou les deux), ainsi que les paramètres d'unité de capacité allouée minimum et maximum pour la table.

La stratégie de mise à l'échelle définit également une utilisation cible. L'utilisation cible correspond au ratio unités de capacité consommées/unités de capacité allouées à un point dans le temps, exprimé sous forme de pourcentage. La mise à l'échelle automatique utilise un algorithme de suivi de cible pour ajuster le débit alloué de la table vers le haut ou vers le bas en réponse aux charges de travail réelles. L'utilisation réelle de la capacité reste ainsi proche de votre utilisation cible.

Vous pouvez également définir les valeurs d'utilisation cibles de la mise à l'échelle automatique entre 20 % et 90 % pour votre capacité de lecture et d'écriture. Le taux d'utilisation cible par défaut est de 70 %. Vous pouvez définir l'utilisation cible à un pourcentage inférieur si votre trafic change rapidement et si vous souhaitez que la capacité commence à augmenter plus tôt. Vous pouvez également définir le taux d'utilisation cible sur un taux plus élevé si le trafic de votre application change plus lentement et que vous souhaitez réduire le coût du débit.

Pour plus d'informations sur les politiques de dimensionnement, consultez la section Politiques de dimensionnement du suivi des cibles pour Application Auto Scaling dans le Guide de l'utilisateur d'Application Auto Scaling.

Lorsque vous créez une politique de dimensionnement, Amazon Keyspaces crée deux paires d' CloudWatch alarmes Amazon en votre nom. Chaque paire représente vos limites supérieure et

inférieure pour les paramètres de débit alloué. Ces CloudWatch alarmes sont déclenchées lorsque l'utilisation réelle de la table s'écarte de votre utilisation cible pendant une période prolongée. Pour en savoir plus sur Amazon CloudWatch, consultez le guide de CloudWatch l'utilisateur Amazon.

Lorsque l'une des CloudWatch alarmes est déclenchée, Amazon Simple Notification Service (Amazon SNS) vous envoie une notification (si vous l'avez activée). L' CloudWatch alarme appelle ensuite Application Auto Scaling pour évaluer votre politique de dimensionnement. Cela envoie ensuite une demande Alter Table à Amazon Keyspaces pour ajuster la capacité allouée à la table à la hausse ou à la baisse, selon le cas. Pour en savoir plus sur les notifications Amazon SNS, consultez Configuration des notifications Amazon SNS.

Amazon Keyspaces traite la demande Alter Table en augmentant (ou en diminuant) la capacité de débit allouée à la table afin qu'elle se rapproche de votre objectif d'utilisation.

> **Note**
>
> Le scalage automatique d'Amazon Keyspaces modifie les paramètres de débit provisionnés uniquement lorsque la charge de travail réelle reste élevée (ou diminuée) pendant une période prolongée de plusieurs minutes. L'algorithme de suivi cherche à garder l'utilisation cible au niveau ou proche de la valeur choisie sur le long terme. Les pics soudains de l'activité de lecture sont gérés par la capacité de transmission en mode rafale intégrée de la table.

Comment fonctionne la mise à l'échelle automatique pour les tables multirégionales

Pour garantir que la capacité de lecture et d'écriture est toujours suffisante pour toutes les répliques de table dans Régions AWS l'ensemble d'une table multirégionale en mode capacité allouée, nous vous recommandons de configurer le dimensionnement automatique d'Amazon Keyspaces.

Lorsque vous utilisez une table multirégionale en mode provisionné avec mise à l'échelle automatique, vous ne pouvez pas désactiver la mise à l'échelle automatique pour une seule réplique de table. Mais vous pouvez ajuster les paramètres de mise à l'échelle automatique de lecture du tableau pour différentes régions. Par exemple, vous pouvez spécifier des paramètres de capacité de lecture et de mise à l'échelle automatique de lecture différents pour chaque région dans laquelle la table est répliquée.

Les paramètres de redimensionnement automatique en lecture que vous configurez pour une réplique de table dans une région spécifiée remplacent les paramètres généraux de mise à l'échelle automatique de la table. La capacité d'écriture doit toutefois rester synchronisée entre toutes les répliques de tables afin de garantir une capacité suffisante pour répliquer les écritures dans toutes les régions.

Le scalage automatique d'Amazon Keyspaces met à jour indépendamment la capacité allouée à la table dans chacune d'entre elles en Région AWS fonction de l'utilisation dans cette région. Par conséquent, la capacité allouée dans chaque région pour une table multirégionale peut être différente lorsque le dimensionnement automatique est activé.

Vous pouvez configurer les paramètres de dimensionnement automatique d'une table multirégionale et de ses répliques à l'aide de la console Amazon Keyspaces, de l'API ou du CQL. AWS CLI Pour plus d'informations sur la façon de créer et de mettre à jour les paramètres de mise à l'échelle automatique pour les tables multirégionales, consultezthe section called “Comment utiliser la réplication multirégionale”.

> **Note**
>
> Si vous utilisez le dimensionnement automatique pour les tables multirégionales, vous devez toujours utiliser les opérations de l'API Amazon Keyspaces pour configurer les paramètres de dimensionnement automatique. Si vous utilisez directement les opérations de l'API Application Auto Scaling pour configurer les paramètres de dimensionnement automatique, vous n'êtes pas en mesure Régions AWS de spécifier le tableau multirégional. Cela peut entraîner des configurations non prises en charge.

Notes d’utilisation

Avant de commencer à utiliser le dimensionnement automatique d'Amazon Keyspaces, vous devez prendre en compte les points suivants :

- Le dimensionnement automatique d'Amazon Keyspaces peut augmenter la capacité de lecture ou d'écriture aussi souvent que nécessaire, conformément à votre politique de dimensionnement. Tous les quotas Amazon Keyspaces restent en vigueur, comme décrit dans. Quotas
- Le dimensionnement automatique d'Amazon Keyspaces ne vous empêche pas de modifier manuellement les paramètres de débit provisionnés. Ces ajustements manuels n'affectent pas les CloudWatch alarmes existantes associées à la politique de dimensionnement.

- Si vous utilisez la console pour créer une table avec une capacité de débit allouée, le dimensionnement automatique d'Amazon Keyspaces est activé par défaut. Vous pouvez modifier vos paramètres de mise à l'échelle automatique à tout moment. Pour plus d'informations, consultez the section called "Utilisation de la console".
- Si vous utilisez AWS CloudFormation pour créer des politiques de dimensionnement, vous devez gérer les politiques de dimensionnement de AWS CloudFormation manière à ce que la pile soit synchronisée avec le modèle de pile. Si vous modifiez les politiques de dimensionnement d'Amazon Keyspaces, elles seront remplacées par les valeurs d'origine du modèle de pile lorsque la AWS CloudFormation pile sera réinitialisée.
- Si vous surveillez le dimensionnement automatique CloudTrail d'Amazon Keyspaces, vous pouvez recevoir des alertes concernant les appels effectués par Application Auto Scaling dans le cadre de son processus de validation de configuration. Vous pouvez filtrer ces alertes en utilisant le `invokedBy` champ qui contient `application-autoscaling.amazonaws.com` ces contrôles de validation.

Gestion des politiques de dimensionnement automatique d'Amazon Keyspaces avec la console

Vous pouvez utiliser la console pour activer le dimensionnement automatique d'Amazon Keyspaces pour les tables nouvelles et existantes. Vous pouvez également utiliser la console pour modifier les paramètres de mise à l'échelle automatique ou désactiver la mise à l'échelle automatique.

> **Note**
>
> Pour des fonctionnalités plus avancées, telles que la définition des temps de recharge évolutifs et externes, utilisez CQL ou the () AWS Command Line Interface pour AWS CLI gérer les politiques de dimensionnement d'Amazon Keyspaces de manière programmatique. Pour plus d'informations, consultez Gestion du dimensionnement automatique d'Amazon Keyspaces avec Cassandra Query Language (CQL) ou Gestion des politiques de dimensionnement d'Amazon Keyspaces à l'aide de la CLI.

Rubriques

- Activation du dimensionnement automatique d'Amazon Keyspaces sur les tables existantes
- Modifier ou désactiver les paramètres de dimensionnement automatique d'Amazon Keyspaces
- Afficher les activités de dimensionnement automatique d'Amazon Keyspaces sur la console

Avant de commencer : accorder des autorisations aux utilisateurs pour le dimensionnement automatique d'Amazon Keyspaces

Pour commencer, vérifiez que l'utilisateur dispose des autorisations appropriées pour créer et gérer les paramètres de mise à l'échelle automatique. Dans AWS Identity and Access Management (IAM), la politique AWS gérée `AmazonKeyspacesFullAccess` est requise pour gérer les politiques de dimensionnement d'Amazon Keyspaces.

> **Important**
>
> Les autorisations `application-autoscaling:*` sont nécessaires pour désactiver la mise à l'échelle automatique sur une table. Vous devez désactiver la mise à l'échelle automatique pour une table avant de pouvoir la supprimer.

Pour configurer un utilisateur IAM pour l'accès à la console Amazon Keyspaces et le dimensionnement automatique d'Amazon Keyspaces, ajoutez la politique suivante.

Pour joindre la **`AmazonKeyspacesFullAccess`** politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le tableau de bord de la console IAM, choisissez Users (Utilisateurs), puis votre utilisateur IAM dans la liste.
3. Sur la page Récapitulatif, choisissez Ajouter des autorisations.
4. Choisissez Attacher directement les stratégies existantes.
5. Dans la liste des politiques, choisissez AmazonKeyspacesFullAccess, puis cliquez sur Suivant : Réviser.
6. Choisissez Add permissions (Ajouter des autorisations).

Création d'une nouvelle table avec le dimensionnement automatique d'Amazon Keyspaces activé

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour créer une nouvelle table avec la mise à l'échelle automatique activée

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Tables, puis Create table (Créer une table).
3. Sur la page Créer une table, dans la section Détails de la table, sélectionnez un espace de touche et attribuez un nom à la nouvelle table.
4. Dans la section Colonnes, créez le schéma de votre table.
5. Dans la section Clé primaire, définissez la clé primaire de la table et sélectionnez les colonnes de clustering facultatives.
6. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
7. Continuez jusqu'à Paramètres de capacité en lecture/écriture.
8. Pour le Mode de capacité, choisissez Provisioned (Alloué).
9. Dans la section Capacité de lecture confirmez que l'option Mettre à l'échelle automatiquement est sélectionnée.

   Dans cette étape, vous sélectionnez les unités de capacité de lecture minimale et maximale pour la table, ainsi que l'utilisation cible.

   - Unités de capacité minimale : entrez la valeur du niveau de débit minimal que le tableau doit toujours être prêt à prendre en charge. La valeur doit être comprise entre 1 et le quota par seconde du débit maximal pour votre compte (40 000 par défaut).

- Unités de capacité maximale : entrez le débit maximal que vous souhaitez allouer pour le tableau. La valeur doit être comprise entre 1 et le quota par seconde du débit maximal pour votre compte (40 000 par défaut).
- Utilisation cible — Entrez un taux d'utilisation cible compris entre 20 % et 90 %. Lorsque le trafic dépasse le taux d'utilisation cible défini, la capacité est automatiquement mise à l'échelle. Lorsque le trafic tombe en dessous de la cible définie, il est automatiquement réduit de nouveau.

> **Note**
>
> Pour en savoir plus sur les quotas par défaut de votre compte et sur la façon de les augmenter, reportez-vous à la section Quotas.

10. Dans la section Capacité d'écriture, choisissez les mêmes paramètres que ceux définis à l'étape précédente pour la capacité de lecture, ou configurez les valeurs de capacité manuellement.
11. Choisissez Créer un tableau. Votre table est créée avec les paramètres de mise à l'échelle automatique spécifiés.

Activation du dimensionnement automatique d'Amazon Keyspaces sur les tables existantes

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour activer le dimensionnement automatique d'Amazon Keyspaces pour une table existante

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la table avec laquelle vous souhaitez travailler, puis accédez à l'onglet Capacité.
3. Dans la section Paramètres de capacité, choisissez Modifier.

4. En mode Capacité, assurez-vous que la table utilise le mode Capacité provisionnée.
5. Sélectionnez Mettre à l'échelle automatiquement et consultez l'étape 6 dans Création d'une nouvelle table avec le dimensionnement automatique d'Amazon Keyspaces activé pour modifier la capacité de lecture et d'écriture.
6. Lorsque les paramètres de mise à l'échelle automatique sont définis, choisissez Enregistrer.

Modifier ou désactiver les paramètres de dimensionnement automatique d'Amazon Keyspaces

Vous pouvez utiliser le AWS Management Console pour modifier les paramètres de dimensionnement automatique de vos Amazon Keyspaces. Pour ce faire, choisissez le tableau que vous souhaitez modifier et accédez à l'onglet Capacité. Dans la section Paramètres de capacité, choisissez Modifier. Vous pouvez désormais modifier les paramètres dans les sections Capacité de lecture ou Capacité d'écriture. Pour plus d'informations sur ces paramètres, consultez la page Création d'une nouvelle table avec le dimensionnement automatique d'Amazon Keyspaces activé.

Pour désactiver le dimensionnement automatique d'Amazon Keyspaces, décochez la case Scale automatically. La désactivation du dimensionnement automatique annule l'enregistrement de la table en tant que cible évolutive avec Application Auto Scaling. Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling pour accéder à votre table Amazon Keyspaces, suivez les étapes décrites dans. the section called “Supprimer un rôle lié à un service pour Amazon Keyspaces”

> **Note**
>
> Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

Afficher les activités de dimensionnement automatique d'Amazon Keyspaces sur la console

Vous pouvez surveiller la manière dont le dimensionnement automatique d'Amazon Keyspaces utilise les ressources en utilisant Amazon CloudWatch, qui génère des statistiques relatives à votre utilisation et à vos performances. Suivez les étapes du guide de l'Application Auto Scaling utilisateur pour créer un CloudWatch tableau de bord.

Gestion du dimensionnement automatique d'Amazon Keyspaces avec Cassandra Query Language (CQL)

Pour créer et gérer les paramètres de mise à l'échelle automatique pour les tables Amazon Keyspaces avec Cassandra Query Language (CQL), vous pouvez utiliser. `cqlsh` Cette rubrique donne un aperçu des tâches de dimensionnement automatique que vous pouvez gérer par programmation à l'aide de CQL.

Pour plus d'informations sur les instructions CQL décrites dans cette rubrique, consultezthe section called “Instructions DDL”.

Rubriques

- Avant de commencer
- Création d'une nouvelle table avec mise à l'échelle automatique à l'aide de CQL
- Activer la mise à l'échelle automatique sur une table existante à l'aide de CQL
- Afficher la configuration de mise à l'échelle automatique d'Amazon Keyspaces de votre table à l'aide de CQL
- Désactiver le dimensionnement automatique d'Amazon Keyspaces pour une table à l'aide de CQL

Avant de commencer

Vous devez effectuer les tâches suivantes avant de pouvoir commencer.

Configurer des autorisations

Si vous ne l'avez pas encore fait, vous devez configurer les autorisations appropriées pour que l'utilisateur puisse créer et gérer les paramètres de mise à l'échelle automatique. Dans AWS Identity and Access Management (IAM), la politique AWS gérée `AmazonKeyspacesFullAccess` est requise pour gérer les politiques de dimensionnement d'Amazon Keyspaces. Pour obtenir des instructions complètes, consultez the section called “Avant de commencer : accorder des autorisations aux utilisateurs pour le dimensionnement automatique d'Amazon Keyspaces”.

Configurer **`cqlsh`**

Si ce n'est pas déjà fait, vous devez l'installer et le configurer`cqlsh`. Pour ce faire, suivez les instructions indiquées surthe section called “Utilisation de l’`cqlsh-expansion`”. Vous pouvez ensuite utiliser le AWS CloudShell pour exécuter les commandes décrites dans les sections suivantes.

Création d'une nouvelle table avec mise à l'échelle automatique à l'aide de CQL

Lorsque vous créez une nouvelle table Amazon Keyspaces, vous pouvez activer automatiquement le dimensionnement automatique en fonction de la capacité d'écriture ou de lecture de la table dans le `CREATE TABLE` relevé. Cela permet à Amazon Keyspaces de contacter Application Auto Scaling en votre nom pour enregistrer la table en tant que cible évolutive et ajuster la capacité d'écriture ou de lecture allouée.

Pour plus d'informations sur la façon de créer une table multirégionale et de configurer différents paramètres de mise à l'échelle automatique pour les répliques de tables, consultez. the section called “Création d'une table multirégionale avec des paramètres par défaut (CQL)”

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) pour effectuer des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour configurer les paramètres de mise à l'échelle automatique d'une table par programmation, vous devez utiliser l'`AUTOSCALING_SETTING`instruction qui contient les paramètres de mise à l'échelle automatique d'Amazon Keyspaces. Les paramètres définissent les conditions qui obligent Amazon Keyspaces à ajuster le débit provisionné de votre table, ainsi que les actions facultatives supplémentaires à effectuer. Dans cet exemple, vous définissez les paramètres de mise à l'échelle automatique pour mytable.

Cette politique contient les éléments suivants :

- `AUTOSCALING_SETTINGS`— Spécifie si Amazon Keyspaces est autorisé à ajuster la capacité de débit en votre nom. Les valeurs suivantes sont obligatoires :
  - `provisioned_write_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `provisioned_read_capacity_autoscaling_update`:
    - `minimum_units`

  - `maximum_units`
- `scaling_policy`— Amazon Keyspaces soutient la politique de suivi des cibles. Pour définir la politique de suivi des cibles, vous devez configurer les paramètres suivants.
  - `target_value`— La mise à l'échelle automatique d'Amazon Keyspaces garantit que le rapport entre la capacité consommée et la capacité allouée reste égal ou proche de cette valeur. Vous définissez `target_value` en tant que pourcentage.
  - `disableScaleIn`: (Facultatif) Un `boolean` qui indique si la table `scale-in` est désactivée ou activée. Ce paramètre est désactivé par défaut. Pour l'activer`scale-in`, définissez la `boolean` valeur sur`FALSE`. Cela signifie que la capacité est automatiquement réduite pour une table en votre nom.
  - `scale_out_cooldown`— Une activité de scale-out augmente le débit provisionné de votre table. Pour ajouter un temps de stabilisation pour les activités de montée en charge, spécifiez une valeur, en secondes, pour `scale_out_cooldown`. Si vous ne spécifiez aucune valeur, la valeur par défaut est 0. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.
  - `scale_in_cooldown`— Une activité de scale-in réduit le débit provisionné de votre table. Pour ajouter un temps de stabilisation pour les activités de mise à l'échelle, spécifiez une valeur, en secondes, pour `scale_in_cooldown`. Si vous ne spécifiez aucune valeur, la valeur par défaut est 0. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.

> **Note**
>
> Pour mieux comprendre le fonctionnement de `target_value`, supposons que vous ayez une table avec un paramètre de débit approvisionné de 200 unités de capacité d'écriture. Vous décidez de créer une politique de mise à l'échelle pour cette table, avec une `target_value` de 70 pour cent.
> Supposons maintenant que vous commencez à générer du trafic d'écriture vers la table de telle sorte que le débit d'écriture réel est de 150 unités de capacité. Le consumed-to-provisioned ratio est maintenant de (150/200), soit 75 %. Ce ratio étant supérieur à votre objectif, le dimensionnement automatique augmente la capacité d'écriture allouée à 215, de

sorte que le ratio soit (150/ 215), soit 69,77 %, le plus proche `target_value` possible de la vôtre, mais sans le dépasser.

Pour mytable, vous avez défini `TargetValue` une capacité de lecture et d'écriture de 50 %. Le dimensionnement automatique d'Amazon Keyspaces ajuste le débit provisionné de la table dans une fourchette de 5 à 10 unités de capacité afin que le consumed-to-provisioned ratio reste égal ou proche de 50 %. Pour la capacité de lecture, vous définissez les valeurs `ScaleInCooldown` pour `ScaleOutCooldown` et sur 60 secondes.

Vous pouvez utiliser l'instruction suivante pour créer une nouvelle table Amazon Keyspaces avec la mise à l'échelle automatique activée.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    }
```

```
};
```

Activer la mise à l'échelle automatique sur une table existante à l'aide de CQL

Pour une table Amazon Keyspaces existante, vous pouvez activer le dimensionnement automatique en fonction de la capacité d'écriture ou de lecture de la table à l'aide de l'`ALTER TABLE`instruction. Si vous mettez à jour une table qui est actuellement en mode capacité à la demande, cela `capacity_mode` est nécessaire. Si votre table est déjà en mode capacité provisionnée, ce champ peut être omis.

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Dans l'exemple suivant, l'instruction met à jour la table mytable, qui est en mode capacité à la demande. L'instruction fait passer le mode de capacité de la table en mode provisionné avec le dimensionnement automatique activé.

La capacité d'écriture est configurée dans une plage de 5 à 10 unités de capacité avec une valeur cible de 50 %. La capacité de lecture est également configurée dans une plage de 5 à 10 unités de capacité avec une valeur cible de 50 %. Pour la capacité de lecture, vous définissez les valeurs `scale_in_cooldown` pour `scale_out_cooldown` et sur 60 secondes.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
```

```
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    }
};
```

Afficher la configuration de mise à l'échelle automatique d'Amazon Keyspaces de votre table à l'aide de CQL

Pour afficher les détails de la configuration de mise à l'échelle automatique d'une table, utilisez la commande suivante.

```
SELECT * FROM system_schema_mcs.autoscaling WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

La sortie de cette commande ressemble à ceci :

```
 keyspace_name | table_name | provisioned_read_capacity_autoscaling_update

                                                                                |
 provisioned_write_capacity_autoscaling_update
---------------+------------
+-------------------------------------------------------------------------------------------------
+-------------------------------------------------------------------------------------------------
 mykeyspace    | mytable    | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 60, 'disable_scale_in': false, 'target_value':
 50, 'scale_in_cooldown': 60}}} | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
```

```
{'scale_out_cooldown': 0, 'disable_scale_in': false, 'target_value': 50,
'scale_in_cooldown': 0}}}
```

Désactiver le dimensionnement automatique d'Amazon Keyspaces pour une table à l'aide de CQL

Vous pouvez désactiver le dimensionnement automatique d'Amazon Keyspaces pour votre table à tout moment. Si vous n'avez plus besoin de redimensionner la capacité de lecture ou d'écriture de votre tableau, vous devriez envisager de désactiver le dimensionnement automatique afin qu'Amazon Keyspaces ne continue pas à modifier les paramètres de capacité de lecture ou d'écriture de votre tableau. Vous pouvez mettre à jour le tableau à l'aide d'une `ALTER TABLE` instruction.

L'instruction suivante désactive la mise à l'échelle automatique de la capacité d'écriture de la table mytable. Il supprime également les CloudWatch alarmes créées en votre nom.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

Gestion des politiques de dimensionnement d'Amazon Keyspaces à l'aide de la CLI

Pour mettre à jour et gérer les paramètres de dimensionnement automatique d'Amazon Keyspaces par programmation, vous pouvez utiliser le AWS Command Line Interface (AWS CLI) ou l'API. AWS Pour gérer les politiques de dimensionnement automatique d'Amazon Keyspaces à l'aide du langage de requête Cassandra (CQL), consultez. the section called “Utilisation de CQL” Cette rubrique donne un aperçu des tâches de dimensionnement automatique que vous pouvez gérer par programmation à l'aide du. AWS CLI

Pour plus d'informations sur les AWS CLI commandes Amazon Keyspaces décrites dans cette rubrique, consultez le AWS CLI Command Reference.

Rubriques

Avant de commencer

Vous devez effectuer les tâches suivantes avant de pouvoir commencer.

Configurer des autorisations

Si vous ne l'avez pas encore fait, vous devez configurer les autorisations appropriées pour que l'utilisateur puisse créer et gérer les paramètres de mise à l'échelle automatique. Dans AWS Identity and Access Management (IAM), la politique AWS gérée `AmazonKeyspacesFullAccess` est requise pour gérer les politiques de dimensionnement d'Amazon Keyspaces. Pour obtenir des instructions complètes, consultez the section called “Avant de commencer : accorder des autorisations aux utilisateurs pour le dimensionnement automatique d'Amazon Keyspaces”.

Installer le AWS CLI

Si vous ne l'avez pas déjà fait, vous devez installer et configurer AWS CLI. Pour ce faire, consultez le guide de l' AWS Command Line Interface utilisateur et suivez les instructions suivantes :

- Installation de AWS CLI
- Configuration de l'interface AWS CLI (français non garanti)

Créez un nouveau tableau avec mise à l'échelle automatique à l'aide du AWS CLI

Lorsque vous créez une nouvelle table Amazon Keyspaces, vous pouvez activer automatiquement le dimensionnement automatique en fonction de la capacité d'écriture ou de lecture de la table lors de

l'`CreateTable`opération. Cela permet à Amazon Keyspaces de contacter Application Auto Scaling en votre nom pour enregistrer la table que vous spécifiez en tant que cible évolutive et ajuster la capacité d'écriture ou de lecture allouée.

Pour plus d'informations sur la création d'une table multirégionale avec une configuration de mise à l'échelle automatique, consultezthe section called “Création d'une nouvelle table multirégionale en mode provisionné avec mise à l'échelle automatique (CLI)”.

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) pour effectuer des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour configurer les paramètres de dimensionnement automatique d'une table par programmation, vous devez utiliser l'`autoScalingSpecification`action qui définit les paramètres du dimensionnement automatique d'Amazon Keyspaces. Les paramètres définissent les conditions qui obligent Amazon Keyspaces à ajuster le débit provisionné de votre table, ainsi que les actions facultatives supplémentaires à effectuer. Dans cet exemple, vous définissez les paramètres de mise à l'échelle automatique pour mytable.

Cette politique contient les éléments suivants :

- `autoScalingSpecification`— Spécifie si Amazon Keyspaces est autorisé à ajuster la capacité et le débit en votre nom. Vous pouvez activer la mise à l'échelle automatique pour la capacité de lecture et pour la capacité d'écriture séparément. Vous devez ensuite définir les paramètres suivants pour `autoScalingSpecification` :
  - `writeCapacityAutoScaling`— Les unités de capacité d'écriture maximale et minimale.
  - `readCapacityAutoScaling`— Les unités de capacité de lecture maximale et minimale.
  - `scalingPolicy`— Amazon Keyspaces soutient la politique de suivi des cibles. Pour définir la politique de suivi des cibles, vous devez configurer les paramètres suivants.
    - `targetValue`— La mise à l'échelle automatique d'Amazon Keyspaces garantit que le rapport entre la capacité consommée et la capacité allouée reste égal ou proche de cette valeur. Vous définissez `targetValue` en tant que pourcentage.

- `disableScaleIn`: (Facultatif) Un `boolean` qui indique si la table `scale-in` est désactivée ou activée. Ce paramètre est désactivé par défaut. Pour l'activer`scale-in`, définissez la `boolean` valeur sur`FALSE`. Cela signifie que la capacité est automatiquement réduite pour une table en votre nom.
- `scaleOutCooldown`— Une activité de scale-out augmente le débit provisionné de votre table. Pour ajouter un temps de stabilisation pour les activités de montée en charge, spécifiez une valeur, en secondes, pour `ScaleOutCooldown`. La valeur par défaut est 0. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.
- `scaleInCooldown`— Une activité de scale-in réduit le débit provisionné de votre table. Pour ajouter un temps de stabilisation pour les activités de mise à l'échelle, spécifiez une valeur, en secondes, pour `ScaleInCooldown`. La valeur par défaut est 0. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.

> **Note**
>
> Pour mieux comprendre le fonctionnement de `TargetValue`, supposons que vous ayez une table avec un paramètre de débit approvisionné de 200 unités de capacité d'écriture. Vous décidez de créer une politique de mise à l'échelle pour cette table, avec une `TargetValue` de 70 pour cent.
> Supposons maintenant que vous commencez à générer du trafic d'écriture vers la table de telle sorte que le débit d'écriture réel est de 150 unités de capacité. Le consumed-to-provisioned ratio est maintenant de (150/200), soit 75 %. Ce ratio étant supérieur à votre objectif, le dimensionnement automatique augmente la capacité d'écriture allouée à 215, de sorte que le ratio soit (150/ 215), soit 69,77 %, le plus proche `TargetValue` possible de la vôtre, mais sans le dépasser.

Pour mytable, vous avez défini `TargetValue` une capacité de lecture et d'écriture de 50 %. Le dimensionnement automatique d'Amazon Keyspaces ajuste le débit provisionné de la table dans une fourchette de 5 à 10 unités de capacité afin que le consumed-to-provisioned ratio reste égal ou proche de 50 %. Pour la capacité de lecture, vous définissez les valeurs `ScaleInCooldown` pour `ScaleOutCooldown` et sur 60 secondes.

Lorsque vous créez des tables avec des paramètres de mise à l'échelle automatique complexes, il est utile de charger les paramètres de mise à l'échelle automatique à partir d'un fichier JSON. Dans l'exemple suivant, vous pouvez télécharger le fichier JSON d'exemple depuis auto-scaling.zip et l'extraire `auto-scaling.json` en prenant note du chemin d'accès au fichier. Dans cet exemple, le fichier JSON se trouve dans le répertoire actuel. Pour connaître les différentes options de chemin de fichier, consultez Comment charger des paramètres à partir d'un fichier.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
            \ --schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=int}],partitionKeys=[{name=pk},{name=ck}]'
            \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
            \ --auto-scaling-specification file://auto-scaling.json
```

Activez la mise à l'échelle automatique sur une table existante à l'aide du AWS CLI

Pour une table Amazon Keyspaces existante, vous pouvez activer le dimensionnement automatique pour la capacité d'écriture ou de lecture de la table à l'`UpdateTable`aide de cette opération. Pour plus d'informations sur la mise à jour des paramètres de mise à l'échelle automatique pour une table multirégionale, consultezthe section called “Mise à jour de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CLI)”.

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Vous pouvez utiliser la commande suivante pour activer le dimensionnement automatique d'Amazon Keyspaces pour une table existante. Les paramètres de mise à l'échelle automatique de la table sont chargés à partir d'un fichier JSON. Dans l'exemple suivant, vous pouvez télécharger le fichier JSON d'exemple depuis auto-scaling.zip et l'extraire `auto-scaling.json` en prenant note du chemin d'accès au fichier. Dans cet exemple, le fichier JSON se trouve dans le répertoire actuel. Pour connaître les différentes options de chemin de fichier, consultez Comment charger des paramètres à partir d'un fichier.

Pour plus d'informations sur les paramètres de mise à l'échelle automatique utilisés dans l'exemple suivant, consultezthe section called “Créez un nouveau tableau avec mise à l'échelle automatique à l'aide du AWS CLI”.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
           \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
           \ --auto-scaling-specification file://auto-scaling.json
```

Consultez la configuration de mise à l'échelle automatique d'Amazon Keyspaces de votre table à l'aide du AWS CLI

Pour afficher la configuration de mise à l'échelle automatique d'une table, vous pouvez utiliser l'`get-table-auto-scaling-settings`opération. La commande CLI suivante en est un exemple.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

La sortie de cette commande ressemble à ceci :

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:5555-5555-5555:/keyspace/mykeyspace/
table/mytable",
    "autoScalingSpecification": {
        "writeCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 0,
                    "scaleOutCooldown": 0,
                    "targetValue": 50.0
                }
            }
        },
        "readCapacityAutoScaling": {
            "autoScalingDisabled": false,
```

```
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 60,
                    "scaleOutCooldown": 60,
                    "targetValue": 50.0
                }
            }
        }
    }
}
```

Désactivez la mise à l'échelle automatique d'Amazon Keyspaces pour un tableau à l'aide du AWS CLI

Vous pouvez désactiver le dimensionnement automatique d'Amazon Keyspaces pour votre table à tout moment. Si vous n'avez plus besoin de redimensionner la capacité de lecture ou d'écriture de votre tableau, vous devriez envisager de désactiver le dimensionnement automatique afin qu'Amazon Keyspaces ne continue pas à modifier les paramètres de capacité de lecture ou d'écriture de votre tableau. Vous pouvez mettre à jour le tableau par une `UpdateTable` opération.

La commande suivante désactive le dimensionnement automatique en fonction de la capacité de lecture de la table. Il supprime également les CloudWatch alarmes créées en votre nom.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
            \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

Utilisation efficace de la capacité en rafale dans Amazon Keyspaces

Amazon Keyspaces offre une certaine flexibilité dans le provisionnement du débit par partition en fournissant une capacité de pointe. Lorsque vous n'utilisez pas pleinement le débit d'une partition, Amazon Keyspaces réserve une partie de cette capacité inutilisée pour des pics de débit ultérieurs afin de gérer les pics d'utilisation.

Amazon Keyspaces conserve actuellement jusqu'à 5 minutes (300 secondes) de capacité de lecture et d'écriture inutilisée. Lors d'une vague occasionnelle d'activité de lecture ou d'écriture, ces unités de capacité supplémentaire peuvent être consommées rapidement, même plus rapidement que la capacité de débit allouée par seconde que vous avez définie pour votre table.

Amazon Keyspaces peut également consommer de la capacité en rafale pour la maintenance en arrière-plan et d'autres tâches sans préavis.

Notez que ces informations relatives à la capacité de transmission en mode rafale sont susceptibles d'évoluer.

Comment estimer la consommation de capacité dans Amazon Keyspaces

Lorsque vous lisez ou écrivez des données dans Amazon Keyspaces, la quantité d'unités de demande de lecture/écriture (RRUS/WRU) ou d'unités de capacité de lecture/écriture (RCU/WCU) consommées par votre requête dépend de la quantité totale de données qu'Amazon Keyspaces doit traiter pour exécuter la requête. Dans certains cas, les données renvoyées au client peuvent être un sous-ensemble des données qu'Amazon Keyspaces a dû lire pour traiter la requête. Pour les écritures conditionnelles, Amazon Keyspaces consomme de la capacité d'écriture même si le contrôle conditionnel échoue.

Pour estimer la quantité totale de données traitées pour une demande, vous devez tenir compte de la taille codée d'une ligne et du nombre total de lignes. Cette rubrique présente quelques exemples de scénarios et de modèles d'accès courants pour montrer comment Amazon Keyspaces traite les requêtes et comment cela affecte la consommation de capacité. Vous pouvez suivre les exemples pour estimer les besoins en capacité de vos tables et utiliser Amazon CloudWatch pour observer la consommation de capacité de lecture et d'écriture pour ces cas d'utilisation.

Pour plus d'informations sur le calcul de la taille codée des lignes dans Amazon Keyspaces, consultez. the section called “Calcul de la taille des lignes”

Rubriques

Requêtes de plage

Pour examiner la consommation de capacité de lecture d'une requête de plage, nous utilisons le tableau d'exemple suivant qui utilise le mode capacité à la demande.

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 50 | <any value that results in a row size larger than 4KB>
a | b | 1 | a | b | 60 | value_1
a | b | 1 | a | b | 70 | <any value that results in a row size larger than 4KB>
```

Exécutez maintenant la requête suivante sur cette table.

```
SELECT * FROM amazon_keyspaces.example_table_1 WHERE pk1='a' AND pk2='b' AND pk3=1 AND
 ck1='a' AND ck2='b' AND ck3 > 50 AND ck3 < 70;
```

Vous recevez le jeu de résultats suivant à partir de la requête et l'opération de lecture effectuée par Amazon Keyspaces consomme 2 RRU en `LOCAL_QUORUM` mode cohérence.

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 60 | value_1
```

Amazon Keyspaces utilise 2 RRU pour évaluer les lignes contenant les valeurs `ck3=60` et `ck3=70` pour traiter la requête. Cependant, Amazon Keyspaces renvoie uniquement la ligne pour laquelle la `WHERE` condition spécifiée dans la requête est vraie, c'est-à-dire la ligne contenant une valeur. `ck3=60` Pour évaluer la plage spécifiée dans la requête, Amazon Keyspaces lit la ligne correspondant à la limite supérieure de la plage, dans ce cas`ck3 = 70`, mais ne renvoie pas cette

ligne dans le résultat. La consommation de capacité de lecture est basée sur les données lues lors du traitement de la requête, et non sur les données renvoyées.

Limiter les requêtes

Lors du traitement d'une requête qui utilise la `LIMIT` clause, Amazon Keyspaces lit les lignes jusqu'à la taille de page maximale en essayant de répondre à la condition spécifiée dans la requête. Si Amazon Keyspaces ne trouve pas suffisamment de données correspondantes correspondant à la `LIMIT` valeur de la première page, un ou plusieurs appels paginés peuvent être nécessaires. Pour poursuivre les lectures sur la page suivante, vous pouvez utiliser un jeton de pagination. La taille de page par défaut est de 1 Mo. Pour réduire la capacité de lecture lorsque vous utilisez `LIMIT` des clauses, vous pouvez réduire la taille de page. Pour plus d'informations sur la pagination, consultezthe section called “Pagination des résultats”.

À titre d'exemple, examinons la requête suivante.

```
SELECT * FROM my_table WHERE partition_key=1234 LIMIT 1;”
```

Si vous ne définissez pas le format de page, Amazon Keyspaces lit 1 Mo de données même s'il ne vous renvoie qu'une seule ligne. Pour qu'Amazon Keyspaces ne lise qu'une seule ligne, vous pouvez définir le format de page sur 1 pour cette requête. Dans ce cas, Amazon Keyspaces ne lira qu'une seule ligne, à condition que vous n'ayez pas de lignes expirées en fonction des ime-to-live paramètres T ou des horodatages côté client. Pour réduire la capacité de lecture, nous vous recommandons de définir une taille de page égale à la `LIMIT` valeur afin de réduire la quantité de données lues par Amazon Keyspaces.

Numérisations de tableaux

Les requêtes qui aboutissent à une analyse complète de la table, par exemple les requêtes utilisant l'`ALLOW FILTERING`option, sont un autre exemple de requêtes qui traitent plus de lectures que ce qu'elles renvoient sous forme de résultats. Et la consommation de capacité de lecture est basée sur les données lues, et non sur les données renvoyées.

Pour l'exemple de numérisation de tables, nous utilisons l'exemple de tableau suivant en mode capacité à la demande.

```
pk | ck | value
---+----+---------
pk | 10 | <any value that results in a row size larger than 4KB>
pk | 20 | value_1
```

```
pk | 30 | <any value that results in a row size larger than 4KB>
```

Amazon Keyspaces crée une table en mode capacité à la demande avec quatre partitions par défaut. Dans cet exemple de tableau, toutes les données sont stockées dans une partition et les trois partitions restantes sont vides.

Exécutez maintenant la requête suivante sur la table.

```
SELECT * from amazon_keyspaces.example_table_2;
```

Cette requête entraîne une opération d'analyse de table au cours de laquelle Amazon Keyspaces analyse les quatre partitions de la table et consomme 6 RRU en `LOCAL_QUORUM` mode de cohérence. Tout d'abord, Amazon Keyspaces consomme 3 RRU pour lire les trois lignes avec. `pk='pk'` Amazon Keyspaces consomme ensuite les 3 RRU supplémentaires pour scanner les trois partitions vides de la table. Comme cette requête entraîne une analyse de table, Amazon Keyspaces analyse toutes les partitions de la table, y compris les partitions ne contenant pas de données.

Transactions légères

Les transactions légères (LWT) vous permettent d'effectuer des opérations d'écriture conditionnelle sur les données de votre table. Les opérations de mise à jour conditionnelle sont utiles lors de l'insertion, de la mise à jour et de la suppression d'enregistrements en fonction de conditions évaluant l'état actuel.

Dans Amazon Keyspaces, toutes les opérations d'écriture nécessitent la cohérence LOCAL_QUORUM et l'utilisation des LWT est gratuite. La différence pour les LWT est que lorsqu'une vérification de condition LWT aboutit à la valeur FALSE, elle consomme des unités de capacité d'écriture. Le nombre d'unités de capacité d'écriture consommées dépend de la taille de la ligne. Si la taille de ligne est de 2 Ko, l'écriture conditionnelle échouée consomme deux unités de capacité d'écriture. Si la ligne n'existe pas actuellement dans la table, l'opération consomme une unité de capacité d'écriture. En surveillant la `ConditionalCheckFailed` métrique, CloudWatch vous pouvez déterminer la capacité consommée par les défaillances du contrôle d'état du LWT.

Estimez la consommation de capacité de lecture et d'écriture avec Amazon CloudWatch

Pour estimer et surveiller la consommation de capacité de lecture et d'écriture, vous pouvez utiliser un CloudWatch tableau de bord. Pour plus d'informations sur les statistiques disponibles pour Amazon Keyspaces, consultez. the section called “Métriques et dimensions”

Pour surveiller les unités de capacité de lecture et d'écriture consommées par une instruction spécifique avec CloudWatch, vous pouvez suivre ces étapes.

1. Création d'une nouvelle table avec des exemples de données
2. Configurez un tableau de bord Amazon Keyspaces pour le CloudWatch tableau. Pour commencer, vous pouvez utiliser un modèle de tableau de bord disponible sur Github.
3. Exécutez l'instruction CQL, par exemple à l'aide de l'`ALLOW FILTERING`option, et vérifiez les unités de capacité de lecture consommées pour l'analyse complète du tableau dans le tableau de bord.

Utilisation d'espaces de touches, de tables et de lignes dans Amazon Keyspaces (pour Apache Cassandra)

Ce chapitre fournit des informations sur l'utilisation des espaces de touches, des tables, des lignes, etc. dans Amazon Keyspaces (pour Apache Cassandra). Pour savoir comment surveiller les espaces de touches et les tables avec Amazon CloudWatch, consultezthe section called “Surveillance avec CloudWatch”.

Rubriques

Utilisation des espaces de saisie dans Amazon Keyspaces

Cette section fournit des informations sur l'utilisation des espaces de touches dans Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Utilisation des espaces de touches du système dans Amazon Keyspaces

Amazon Keyspaces utilise quatre espaces de touches système :

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

Les sections suivantes fournissent des informations sur les espaces de touches du système et les tables système pris en charge dans Amazon Keyspaces.

`system`

Il s'agit d'un keyspace de Cassandra. Amazon Keyspaces utilise les tableaux suivants.

Noms des tables	Noms de colonnes	Commentaires
`local`	`key, bootstrapped, broadcast_address, cluster_name, cql_version, data_center, gossip_generation, host_id, listen_address, native_protocol_version, partitioner, rack, release_version, rpc_address, schema_version, thrift_version, tokens, truncated_at`	Informations sur le keyspace local.
`peers`	`peer, data_center, host_id, preferred_ip, rack, release_version, rpc_address, schema_version, tokens`	Consultez ce tableau pour voir les points de terminaison disponibles. Par exemple, si vous vous connectez via un point de terminaison public, une liste de neuf adresses IP disponibles s'affiche. Si vous vous connectez via un point de terminaison FIPS, une liste de trois adresses IP s'affiche. Si vous vous connectez via un point de terminaison AWS PrivateLink VPC, la liste des

Noms des tables	Noms de colonnes	Commentaires
		adresses IP que vous avez configurées s'affiche. Pour de plus amples informations, veuillez consulter the section called “Remplissage des entrées de `system.peers` table avec les informations de point de terminaison VPC de l'interface”.
`size_estimates`	`keyspace_name, table_name, range_start, range_end, mean_partition_size, partitions_count`	Ce tableau définit la taille totale et le nombre de partitions pour chaque plage de jetons pour chaque table. Cela est nécessaire pour le connecteur Apache Cassandra Spark, qui utilise la taille de partition estimée pour distribuer le travail.
`prepared_statements`	`prepared_id, logged_keyspace, query_string`	Ce tableau contient des informations sur les requêtes enregistrées.

`system_schema`

Il s'agit d'un keyspace de Cassandra. Amazon Keyspaces utilise les tableaux suivants.

Noms des tables	Noms de colonnes	Commentaires
`keyspaces`	`keyspace_name, durable_writes, replication`	Informations relatives à un keyspace spécifique.

Noms des tables	Noms de colonnes	Commentaires
`tables`	`keyspace_name, table_name, bloom_filter_fp_chance, caching, comment, compaction, compression, crc_check_chance, dclocal_read_repair_chance, default_time_to_live, extensions, flags, gc_grace_seconds, id, max_index_interval, memtable_flush_period_in_ms, min_index_interval, read_repair_chance, speculative_retry`	Informations relatives à une table spécifique.
`columns`	`keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type`	Informations relatives à une colonne spécifique.

`system_schema_mcs`

Il s'agit d'un espace de touches Amazon Keyspaces qui stocke des informations ou des paramètres spécifiques à AWS Amazon Keyspaces.

Noms des tables	Noms de colonnes	Commentaires
`keyspaces`	`keyspace_name, durable_writes, replication`	Interrogez cette table pour savoir par programmation si un espace de touches a été créé. Pour de plus amples informations, veuillez consulter the section called “Création d'espaces clés”.
`tables`	`keyspace_name, creation_time, speculative_retry, cdc, gc_grace_seconds, crc_check_chance, min_index_interval, bloom_filter_fp_chance, flags, custom_properties, dclocal_read_repair_chance, table_name, caching, default_time_to_live, read_repair_chance, max_index_interval, extensions, compaction, comment, id, compression, memtable_flush_period_in_ms, status`	Interrogez cette table pour connaître le statut d'une table spécifique. Pour de plus amples informations, veuillez consulter the section called “Création de tables”. Vous pouvez également consulter ce tableau pour répertorier les paramètres spécifiques à Amazon Keyspaces et stockés sous le nom de. `custom_properties` Par exemple : • `capacity_mode` • `client_side_timestamps` • `encryption_specification` • `point_in_time_recovery` • `ttl`
`tables_history`	`keyspace_name, table_name, event_time, creation_time,`	Interrogez cette table pour en savoir plus sur les modificat

Noms des tables	Noms de colonnes	Commentaires
	`custom_properties, event`	ions apportées au schéma d'une table spécifique.
`columns`	`keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type`	Cette table est identique à la table Cassandra dans le `system_schema` keyspace.
`tags`	`resource_id, keyspace_name, resource_name, resource_type, tags`	Interrogez ce tableau pour savoir si un espace de touches contient des balises. Pour de plus amples informations, veuillez consulter the section called “Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide du CQL”.
`autoscaling`	`keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update`	Interrogez cette table pour obtenir les paramètres de dimensionnement automatique d'une table provisionnée. Notez que ces paramètres ne seront pas disponibles tant que le tableau ne sera pas actif. Pour interroger cette table, vous devez spécifier `keyspace_name` et `table_name` dans la `WHERE` clause. Pour de plus amples informations, veuillez consulter the section called “Utilisation de CQL”.

system_multiregion_info

Il s'agit d'un espace de touches Amazon Keyspaces qui stocke des informations sur la réplication multirégionale.

Noms des tables	Noms de colonnes	Commentaires
tables	keyspace_name, table_name, region, status	Ce tableau contient des informations sur les tables multirégionales, par exemple les informations dans Régions AWS lesquelles la table est répliquée et le statut de la table. Vous pouvez également consulter ce tableau pour répertorier les paramètre s spécifiques à Amazon Keyspaces stockés sous le nom de. custom_pr operties Par exemple : • capacity_mode Pour interroger cette table, vous devez spécifier keyspace_name et table_name dans la WHERE clause. Pour de plus amples informations, veuillez consulter the section called “Création d'un keyspace multirégional (CQL)”.
autoscaling	keyspace_name, table_name, provision ed_read_capacity_a utoscaling_update,	Interrogez cette table pour obtenir les paramètres de dimensionnement automatiq ue d'une table provisionnée

Noms des tables	Noms de colonnes	Commentaires
	`provisioned_write_capacity_autoscaling_update, region`	multirégionale. Notez que ces paramètres ne seront pas disponibles tant que le tableau ne sera pas actif. Pour interroger cette table, vous devez spécifier `keyspace_name` et `table_name` dans la `WHERE` clause. Pour de plus amples informations, veuillez consulter the section called “Utilisation de CQL”.

Création d'espaces de touches dans Amazon Keyspaces

Amazon Keyspaces exécute des opérations de langage de définition de données (DDL), telles que la création et la suppression d'espaces clés, de manière asynchrone.

Vous pouvez surveiller l'état de création de nouveaux espaces clés dans le AWS Management Console, ce qui indique lorsqu'un espace clé est en attente ou actif. Vous pouvez également contrôler l'état de création d'un nouvel espace de touches par programmation à l'aide de l'espace de touches. `system_schema_mcs` Un espace de touche devient visible dans le `system_schema_mcs` `keyspaces` tableau lorsqu'il est prêt à être utilisé.

Le modèle de conception recommandé pour vérifier qu'un nouvel espace de touches est prêt à être utilisé consiste à interroger la table Amazon `system_schema_mcs` `keyspaces` Keyspaces (system_schema_mcs.*). Pour obtenir la liste des instructions DDL pour les espaces clés, consultez la the section called “Keyspaces” section du manuel de référence du langage CQL.

La requête suivante indique si un espace de touches a été créé avec succès.

```
SELECT * FROM system_schema_mcs.keyspaces WHERE keyspace_name = 'mykeyspace';
```

Pour un espace de touches créé avec succès, le résultat de la requête ressemble à ce qui suit.

```
keyspace_name | durable_writes  | replication
```

```
--------------+-----------------+--------------
   mykeyspace | true            |{...} 1 item
```

Utilisation de tables dans Amazon Keyspaces

Cette section fournit des informations sur l'utilisation des tables dans Amazon Keyspaces (pour Apache Cassandra).

Rubriques

- Création de tables dans Amazon Keyspaces
- Utilisation de tables multirégionales dans Amazon Keyspaces
- Colonnes statiques dans Amazon Keyspaces

Création de tables dans Amazon Keyspaces

Amazon Keyspaces exécute des opérations de langage de définition de données (DDL), telles que la création et la suppression de tables, de manière asynchrone. Vous pouvez surveiller l'état de création de nouvelles tables dans le AWS Management Console, qui indique si une table est en attente ou active. Vous pouvez également surveiller l'état de création d'une nouvelle table par programmation à l'aide de la table du schéma système.

Un tableau apparaît comme actif dans le schéma du système lorsqu'il est prêt à être utilisé. Le modèle de conception recommandé pour vérifier qu'une nouvelle table est prête à être utilisée consiste à interroger les tables de schéma du système Amazon Keyspaces ()`system_schema_mcs.*`. Pour obtenir la liste des instructions DDL pour les tables, consultez la the section called “Tables” section du manuel de référence du langage CQL.

La requête suivante indique le statut d'une table.

```
SELECT keyspace_name, table_name, status FROM system_schema_mcs.tables WHERE
 keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Pour une table toujours en cours de création et en attente, le résultat de la requête ressemble à ceci.

```
keyspace_name | table_name | status
```

```
--------------+------------+--------
   mykeyspace |    mytable | CREATING
```

Pour une table qui a été créée avec succès et qui est active, le résultat de la requête est le suivant.

```
keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | ACTIVE
```

Utilisation de tables multirégionales dans Amazon Keyspaces

La capacité de débit d'écriture d'une table multirégionale doit être configurée de deux manières :

- Mode de capacité à la demande, mesuré en unités de demande d'écriture (WRU)
- Mode capacité provisionnée avec mise à l'échelle automatique, mesurée en unités de capacité d'écriture (WCU)

Vous pouvez utiliser le mode capacité provisionnée avec mise à l'échelle automatique ou le mode capacité à la demande pour garantir qu'une table multirégionale dispose d'une capacité suffisante pour effectuer des écritures répliquées pour toutes les tables. Régions AWS

> **Note**
>
> La modification du mode de capacité de la table dans l'une des régions modifie le mode de capacité de toutes les répliques.

Par défaut, Amazon Keyspaces utilise le mode à la demande pour les tables multirégionales. Avec le mode à la demande, vous n'avez pas besoin de spécifier le débit de lecture et d'écriture que vous souhaitez que votre application exécute. Amazon Keyspaces s'adapte instantanément à vos charges de travail lorsqu'elles augmentent ou diminuent pour atteindre un niveau de trafic atteint précédemment. Si le niveau de trafic d'une charge de travail atteint un nouveau pic, Amazon Keyspaces s'adapte rapidement pour s'adapter à la charge de travail.

Si vous choisissez le mode de capacité provisionnée pour une table, vous devez configurer le nombre d'unités de capacité de lecture (RCU) et d'unités de capacité d'écriture (WCU) par seconde dont votre application a besoin.

Pour planifier les besoins en capacité de débit d'une table multirégionale, vous devez d'abord estimer le nombre de WCU nécessaires par seconde pour chaque région. Vous ajoutez ensuite les écritures provenant de toutes les régions dans lesquelles votre table est répliquée et vous utilisez la somme pour allouer de la capacité à chaque région. Cela est nécessaire car chaque écriture effectuée dans une région doit également être répétée dans chaque région de réplication.

Si la table ne dispose pas d'une capacité suffisante pour gérer les écritures provenant de toutes les régions, des exceptions de capacité se produiront. En outre, les temps d'attente pour la réplication interrégionale augmenteront.

Par exemple, si vous avez une table multirégionale dans laquelle vous attendez 5 écritures par seconde dans l'est des États-Unis (Virginie du Nord), 10 écritures par seconde dans l'est des États-Unis (Ohio) et 5 écritures par seconde en Europe (Irlande), vous devez vous attendre à ce que la table consomme 20 unités WCU dans chaque région : États-Unis Est (Virginie du Nord), États-Unis Est (Ohio) et Europe (Irlande). Cela signifie que dans cet exemple, vous devez provisionner 20 WCU pour chacune des répliques de la table. Vous pouvez surveiller la consommation de capacité de votre table à l'aide d'Amazon CloudWatch. Pour de plus amples informations, veuillez consulter the section called “Surveillance avec CloudWatch”.

Comme chaque écriture multirégionale est facturée 1,25 fois les WCU, vous verrez un total de 75 WCU facturées dans cet exemple. Pour plus d'informations sur les tarifs, consultez les tarifs d'Amazon Keyspaces (pour Apache Cassandra).

Pour plus d'informations sur la capacité allouée avec le dimensionnement the section called “Gérez la capacité de débit grâce à la mise à l'échelle automatique” automatique d'Amazon Keyspaces, consultez.

> **Note**
>
> Si une table s'exécute en mode capacité allouée avec mise à l'échelle automatique, la capacité d'écriture allouée est autorisée à flotter dans les limites de ces paramètres de dimensionnement automatique pour chaque région.

Colonnes statiques dans Amazon Keyspaces

Dans un tableau Amazon Keyspaces avec des colonnes de regroupement, vous pouvez utiliser le `STATIC` mot clé pour créer une colonne statique. La valeur stockée dans une colonne statique est partagée entre toutes les lignes d'une partition logique. Lorsque vous mettez à jour la valeur de cette

colonne, Amazon Keyspaces applique la modification automatiquement à toutes les lignes de la partition.

Cette section explique comment calculer la taille codée des données lorsque vous écrivez dans des colonnes statiques. Ce processus est géré séparément du processus qui écrit des données dans les colonnes non statiques d'une ligne. Outre les quotas de taille pour les données statiques, les opérations de lecture et d'écriture sur des colonnes statiques affectent également la capacité de mesure et de débit des tables indépendamment.

Calcul de la taille de colonne statique par partition logique dans Amazon Keyspaces

Cette section explique comment estimer la taille codée des colonnes statiques dans Amazon Keyspaces. La taille codée est utilisée lorsque vous calculez votre facture et votre quota d'utilisation. Vous devez également utiliser la taille codée lorsque vous calculez les exigences de capacité de débit allouées pour les tables. Pour calculer la taille codée des colonnes statiques dans Amazon Keyspaces, vous pouvez suivre les directives suivantes.

- Les clés de partition peuvent contenir jusqu'à 2 048 octets de données. Chaque colonne clé de la clé de partition nécessite jusqu'à 3 octets de métadonnées. Ces octets de métadonnées sont pris en compte dans votre quota de taille de données statiques de 1 Mo par partition. Lorsque vous calculez la taille de vos données statiques, vous devez partir du principe que chaque colonne de clé de partition utilise les 3 octets complets de métadonnées.
- Utilisez la taille brute des valeurs de données des colonnes statiques en fonction du type de données. Pour plus d'informations sur les types de données, consultez the section called “Types de données”,
- Ajoutez 104 octets à la taille des données statiques pour les métadonnées.
- Les colonnes de clustering et les colonnes clés ordinaires non primaires ne sont pas prises en compte dans la taille des données statiques. Pour savoir comment estimer la taille des données non statiques au sein des lignes, consultezthe section called “Calcul de la taille des lignes”.

La taille totale codée d'une colonne statique est basée sur la formule suivante :

```
partition key columns + static columns + metadata = total encoded size of static data
```

Prenons l'exemple suivant d'une table où toutes les colonnes sont de type entier. La table comporte deux colonnes de clé de partition, deux colonnes de clustering, une colonne normale et une colonne statique.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2
 int, reg_col1 int, static_col1 int static, primary key((pk_col1, pk_col2),ck_col1,
 ck_col2));
```

Dans cet exemple, nous calculons la taille des données statiques de l'instruction suivante :

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, static_col1) values(1,2,6);
```

Pour estimer le nombre total d'octets requis par cette opération d'écriture, vous pouvez suivre les étapes suivantes.

1. Calculez la taille d'une colonne de clé de partition en ajoutant les octets correspondant au type de données stocké dans la colonne et les octets de métadonnées. Répétez cette opération pour toutes les colonnes clés de partition.

   a. Calculez la taille de la première colonne de la clé de partition (pk_col1) :

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   b. Calculez la taille de la deuxième colonne de la clé de partition (pk_col2) :

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   c. Ajoutez les deux colonnes pour obtenir la taille totale estimée des colonnes clés de partition :

   ```
   7 bytes + 7 bytes = 14 bytes for the partition key columns
   ```

2. Ajoutez la taille des colonnes statiques. Dans cet exemple, nous n'avons qu'une seule colonne statique qui stocke un entier (ce qui nécessite 4 octets).
3. Enfin, pour obtenir la taille totale codée des données des colonnes statiques, additionnez les octets pour les colonnes clés primaires et les colonnes statiques, puis ajoutez les 104 octets supplémentaires pour les métadonnées :

   ```
   14 bytes for the partition key columns + 4 bytes for the static column + 104 bytes
    for metadata = 122 bytes.
   ```

Vous pouvez également mettre à jour les données statiques et non statiques avec la même instruction. Pour estimer la taille totale de l'opération d'écriture, vous devez d'abord calculer la taille de la mise à jour des données non statiques. Calculez ensuite la taille de la mise à jour de la ligne comme indiqué dans l'exemple surthe section called “Calcul de la taille des lignes”, et ajoutez les résultats.

Dans ce cas, vous pouvez écrire un total de 2 Mo : 1 Mo est le quota de taille de ligne maximum, et 1 Mo est le quota de taille de données statique maximale par partition logique.

Pour calculer la taille totale d'une mise à jour de données statiques et non statiques dans la même instruction, vous pouvez utiliser la formule suivante :

```
(partition key columns + static columns + metadata = total encoded size of static data)
 + (partition key columns + clustering columns + regular columns + row metadata = total
 encoded size of row)
= total encoded size of data written
```

Prenons l'exemple suivant d'une table où toutes les colonnes sont de type entier. La table comporte deux colonnes de clé de partition, deux colonnes de clustering, une colonne normale et une colonne statique.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2
 int, reg_col1 int, static_col1 int static, primary key((pk_col1, pk_col2),ck_col1,
 ck_col2));
```

Dans cet exemple, nous calculons la taille des données lorsque nous écrivons une ligne dans le tableau, comme indiqué dans l'instruction suivante :

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1,
 static_col1) values(2,3,4,5,6,7);
```

Pour estimer le nombre total d'octets requis par cette opération d'écriture, vous pouvez suivre les étapes suivantes.

1. Calculez la taille totale codée des données statiques comme indiqué précédemment. Dans cet exemple, il s'agit de 122 octets.
2. Ajoutez la taille totale codée de la ligne en fonction de la mise à jour des données non statiques, en suivant les étapes décrites dansthe section called “Calcul de la taille des lignes”. Dans cet exemple, la taille totale de la mise à jour des lignes est de 134 octets.

```
122 bytes for static data + 134 bytes for nonstatic data = 256 bytes.
```

Mesurer les opérations de lecture/écriture de données statiques dans Amazon Keyspaces

Les données statiques sont associées à des partitions logiques dans Cassandra, et non à des lignes individuelles. Les partitions logiques d'Amazon Keyspaces peuvent avoir une taille pratiquement indépendante en s'étendant sur plusieurs partitions de stockage physiques. Par conséquent, les compteurs Amazon Keyspaces écrivent les opérations sur les données statiques et non statiques séparément. En outre, les écritures qui incluent à la fois des données statiques et non statiques nécessitent des opérations sous-jacentes supplémentaires pour assurer la cohérence des données.

Si vous effectuez une opération d'écriture mixte de données statiques et non statiques, cela entraîne deux opérations d'écriture distinctes, l'une pour les données non statiques et l'autre pour les données statiques. Cela s'applique aux modes de capacité de lecture/écriture à la demande et provisionnée.

L'exemple suivant explique comment estimer les unités de capacité de lecture (RCU) et d'écriture (WCU) requises lorsque vous calculez les exigences de capacité de débit allouée pour les tables d'Amazon Keyspaces comportant des colonnes statiques. Vous pouvez estimer la capacité dont votre table a besoin pour traiter les écritures incluant des données statiques et non statiques à l'aide de la formule suivante :

```
2 x WCUs required for nonstatic data + 2 x WCUs required for static data
```

Par exemple, si votre application écrit 27 Ko de données par seconde et que chaque écriture inclut 25,5 Ko de données non statiques et 1,5 Ko de données statiques, votre table nécessite 56 WCU (2 x 26 WCU + 2 x 2 WCU).

Amazon Keyspaces mesure les lectures de données statiques et non statiques de la même manière que les lectures de plusieurs lignes. Par conséquent, le prix de lecture de données statiques et non statiques au cours d'une même opération est basé sur la taille agrégée des données traitées pour effectuer la lecture.

Pour savoir comment surveiller les ressources sans serveur avec Amazon CloudWatch, consultezthe section called “Surveillance avec CloudWatch”.

Utilisation des lignes dans Amazon Keyspaces

Cette section fournit des informations sur l'utilisation des lignes dans Amazon Keyspaces (pour Apache Cassandra). Les tables sont les principales structures de données d'Amazon Keyspaces et les données des tables sont organisées en colonnes et en lignes.

Rubriques

Calcul de la taille des lignes dans Amazon Keyspaces

Amazon Keyspaces fournit un stockage entièrement géré qui offre des performances de lecture et d'écriture à un chiffre en millisecondes et stocke les données de manière durable dans plusieurs zones de disponibilité. AWS Amazon Keyspaces associe des métadonnées à toutes les lignes et à toutes les colonnes clés primaires afin de garantir un accès efficace aux données et une haute disponibilité.

Cette section explique comment estimer la taille codée des lignes dans Amazon Keyspaces. La taille de ligne codée est utilisée lors du calcul de votre facture et de votre quota d'utilisation. Vous devez également utiliser la taille de ligne codée lorsque vous calculez les exigences de capacité de débit allouées pour les tables. Pour calculer la taille codée des lignes dans Amazon Keyspaces, vous pouvez suivre les directives suivantes.

- Pour les colonnes ordinaires, qui ne sont pas des clés primaires, les colonnes de clustering ou les `STATIC` colonnes, utilisez la taille brute des données de cellule en fonction du type de données et ajoutez les métadonnées requises. Pour plus d'informations sur les types de données pris en charge dans Amazon Keyspaces, consultez. the section called “Types de données” Certaines des principales différences dans la manière dont Amazon Keyspaces stocke les valeurs des types de données et les métadonnées sont répertoriées ci-dessous.
- L'espace requis pour chaque nom de colonne est stocké à l'aide d'un identifiant de colonne et ajouté à chaque valeur de données stockée dans la colonne. La valeur de stockage de l'identifiant de colonne dépend du nombre total de colonnes de votre table :
  - 1 à 62 colonnes : 1 octet
  - 63 à 124 colonnes : 2 octets
  - 125 à 186 colonnes : 3 octets

Pour chaque 62 colonnes supplémentaires, ajoutez 1 octet. Notez que dans Amazon Keyspaces, jusqu'à 225 colonnes normales peuvent être modifiées à l'aide d'une seule instruction `INSERT` ou `UPDATE` d'une seule instruction. Pour en savoir plus, consultez the section called “Quotas de service Amazon Keyspaces”.

- Les clés de partition peuvent contenir jusqu'à 2 048 octets de données. Chaque colonne clé de la clé de partition nécessite jusqu'à 3 octets de métadonnées. Lorsque vous calculez la taille de votre ligne, vous devez partir du principe que chaque colonne de clé de partition utilise les 3 octets complets de métadonnées.
- Les colonnes de clustering peuvent stocker jusqu'à 850 octets de données. Outre la taille de la valeur des données, chaque colonne de clustering nécessite jusqu'à 20 % de la taille de la valeur des données pour les métadonnées. Lorsque vous calculez la taille de votre ligne, vous devez ajouter 1 octet de métadonnées pour chaque valeur de 5 octets de données de colonne de clustering.
- Amazon Keyspaces enregistre deux fois la valeur des données de chaque clé de partition et de chaque colonne de clé de clustering. La surcharge supplémentaire est utilisée pour des requêtes efficaces et une indexation intégrée.
- Les types de données Cassandra `ASCII` et `VARCHAR` string sont tous stockés dans Amazon Keyspaces en Unicode avec un codage binaire UTF-8. `TEXT` La taille d'une chaîne dans Amazon Keyspaces est égale au nombre d'octets codés en UTF-8.
- Cassandra`INT`, `BIGINT``SMALLINT`, et les types de `TINYINT` données sont stockés dans Amazon Keyspaces sous forme de valeurs de données de longueur variable, avec un maximum de 38 chiffres significatifs. Les zéros de début et de fin sont tronqués. La taille de chacun de ces types de données est d'environ 1 octet pour deux chiffres significatifs + 1 octet.
- `BLOB`Dans Amazon Keyspaces, A est stocké avec la longueur d'octet brute de la valeur.
- La taille d'une `Null` valeur ou d'une `Boolean` valeur est de 1 octet.
- Colonne qui stocke des types de données de collection tels que `LIST` ou `MAP` nécessitant 3 octets de métadonnées, quel que soit son contenu. La taille d'un `LIST` or `MAP` est (identifiant de colonne) + somme (taille des éléments imbriqués) + (3 octets). La taille d'un `LIST` or vide `MAP` est (identifiant de colonne) + (3 octets). Chaque individu `LIST` ou `MAP` élément nécessite également 1 octet de métadonnées.
- `STATIC`Les données de colonne ne sont pas prises en compte dans la taille de ligne maximale de 1 Mo. Pour calculer la taille des données des colonnes statiques, consultezthe section called “Calcul de la taille de colonne statique par partition logique”.

- Les horodatages côté client sont enregistrés pour chaque colonne de chaque ligne lorsque la fonctionnalité est activée. Ces horodatages occupent environ 20 à 40 octets (selon vos données) et contribuent au coût de stockage et de débit de la ligne. Pour en savoir plus, consultez the section called “Horodatages côté client dans Amazon Keyspaces”.
- Ajoutez 100 octets à la taille de chaque ligne pour les métadonnées de ligne.

La taille totale d'une ligne de données codée est basée sur la formule suivante :

```
partition key columns + clustering columns + regular columns + row metadata = total
 encoded size of row
```

> **Important**
>
> Toutes les métadonnées de colonne, par exemple les identifiants de colonne, les métadonnées de clé de partition, les métadonnées de colonne de clustering, ainsi que les horodatages côté client et les métadonnées de ligne sont prises en compte dans la taille de ligne maximale de 1 Mo.

Prenons l'exemple suivant d'une table où toutes les colonnes sont de type entier. La table comporte deux colonnes de clé de partition, deux colonnes de clustering et une colonne normale. Comme ce tableau comporte cinq colonnes, l'espace requis pour l'identifiant du nom de colonne est de 1 octet.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2 int,
 reg_col1 int, primary key((pk_col1, pk_col2),ck_col1, ck_col2));
```

Dans cet exemple, nous calculons la taille des données lorsque nous écrivons une ligne dans le tableau, comme indiqué dans l'instruction suivante :

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1)
 values(1,2,3,4,5);
```

Pour estimer le nombre total d'octets requis par cette opération d'écriture, vous pouvez suivre les étapes suivantes.

1. Calculez la taille d'une colonne de clé de partition en ajoutant les octets correspondant au type de données stocké dans la colonne et les octets de métadonnées. Répétez cette opération pour toutes les colonnes clés de partition.

a. Calculez la taille de la première colonne de la clé de partition (pk_col1) :

```
(2 bytes for the integer data type) x 2 + 1 byte for the column id + 3 bytes
 for partition key metadata = 8 bytes
```

b. Calculez la taille de la deuxième colonne de la clé de partition (pk_col2) :

```
(2 bytes for the integer data type) x 2 + 1 byte for the column id + 3 bytes
 for partition key metadata = 8 bytes
```

c. Ajoutez les deux colonnes pour obtenir la taille totale estimée des colonnes clés de partition :

```
8 bytes + 8 bytes = 16 bytes for the partition key columns
```

2. Calculez la taille de la colonne de clustering en ajoutant les octets correspondant au type de données stocké dans la colonne et les octets de métadonnées. Répétez cette opération pour toutes les colonnes de clustering.

a. Calculez la taille de la première colonne de la colonne de clustering (ck_col1) :

```
(2 bytes for the integer data type) x 2 + 20% of the data value (2 bytes) for
 clustering column metadata + 1 byte for the column id  = 6 bytes
```

b. Calculez la taille de la deuxième colonne de la colonne de clustering (ck_col2) :

```
(2 bytes for the integer data type) x 2 + 20% of the data value (2 bytes) for
 clustering column metadata + 1 byte for the column id = 6 bytes
```

c. Ajoutez les deux colonnes pour obtenir la taille totale estimée des colonnes de clustering :

```
6 bytes + 6 bytes = 12 bytes for the clustering columns
```

3. Ajoutez la taille des colonnes normales. Dans cet exemple, nous n'avons qu'une seule colonne qui stocke un entier à un chiffre, ce qui nécessite 2 octets dont 1 octet pour l'identifiant de colonne.

4. Enfin, pour obtenir la taille totale des lignes codées, additionnez les octets de toutes les colonnes et ajoutez les 100 octets supplémentaires pour les métadonnées des lignes :

```
16 bytes for the partition key columns + 12 bytes for clustering columns + 3 bytes
 for the regular column + 100 bytes for row metadata = 131 bytes.
```

Pour savoir comment surveiller les ressources sans serveur avec Amazon CloudWatch, consultezthe section called “Surveillance avec CloudWatch”.

Utilisation des requêtes dans Amazon Keyspaces

Cette section explique comment utiliser les requêtes dans Amazon Keyspaces (pour Apache Cassandra). Les instructions CQL disponibles pour interroger, transformer et gérer les données sont`SELECT`, `INSERT``UPDATE`, et`DELETE`. Les rubriques suivantes décrivent certaines des options les plus complexes disponibles lors de l'utilisation de requêtes. Pour obtenir la syntaxe complète du langage avec des exemples, voirthe section called “Instructions DML”.

Rubriques

- Utilisation de l'INopérateur avec la SELECT déclaration dans Amazon Keyspaces
- Résultats des commandes dans Amazon Keyspaces
- Pagination des résultats dans Amazon Keyspaces

Utilisation de l'**`IN`**opérateur avec la **`SELECT`** déclaration dans Amazon Keyspaces

SÉLECTIONNEZ DANS

Vous pouvez interroger les données des tables à l'aide de l'`SELECT`instruction, qui lit une ou plusieurs colonnes pour une ou plusieurs lignes d'une table et renvoie un jeu de résultats contenant les lignes correspondant à la demande. Une `SELECT` instruction contient un `select_clause` qui détermine les colonnes à lire et à renvoyer dans le jeu de résultats. La clause peut contenir des instructions pour transformer les données avant de les renvoyer. La `WHERE` clause facultative spécifie les lignes qui doivent être interrogées et est composée de relations sur les colonnes qui font partie de la clé primaire. Amazon Keyspaces prend en charge le `IN` mot clé dans la `WHERE` clause. Cette section utilise des exemples pour montrer comment Amazon Keyspaces traite les `SELECT` instructions contenant le `IN` mot clé.

Cet exemple montre comment Amazon Keyspaces décompose l'*SELECT*instruction contenant le *IN* mot clé en sous-requêtes. Dans cet exemple, nous utilisons une table portant le nommy_keyspace.customers. La table comporte une colonne de clé primairedepartment_id, deux colonnes sales_region_id de clustering et une colonne contenant le nom du client dans la customer_name colonne. sales_representative_id

```
SELECT * FROM my_keyspace.customers;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          0            |        0        |            0            |    a
          0            |        0        |            1            |    b
          0            |        1        |            0            |    c
          0            |        1        |            1            |    d
          1            |        0        |            0            |    e
          1            |        0        |            1            |    f
          1            |        1        |            0            |    g
          1            |        1        |            1            |    h
```

À l'aide de ce tableau, vous pouvez exécuter l'SELECTinstruction suivante pour trouver les clients des départements et des régions de vente qui vous intéressent avec le IN mot clé dans la WHERE clause. La déclaration suivante en est un exemple.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1) AND sales_region_id
 IN (0, 1);
```

Amazon Keyspaces divise cette instruction en quatre sous-requêtes, comme indiqué dans le résultat suivant.

```
SELECT * FROM my_keyspace.customers WHERE department_id = 0 AND sales_region_id = 0;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        0        |           0             |    a
  0            |        0        |           1             |    b

SELECT * FROM my_keyspace.customers WHERE department_id = 0 AND sales_region_id = 1;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        1        |          0              |    c
```

```
  0            |        1        |          1              |    d

SELECT * FROM my_keyspace.customers WHERE department_id = 1 AND sales_region_id = 0;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  1            |        0        |          0              |    e
  1            |        0        |          1              |    f

SELECT * FROM my_keyspace.customers WHERE department_id = 1 AND sales_region_id = 1;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  1            |        1        |           0             |    g
  1            |        1        |           1             |    h
```

Lorsque le `IN` mot clé est utilisé, Amazon Keyspaces pagine automatiquement les résultats dans les cas suivants :

- Après chaque dixième sous-requête traitée.
- Après avoir traité 1 Mo d'E/S logiques.
- Si vous avez configuré un`PAGE SIZE`, Amazon Keyspaces pagine après avoir lu le nombre de requêtes à traiter en fonction de l'ensemble. `PAGE SIZE`
- Lorsque vous utilisez le `LIMIT` mot clé pour réduire le nombre de lignes renvoyées, Amazon Keyspaces pagine après avoir lu le nombre de requêtes à traiter en fonction de l'ensemble. `LIMIT`

Le tableau suivant est utilisé pour illustrer cela par un exemple.

Pour plus d'informations sur la pagination, consultezthe section called “Pagination des résultats”.

```
SELECT * FROM my_keyspace.customers;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          2            |        0        |          0              |    g
          2            |        1        |          1              |    h
          2            |        2        |          2              |    i
          0            |        0        |          0              |    a
          0            |        1        |          1              |    b
          0            |        2        |          2              |    c
          1            |        0        |          0              |    d
```

```
          1           |        1        |          1              |    e
          1           |        2        |          2              |    f
          3           |        0        |          0              |    j
          3           |        1        |          1              |    k
          3           |        2        |          2              |    l
```

Vous pouvez exécuter l'instruction suivante sur ce tableau pour voir comment fonctionne la pagination.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1, 2, 3) AND
 sales_region_id IN (0, 1, 2) AND sales_representative_id IN (0, 1);
```

Amazon Keyspaces traite cette instruction sous la forme de 24 sous-requêtes, car la cardinalité du produit cartésien de tous les `IN` termes contenus dans cette requête est de 24.

```
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        0        |          0              |    a
  0            |        1        |          1              |    b
  1            |        0        |          0              |    d
  1            |        1        |          1              |    e

---MORE---
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  2            |        0        |          0              |    g
  2            |        1        |          1              |    h
  3            |        0        |          0              |    j

---MORE---
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  3            |        1        |          1              |    k
```

Cet exemple montre comment utiliser la `ORDER BY` clause dans une `SELECT` instruction contenant le `IN` mot clé.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (3, 2, 1) ORDER BY
 sales_region_id DESC;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
```

```
        3             |        2        |          2              |    l
        3             |        1        |          1              |    k
        3             |        0        |          0              |    j
        2             |        2        |          2              |    i
        2             |        1        |          1              |    h
        2             |        0        |          0              |    g
        1             |        2        |          2              |    f
        1             |        1        |          1              |    e
        1             |        0        |          0              |    d
```

Les sous-requêtes sont traitées dans l'ordre dans lequel les colonnes de clé de partition et de clé de clustering sont présentées dans la requête. Dans l'exemple ci-dessous, les sous-requêtes pour la valeur de clé de partition « 2 » sont traitées en premier, suivies des sous-requêtes pour les valeurs de clé de partition « 3 » et « 1 ». Les résultats d'une sous-requête donnée sont classés en fonction de la clause d'ordre de la requête, le cas échéant, ou de l'ordre de regroupement de la table défini lors de la création de la table.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (2, 3, 1) ORDER BY
 sales_region_id DESC;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          2            |        2        |          2              |    i
          2            |        1        |          1              |    h
          2            |        0        |          0              |    g
          3            |        2        |          2              |    l
          3            |        1        |          1              |    k
          3            |        0        |          0              |    j
          1            |        2        |          2              |    f
          1            |        1        |          1              |    e
          1            |        0        |          0              |    d
```

Résultats des commandes dans Amazon Keyspaces

La `ORDER BY` clause spécifie l'ordre de tri des résultats renvoyés dans une `SELECT` instruction. L'instruction prend une liste de noms de colonnes comme arguments et pour chaque colonne, vous pouvez spécifier l'ordre de tri des données. Vous pouvez uniquement spécifier des colonnes de clustering dans les clauses de commande, les colonnes non clustering ne sont pas autorisées.

Les deux options d'ordre de tri disponibles pour les résultats renvoyés sont `ASC` l'ordre de tri croissant et l'ordre `DESC` de tri décroissant.

```
SELECT * FROM my_keyspace.my_table ORDER BY (col1 ASC, col2 DESC, col3 ASC);

 col1 | col2 | col3
------+------+------
  0   |  6   |  a
  1   |  5   |  b
  2   |  4   |  c
  3   |  3   |  d
  4   |  2   |  e
  5   |  1   |  f
  6   |  0   |  g
```

```
SELECT * FROM my_keyspace.my_table ORDER BY (col1 DESC, col2 ASC, col3 DESC);

 col1 | col2 | col3
------+------+------
  6   |  0   |  g
  5   |  1   |  f
  4   |  2   |  e
  3   |  3   |  d
  2   |  4   |  c
  1   |  5   |  b
  0   |  6   |  a
```

Si vous ne spécifiez pas l'ordre de tri dans l'instruction de requête, l'ordre par défaut de la colonne de clustering est utilisé.

Les ordres de tri possibles que vous pouvez utiliser dans une clause de classement dépendent de l'ordre de tri attribué à chaque colonne de clustering lors de la création de la table. Les résultats des requêtes ne peuvent être triés que dans l'ordre défini pour toutes les colonnes de clustering lors de la création de la table ou dans l'ordre inverse de l'ordre de tri défini. Les autres combinaisons possibles ne sont pas autorisées.

Par exemple, si la table `CLUSTERING ORDER` est (col1 ASC, col2 DESC, col3 ASC), les paramètres valides pour `ORDER BY` sont soit (col1 ASC, col2 DESC, col3 ASC) soit (col1 DESC, col2 ASC, col3 DESC). Pour plus d'informations sur`CLUSTERING ORDER`, voir `table_options` ci-dessousthe section called “CREATE TABLE”.

Pagination des résultats dans Amazon Keyspaces

Amazon Keyspaces pagine automatiquement les résultats des `SELECT` instructions lorsque les données lues pour traiter la `SELECT` déclaration dépassent 1 Mo. Avec la pagination, les résultats de l'`SELECT`instruction sont divisés en « pages » de données de 1 Mo (ou moins). Une application peut traiter la première page des résultats, puis la deuxième page, et ainsi de suite. Les clients doivent toujours vérifier la présence de jetons de pagination lorsqu'ils traitent `SELECT` des requêtes renvoyant plusieurs lignes.

Si un client fournit un fichier `PAGE SIZE` nécessitant la lecture de plus de 1 Mo de données, Amazon Keyspaces divise automatiquement les résultats en plusieurs pages en fonction des incréments de lecture de données de 1 Mo.

Par exemple, si la taille moyenne d'une ligne est de 100 Ko et que vous spécifiez une valeur `PAGE SIZE` de 20, Amazon Keyspaces pagine automatiquement les données après avoir lu 10 lignes (1 000 Ko de données lues).

Amazon Keyspaces pagine les résultats en fonction du nombre de lignes lues pour traiter une demande et non du nombre de lignes renvoyées dans le jeu de résultats. Il est donc possible que certaines pages ne contiennent aucune ligne si vous exécutez des requêtes filtrées.

Par exemple, si vous définissez le `PAGE SIZE` paramètre sur 10 et que Keyspaces évalue 30 lignes pour traiter votre requête`SELECT`, Amazon Keyspaces renverra trois pages. Si seul un sous-ensemble de lignes correspond à votre requête, certaines pages peuvent comporter moins de 10 lignes. Pour un exemple `PAGE SIZE` de la manière dont les `LIMIT` requêtes of peuvent affecter la capacité de lecture, consultezthe section called “Limiter les requêtes”.

Utilisation des partitionneurs dans Amazon Keyspaces

Dans Apache Cassandra, les partitionneurs contrôlent les nœuds sur lesquels les données sont stockées dans le cluster. Les partitionneurs créent un jeton numérique à l'aide d'une valeur hachée de la clé de partition. Cassandra utilise ce jeton pour distribuer les données entre les nœuds. Les clients peuvent également utiliser ces jetons dans les `SELECT` opérations et les `WHERE` clauses afin d'optimiser les opérations de lecture et d'écriture. Par exemple, les clients peuvent exécuter efficacement des requêtes parallèles sur de grandes tables en spécifiant des plages de jetons distinctes à interroger dans chaque tâche parallèle.

Amazon Keyspaces propose trois partitionneurs différents.

Murmur3 Partitioner (par défaut)

Compatible avec Apache Cassandra. `Murmur3Partitioner` `Murmur3Partitioner`Il s'agit du partitionneur Cassandra par défaut dans Amazon Keyspaces et dans Cassandra 1.2 et versions ultérieures.

RandomPartitioner

Compatible avec Apache Cassandra. `RandomPartitioner` `RandomPartitioner`Il s'agit du partitionneur Cassandra par défaut pour les versions antérieures à Cassandra 1.2.

Partitionneur par défaut de Keyspaces

`DefaultPartitioner`Renvoie les mêmes résultats `token` de fonction que le`RandomPartitioner`.

Le paramètre du partitionneur est appliqué par région au niveau du compte. Par exemple, si vous modifiez le partitionneur dans l'est des États-Unis (Virginie du Nord), la modification est appliquée à toutes les tables du même compte dans cette région. Vous pouvez modifier votre partitionneur en toute sécurité à tout moment. Notez que la modification de configuration prend environ 10 minutes. Vous n'avez pas besoin de recharger vos données Amazon Keyspaces lorsque vous modifiez le paramètre du partitionneur. Les clients utiliseront automatiquement le nouveau paramètre du partitionneur lors de leur prochaine connexion.

Vous pouvez modifier le partitionneur à l'aide du langage CQL (AWS Management Consoleou Cassandra Query Language).

AWS Management Console

Pour modifier le partitionneur à l'aide de la console Amazon Keyspaces

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Configuration.
3. Sur la page de configuration, accédez à Modifier le partitionneur.
4. Sélectionnez le partitionneur compatible avec votre version de Cassandra. Le changement de partitionneur prend environ 10 minutes pour s'appliquer.

> **Note**
>
> Une fois la modification de configuration terminée, vous devez vous déconnecter et vous reconnecter à Amazon Keyspaces pour les demandes d'utilisation du nouveau partitionneur.

Cassandra Query Language (CQL)

1. Pour voir quel partitionneur est configuré pour le compte, vous pouvez utiliser la requête suivante.

```
SELECT partitioner from system.local;
```

   Si le partitionneur n'a pas été modifié, le résultat de la requête est le suivant.

```
partitioner
-----------------------------------------
com.amazonaws.cassandra.DefaultPartitioner
```

2. Pour mettre à jour le partitionneur vers le `Murmur3` partitionneur, vous pouvez utiliser l'instruction suivante.

```
UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

3. Notez que cette modification de configuration prend environ 10 minutes. Pour confirmer que le partitionneur a été configuré, vous pouvez exécuter à nouveau la `SELECT` requête. Notez qu'en raison d'une éventuelle cohérence de lecture, la réponse peut ne pas encore refléter les résultats du changement de partitionneur récemment effectué. Si vous répétez l'`SELECT`opération peu de temps après, la réponse doit renvoyer les données les plus récentes.

```
SELECT partitioner from system.local;
```

> **Note**
>
> Vous devez vous déconnecter puis vous reconnecter à Amazon Keyspaces pour que les demandes utilisent le nouveau partitionneur.

Utilisation de balises et d'étiquettes pour les ressources Amazon Keyspaces

Vous pouvez étiqueter les ressources Amazon Keyspaces (pour Apache Cassandra) à l'aide de balises. Les balises vous permettent de classer vos ressources de différentes manières, par exemple en fonction de leur objectif, de leur propriétaire, de leur environnement ou d'autres critères. Les balises vous permettent d'effectuer les actions suivantes :

- Identifier rapidement une ressource en fonction des balises que vous lui avez attribuées.
- Voir AWS les factures ventilées par tags.
- Contrôlez l'accès aux ressources Amazon Keyspaces à l'aide de balises. Pour des exemples de politique IAM utilisant des balises, consultezthe section called “Autorisation basée sur les tags Amazon Keyspaces”.

Le balisage est pris en charge par AWS des services tels qu'Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Keyspaces, etc. Un balisage efficace peut fournir un aperçu sur les coûts en vous permettant de créer des rapports sur les services associés à une balise spécifique.

Pour commencer le balisage, procédez comme suit :

1. Comprendre Restrictions de balisage pour Amazon Keyspaces.
2. Créer des balises en utilisant Opérations de balisage pour Amazon Keyspaces.
3. Rapports de répartition des coûts pour Amazon KeyspacesÀ utiliser pour suivre vos AWS coûts par tag actif.

En dernier lieu, il est conseillé de suivre les politique de balisage optimales. Pour plus d'informations, consultez Politiques d'étiquetage AWS.

Restrictions de balisage pour Amazon Keyspaces

Chaque étiquette est constituée d'une clé et d'une valeur, que vous définissez. Les restrictions suivantes s'appliquent :

- Chaque espace ou table Amazon Keyspaces ne peut avoir qu'une seule étiquette avec la même clé. Si vous tentez d'ajouter une balise existante (même clé), la valeur de la balise existante est mise à jour avec la nouvelle valeur.
- Les balises appliquées à un espace de touches ne s'appliquent pas automatiquement aux tables de cet espace de touches. Pour appliquer la même étiquette à un espace clé et à toutes ses tables, chaque ressource doit être balisée individuellement.
- Lorsque vous créez un espace clé ou un tableau multirégional, toutes les balises que vous définissez au cours du processus de création sont automatiquement appliquées à tous les espaces clés et à tous les tableaux de toutes les régions. Lorsque vous modifiez des balises existantes à l'aide de `ALTER KEYSPACE` ou`ALTER TABLE`, la mise à jour s'applique uniquement à l'espace clavier ou à la table de la région dans laquelle vous effectuez la modification.
- Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé). Dans Amazon Keyspaces, la valeur ne peut pas être vide ou null.
- Les clés et valeurs de balise sont sensibles à la casse.
- La longueur de clé maximale est de 128 caractères Unicode.
- La longueur de valeur maximale est de 256 caractères Unicode.
- Les caractères autorisés sont les lettres, les espaces blancs et les chiffres, ainsi que les caractères spéciaux suivants : `+ - = . _ : /`
- Le nombre maximum d'identifications par ressource est de 50.
- AWSLes noms et les valeurs de balises attribuées par reçoivent automatiquement le `aws:` préfixe, que vous ne pouvez pas attribuer. AWS-les noms de balises attribués ne sont pas pris en compte dans la limite de 50 balises. Les noms des balises attribuées par l'utilisateur ont le préfixe `user:` dans le rapport de répartition des coûts.
- Vous ne pouvez pas antidater l'application d'une balise.

Opérations de balisage pour Amazon Keyspaces

Vous pouvez ajouter, répertorier, modifier ou supprimer des balises pour les espaces clés et les tables à l'aide de la console Amazon Keyspaces (pour Apache Cassandra), de laAWS CLI ou du langage de requête Cassandra (CQL). Vous pouvez ensuite activer ces balises définies par

l'utilisateur afin de les faire apparaître sur la console AWS Billing and Cost Management pour le suivi de la répartition des coûts. Pour plus d'informations, veuillez consulter Rapports de répartition des coûts pour Amazon Keyspaces.

Pour la modification en bloc, vous pouvez également utiliser l'éditeur d'étiquettes sur la console. Pour plus d'informations, veuillez consulter Utilisation de Tag Editor dans le Guide de l'utilisateur AWS Resource Groups.

Rubriques

- Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide de la console
- Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide de laAWS CLI
- Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide du CQL

Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide de la console

Vous pouvez utiliser la console Amazon Keyspaces pour ajouter des étiquettes à de nouvelles tables et de nouvelles tables lors de leur création. Vous pouvez également ajouter, répertorier, modifier ou supprimer des balises pour les tables existantes.

Pour baliser des keyspaces lors de leur création (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces, puis Créer un keyspace.
3. Dans la page Créer un keyspace indiquez un nom pour le keyspace. Entrez une clé et une valeur pour la balise, puis choisissez Ajouter une nouvelle balise.
4. Choisissez Create keyspace.

Pour baliser les tables lors de leur création (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Tables, puis Create table (Créer une table).
3. Sur la page Créer une table, dans la section Détails de la table, sélectionnez un espace clé et donnez un nom à la table.

4. Dans la section Schéma, créez le schéma de votre table.
5. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
6. Passez à la section Tags du tableau — facultative, et choisissez Ajouter une nouvelle balise pour créer de nouvelles balises.
7. Choisissez Créer un tableau.

Pour baliser des ressources existantes (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces ou Tables.
3. Choisissez un keyspace ou une table dans la liste. Choisissez ensuite Gérer les balises pour ajouter, modifier ou supprimer vos balises.

Pour plus d'informations sur la structure des balises, consultez Restrictions de balisage pour Amazon Keyspaces.

Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide de laAWS CLI

Les exemples de cette section montrent comment utiliser l'AWSinterface de ligne de commande pour spécifier des balises lorsque vous créez des espaces clés et des tables, comment ajouter ou supprimer des balises dans des ressources existantes et comment répertorier des balises.

L'exemple suivant montre comment créer une nouvelle table avec des étiquettes. La commande crée une table MyTable dans un espace de touches MyKeyspace déjà existant. Notez que la commande a été divisée en différentes lignes pour faciliter la lisibilité.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --tags 'key=key1,value=val1' 'key=key2,value=val2'
```

L'exemple suivant montre comment ajouter de nouvelles étiquettes à une table existante.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111222333444:/
keyspace/myKeyspace/table/myTable' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

L'exemple suivant montre comment répertorier les étiquettes de la ressource spécifiée.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111222333444:/keyspace/myKeyspace/table/myTable'
```

La sortie de la dernière commande ressemble à ceci.

```
{
    "tags": [
        {
            "key": "key1",
            "value": "val1"
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

Ajout d'étiquettes à des tables et des espaces clés nouveaux ou existants à l'aide du CQL

Les exemples suivants montrent comment utiliser CQL pour spécifier des balises lorsque vous créez des keyspaces et des tables, comment baliser des ressources existantes et comment lire des balises.

L'exemple suivant crée un nouveau keyspace avec des balises.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'} ;
```

L'exemple suivant crée une table avec des balises.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

Pour baliser les ressources dans une instruction avec d'autres commandes

```
CREATE KEYSPACE mykeyspace WITH REPLICATION = {'class': 'Simple Strategy'} AND TAGS
 = {'key1':'val1', 'key2':'val2'};
```

L'exemple suivant montre comment ajouter ou supprimer des balises sur des keyspaces et des tables existants.

```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

```
ALTER TABLE mytable DROP TAGS {'key1':'val1', 'key2':'val2'};
```

Pour lire les balises attachées à une ressource, utilisez l'instruction CQL suivante.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

La clause WHERE est obligatoire et doit avoir l'un des formats suivants :

- keyspace_name = **'mykeyspace'** AND resource_type = 'keyspace'
- keyspace_name = **'mykeyspace'** AND resource_name = **'mytable'**
- resource_id = *arn*

Exemples :

La requête suivante indique si un keyspace comporte des balises.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_type = 'keyspace';
```

La sortie de la commande ressemble à ce qui suit.

```
resource_id                                                  | keyspace_name |
 resource_name | resource_type | tags
```

```
-------------------------------------------------------------------+--------------
+--------------+--------------+------
arn:aws:cassandra:us-east-1:123456789:/keyspace/mykeyspace/      | mykeyspace    |
 mykeyspace    | keyspace      | {'key1': 'val1', 'key2': 'val2'}
```

La requête suivante affiche les balises d'une table.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_name = 'mytable';
```

La sortie de cette requête se présente comme suit :

```
resource_id                                                                 |
 keyspace_name | resource_name | resource_type | tags
-------------------------------------------------------------------------------
+--------------+--------------+--------------+------
arn:aws:cassandra:us-east-1:123456789:/keyspace/mykeyspace/table/mytable    |
 mykeyspace    | mytable       | table         | {'key1': 'val1', 'key2': 'val2'}
```

Rapports de répartition des coûts pour Amazon Keyspaces

AWS utilise des étiquettes pour organiser les coûts des ressources dans votre rapport de répartition des coûts. AWS fournit deux types d'étiquettes de répartition des coûts :

- Etiquette générée par AWS. AWS définit, crée et applique cette étiquette pour vous.
- Etiquettes définies par l'utilisateur Vous définissez, créez et appliquez ces étiquettes.

Vous devez activer les deux types d'étiquettes séparément pour qu'elles apparaissent dans Cost Explorer ou sur un rapport de répartition des coûts.

Pour activer les étiquettes générées par AWS :

1. Connectez-vous à l'AWS Management Console et ouvrez la console de facturation et de gestion des coûts à l'adresse https://console.aws.amazon.com/billing/home#/.

2. Dans le volet de navigation, choisissez Balises de répartition des coûts.
3. Sous Identifications de répartition des coûts générées par AWS, choisissez Activer.

Pour activer les étiquettes définies par l'utilisateur :

1. Connectez-vous à l'AWS Management Console et ouvrez la console de facturation et de gestion des coûts à l'adresse https://console.aws.amazon.com/billing/home#/.
2. Dans le volet de navigation, choisissez Balises de répartition des coûts.
3. Sous User-Defines Cost Allocation Tags (Etiquettes de répartition des coûts définies par l'utilisateur), choisissez Activer.

Une fois les étiquettes créées et activées, AWS génère un rapport de répartition des coûts faisant apparaître votre utilisation et vos coûts regroupés par étiquettes actives. Le rapport de répartition des coûts inclut tous vos coûts AWS pour chaque période de facturation. Ce rapport inclut les ressources balisées et non balisées, afin que vous puissiez organiser clairement les frais pour les ressources.

> **Note**
>
> Actuellement, les données transférées à partir d'Amazon Keyspaces ne sont pas réparties par étiquettes sur les rapports de répartition des coûts.

Pour plus d'informations, consultez Utilisation des étiquettes d'allocation des coûts.

Bonnes pratiques en matière de conception et d'architecture avec Amazon Keyspaces

Utilisez cette section pour trouver rapidement des recommandations visant à optimiser les performances et à minimiser les coûts de débit lorsque vous travaillez avec Amazon Keyspaces.

Table des matières

Conception NoSQL pour Amazon Keyspaces

Les systèmes de base de données NoSQL tels qu'Amazon Keyspaces utilisent des modèles alternatifs pour la gestion des données, tels que les paires clé-valeur ou le stockage de documents. Lorsque vous passez d'un système de gestion de base de données relationnelle à un système de base de données NoSQL tel qu'Amazon Keyspaces, il est important de comprendre les principales différences et les approches de conception spécifiques.

Rubriques

Différences entre le système de base de données relationnelle et le système NoSQL

Les systèmes de gestion de bases de données relationnelles (SGBDR) et les bases de données NoSQL ont chacun leurs forces et leurs faiblesses :

- Dans les SGBDR, les données peuvent être interrogées avec souplesse, mais les requêtes sont relativement coûteuses et ne sont pas bien mises à l'échelle dans les situations de trafic intense (consultez the section called “Modélisation des données”).
- Dans une base de données NoSQL telle qu'Amazon Keyspaces, les données peuvent être consultées efficacement d'un nombre limité de manières, en dehors desquelles les requêtes peuvent être coûteuses et lentes.

Ces différences entraînent une conception des bases de données très différente d'un système à l'autre :

- Dans un SGBDR, la conception se concentre avant tout sur la flexibilité, sans se préoccuper des détails de l'implémentation ni des performances. En général, l'optimisation des requêtes n'affecte pas la conception du schéma, mais la normalisation est très importante.
- Dans Amazon Keyspaces, vous concevez votre schéma spécifiquement pour que les requêtes les plus courantes et les plus importantes soient aussi rapides et économiques que possible. Les structures de vos données sont adaptées aux exigences spécifiques de vos cas d'utilisation.

Deux concepts essentiels de la conception NoSQL

Pour concevoir un système NoSQL, il faut un autre état d'esprit que pour un SGBDR. Pour un SGBDR, vous pouvez créer un modèle de données normalisé sans réfléchir aux modèles d'accès. Vous pouvez l'étendre ultérieurement, pour répondre à de nouvelles questions et de nouveaux besoins d'interrogation. Vous pouvez organiser chaque type de données dans sa propre table.

En quoi la conception d'un système NoSQL est différente

- En revanche, vous ne devriez pas commencer à concevoir votre schéma pour Amazon Keyspaces tant que vous ne connaissez pas les questions auxquelles il doit répondre. Il est essentiel d'identifier au préalable les problèmes métier et les cas d'utilisation de l'application.
- Vous devez gérer le moins de tables possible dans une application Amazon Keyspaces. Le fait de disposer de moins de tables permet de rendre les choses plus évolutives, de réduire la gestion des autorisations et de réduire la charge de travail de votre application Amazon Keyspaces. Cela peut également contribuer à maintenir des coûts de sauvegarde globalement plus faibles.

Approche de la conception NoSQL

La première étape de la conception de votre application Amazon Keyspaces consiste à identifier les modèles de requêtes spécifiques auxquels le système doit répondre.

Il est notamment important de comprendre trois propriétés fondamentales des modèles d'accès de votre application avant de commencer :

- Taille des données : connaître la quantité de données qui sera stockée et demandée en une seule fois permet de déterminer le moyen le plus efficace de partitionner les données.
- Forme des données : au lieu de remodéliser les données lors du traitement d'une requête (comme c'est le cas dans un SGBDR), une base de données NoSQL organise les données de manière à ce que leur forme dans la base de données corresponde aux requêtes. C'est un élément clé pour augmenter la vitesse et la scalabilité.
- Vélocité des données : Amazon Keyspaces évolue en augmentant le nombre de partitions physiques disponibles pour traiter les requêtes et en distribuant efficacement les données entre ces partitions. Connaître à l'avance les pics des charges de requête peut aider à déterminer comment partitionner les données afin de tirer pleinement parti de la capacité I/O.

Après avoir identifié les besoins de requête spécifiques, vous pouvez organiser les données en fonction des principes généraux qui déterminent la performance :

- Rassemblez les données connexes. Il y a 20 ans, les recherches visant à optimiser les tables de routage ont démontré que l'emplacement des références était le facteur le plus important pour accélérer le temps de réponse : il convient de rassembler les données connexes au même endroit. C'est toujours le cas dans les systèmes NoSQL aujourd'hui, dans lesquels le fait de conserver les données connexes à proximité a un impact très important sur les coûts et les performances. Au lieu

de répartir des éléments de données connexes sur plusieurs tables, il est préférable de conserver les éléments connexes dans votre système NoSQL aussi près que possible les uns des autres.

En règle générale, vous devez gérer le moins de tables possible dans une application Amazon Keyspaces.

Les cas où des données de série chronologique volumineuses sont impliquées ou dans lesquels les ensembles de données ont des modèles d'accès très différents sont des exemples d'exceptions. Une table unique avec des index inversés peut généralement permettre à des requêtes simples de créer et récupérer les structures de données hiérarchiques complexes dont votre application a besoin.

- Utilisez l'ordre de tri. Les éléments connexes peuvent être regroupés et interrogés de manière efficace si la conception des clés permet de les trier ensemble. Il s'agit d'une stratégie de conception NoSQL importante.
- Répartissez les requêtes. Il est également important qu'un volume élevé de requêtes ne soit pas centré sur une même partie de la base de données, où la capacité d'E/S peut être dépassée. Au lieu de cela, vous devez concevoir les clés de données de manière à ce que le trafic soit, autant que possible, répartit de manière uniforme sur les partitions, évitant ainsi les « zones sensibles ».

Ces principes généraux se traduisent par des modèles de conception courants que vous pouvez utiliser pour modéliser efficacement les données dans Amazon Keyspaces.

Connexions du pilote client à Amazon Keyspaces (pour Apache Cassandra)

Pour communiquer avec Amazon Keyspaces, vous pouvez utiliser n'importe quel pilote client Apache Cassandra existant de votre choix. Amazon Keyspaces étant un service sans serveur, nous vous recommandons d'optimiser la configuration de connexion de votre pilote client en fonction des besoins de débit de votre application. Cette rubrique présente les meilleures pratiques, notamment le calcul du nombre de connexions requises par votre application, ainsi que la surveillance et le traitement des erreurs des connexions.

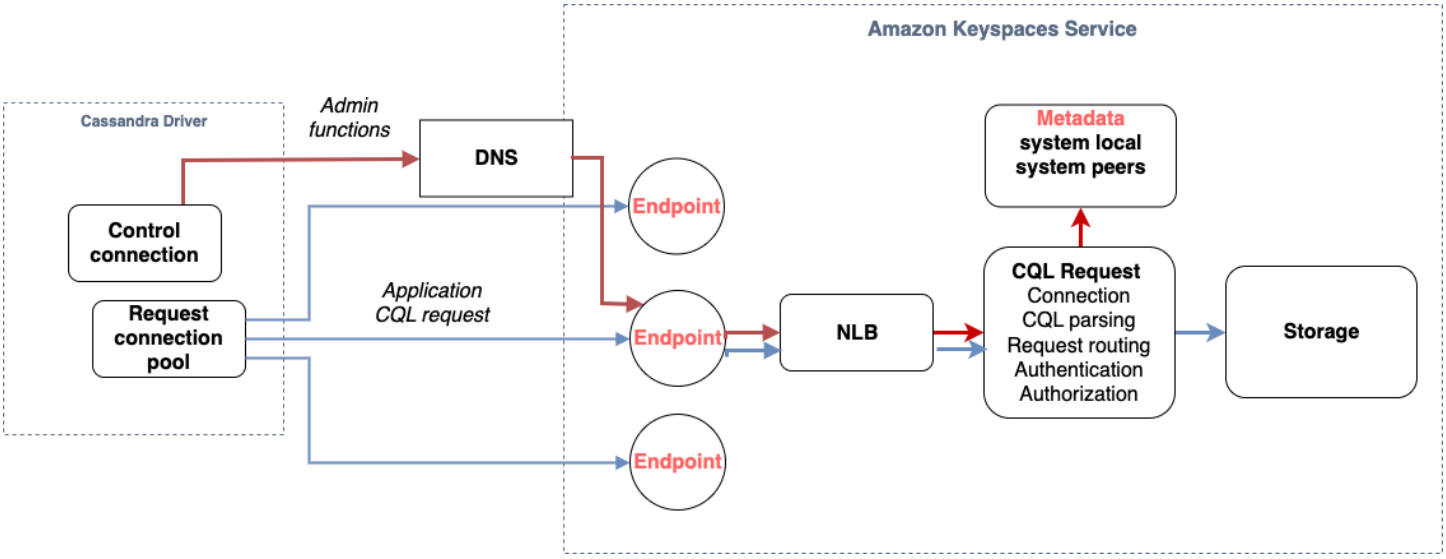
Rubriques

- Comment surveiller les connexions dans Amazon Keyspaces
- Comment gérer les erreurs de connexion dans Amazon Keyspaces

Comment fonctionnent les connexions dans Amazon Keyspaces

Cette section donne un aperçu du fonctionnement des connexions client-pilote dans Amazon Keyspaces. Étant donné que la mauvaise configuration du pilote client Cassandra peut entraîner `PerConnectionRequestExceeded` des événements dans Amazon Keyspaces, il est nécessaire de configurer le nombre approprié de connexions dans la configuration du pilote client pour éviter ces erreurs de connexion et d'autres similaires.

Lors de la connexion à Amazon Keyspaces, le pilote a besoin d'un point de terminaison initial pour établir une connexion initiale. Amazon Keyspaces utilise le DNS pour acheminer la connexion initiale vers l'un des nombreux points de terminaison disponibles. Les points de terminaison sont attachés à des équilibreurs de charge réseau qui établissent à leur tour une connexion avec l'un des gestionnaires de demandes du parc. Une fois la connexion initiale établie, le pilote client collecte des informations sur tous les points de terminaison disponibles dans le `system.peers` tableau. Grâce à ces informations, le pilote client peut créer des connexions supplémentaires aux points de terminaison répertoriés. Le nombre de connexions que le pilote client peut créer est limité par le nombre de connexions locales spécifié dans les paramètres du pilote client. Par défaut, la plupart des pilotes clients établissent une connexion par point de terminaison, établissent un pool de connexions avec Cassandra et équilibrent la charge des requêtes sur ce pool de connexions. Bien que plusieurs connexions puissent être établies vers le même point de terminaison, elles peuvent être connectées derrière l'équilibreur de charge réseau à de nombreux gestionnaires de demandes différents. Lorsque vous vous connectez via le point de terminaison public, l'établissement d'une connexion à chacun des neuf points de terminaison répertoriés dans le `system.peers` tableau entraîne neuf connexions à différents gestionnaires de demandes.



Comment configurer les connexions dans Amazon Keyspaces

Amazon Keyspaces prend en charge jusqu'à 3 000 requêtes CQL par connexion TCP et par seconde. Étant donné qu'il n'existe aucune limite quant au nombre de connexions qu'un pilote peut établir, nous recommandons de ne cibler que 500 demandes CQL par seconde et par connexion afin de prendre en compte les surcharges, les pics de trafic et un meilleur équilibrage de charge. Suivez ces étapes pour vous assurer que la connexion de votre pilote est correctement configurée en fonction des besoins de votre application.

Augmentez le nombre de connexions par adresse IP que votre pilote gère dans son pool de connexions.

- La plupart des pilotes Cassandra établissent un pool de connexions avec Cassandra et équilibrent la charge des requêtes sur ce pool de connexions. Le comportement par défaut de la plupart des pilotes consiste à établir une connexion unique avec chaque point de terminaison. Amazon Keyspaces expose neuf adresses IP homologues aux conducteurs. Ainsi, selon le comportement par défaut de la plupart des conducteurs, cela se traduit par 9 connexions. Amazon Keyspaces prend en charge jusqu'à 3 000 requêtes CQL par connexion TCP par seconde. Par conséquent, le débit de requêtes CQL maximal d'un pilote utilisant les paramètres par défaut est de 27 000 requêtes CQL par seconde. Si vous utilisez les paramètres par défaut du pilote, il se peut qu'une seule connexion doive traiter un débit de requêtes CQL supérieur au débit maximal de 3 000 requêtes CQL par seconde. Cela pourrait entraîner des `PerConnectionRequestExceeded` événements.
- Pour éviter les `PerConnectionRequestExceeded` événements, vous devez configurer le pilote pour créer des connexions supplémentaires par point de terminaison afin de distribuer le débit.

- Pour Amazon Keyspaces, il est recommandé de partir du principe que chaque connexion peut prendre en charge 500 requêtes CQL par seconde.
- Cela signifie que pour une application de production qui doit prendre en charge environ 27 000 requêtes CQL par seconde réparties sur les neuf points de terminaison disponibles, vous devez configurer six connexions par point de terminaison. Cela garantit que chaque connexion ne traite pas plus de 500 demandes par seconde.

Calculez le nombre de connexions par adresse IP que vous devez configurer pour votre pilote en fonction des besoins de votre application.

Pour déterminer le nombre de connexions que vous devez configurer par point de terminaison pour votre application, considérez l'exemple suivant. Vous avez une application qui doit prendre en charge 20 000 requêtes CQL par seconde comprenant 10 000 `INSERTSELECT`, 5 000 et 5 000 `DELETE` opérations. L'application Java s'exécute sur trois instances sur Amazon Elastic Container Service (Amazon ECS), où chaque instance établit une session unique avec Amazon Keyspaces. Le calcul que vous pouvez utiliser pour estimer le nombre de connexions que vous devez configurer pour votre pilote utilise l'entrée suivante.

1. Le nombre de demandes par seconde que votre application doit prendre en charge.
2. Le nombre d'instances disponibles, une étant soustraite pour tenir compte de la maintenance ou de la panne.
3. Le nombre de points de terminaison disponibles. Si vous vous connectez via des points de terminaison publics, vous disposez de neuf points de terminaison disponibles. Si vous utilisez des points de terminaison VPC, vous avez entre deux et cinq points de terminaison disponibles, selon la région.
4. Utiliser 500 requêtes CQL par seconde et par connexion est une bonne pratique pour Amazon Keyspaces.
5. Arrondissez le résultat.

Dans cet exemple, la formule ressemble à ceci.

```
20,000 CQL queries / (3 instances - 1 failure) / 9 public endpoints / 500 CQL queries
 per second = ROUND(2.22) = 3
```

Sur la base de ce calcul, vous devez spécifier trois connexions locales par point de terminaison dans la configuration du pilote. Pour les connexions à distance, configurez une seule connexion par point de terminaison.

Comment configurer les connexions via les points de terminaison VPC dans Amazon Keyspaces

Lorsque vous vous connectez via des points de terminaison VPC privés, vous disposez probablement de 3 points de terminaison disponibles. Le nombre de points de terminaison VPC peut être différent par région, en fonction du nombre de zones de disponibilité et du nombre de sous-réseaux dans le VPC attribué. La région USA Est (Virginie du Nord) compte cinq zones de disponibilité et vous pouvez avoir jusqu'à cinq points de terminaison Amazon Keyspaces. La région USA Ouest (Californie du Nord) comporte deux zones de disponibilité et vous pouvez avoir jusqu'à deux points de terminaison Amazon Keyspaces. Le nombre de points de terminaison n'a aucun impact sur l'échelle, mais il augmente le nombre de connexions que vous devez établir dans la configuration du pilote. Prenez l'exemple de code suivant. Votre application doit prendre en charge 20 000 requêtes CQL et s'exécute sur trois instances sur Amazon ECS, chaque instance établissant une session unique avec Amazon Keyspaces. La seule différence réside dans le nombre de points de terminaison disponibles dans les différents Régions AWS.

Connexions requises dans la région USA Est (Virginie du Nord) :

```
20,000 CQL queries / (3 instances - 1 failure) / 5 private VPC endpoints / 500 CQL
 queries per second = 4 local connections
```

Connexions requises dans la région de l'ouest des États-Unis (Californie du Nord) :

```
20,000 CQL queries / (3 instances - 1 failure) / 2 private VPC endpoints / 500 CQL
 queries per second = 10 local connections
```

> **Important**
>
> Lorsque vous utilisez des points de terminaison VPC privés, des autorisations supplémentaires sont nécessaires pour qu'Amazon Keyspaces découvre dynamiquement les points de terminaison VPC disponibles et renseigne le tableau. `system.peers` Pour plus d'informations, consultez the section called “Remplissage des entrées de `system.peers` table avec les informations de point de terminaison VPC de l'interface”.

Lorsque vous accédez à Amazon Keyspaces via un point de terminaison VPC privé en utilisant un autre point de terminaison Compte AWS, il est probable que vous ne voyiez qu'un seul point de terminaison Amazon Keyspaces. Encore une fois, cela n'a aucun impact sur l'ampleur du débit possible vers Amazon Keyspaces, mais cela peut vous obliger à augmenter le nombre de connexions dans la configuration de votre pilote. Cet exemple montre le même calcul pour un seul point de terminaison disponible.

```
20,000 CQL queries / (3 instances - 1 failure) / 1 private VPC endpoints / 500 CQL
 queries per second = 20 local connections
```

Pour en savoir plus sur l'accès entre comptes à Amazon Keyspaces via un VPC partagé, consultez. the section called “Accès entre comptes dans un VPC partagé”

Comment surveiller les connexions dans Amazon Keyspaces

Pour identifier le nombre de points de terminaison auxquels votre application est connectée, vous pouvez enregistrer le nombre de pairs découverts dans le `system.peers` tableau. L'exemple suivant est un exemple de code Java qui affiche le nombre de pairs une fois la connexion établie.

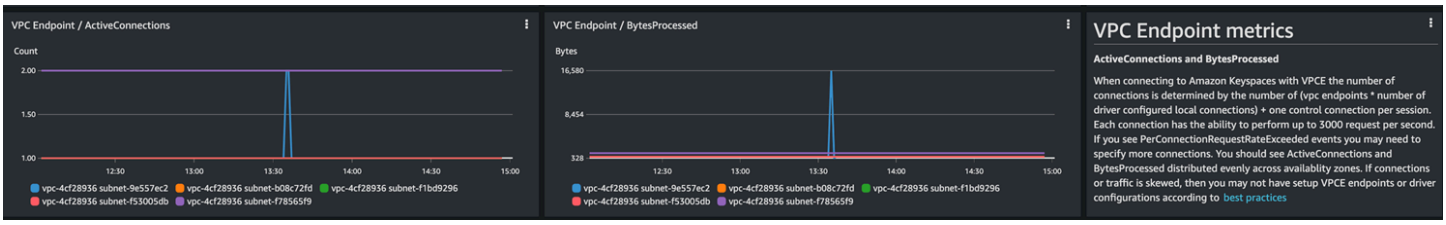
```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

> **Note**
>
> La console ou AWS console CQL n'est pas déployée au sein d'un VPC et utilise donc le point de terminaison public. Par conséquent, l'exécution de la `system.peers` requête à partir d'applications situées en dehors du VPCE se traduit souvent par 9 pairs. Il peut également être utile d'imprimer les adresses IP de chaque homologue.

Vous pouvez également observer le nombre de pairs lorsque vous utilisez un point de terminaison VPC en configurant les métriques Amazon CloudWatch VPCE. Dans CloudWatch, vous pouvez voir le nombre de connexions établies avec le point de terminaison du VPC. Les pilotes Cassandra établissent une connexion pour chaque point de terminaison afin d'envoyer des requêtes CQL et une connexion de contrôle pour recueillir des informations sur les tables système. L'image ci-dessous montre les CloudWatch métriques du point de terminaison VPC après la connexion à Amazon Keyspaces avec une connexion configurée dans les paramètres du pilote. La métrique indique six

connexions actives, dont une connexion de contrôle et cinq connexions (une par point de terminaison dans les zones de disponibilité).



Pour commencer à surveiller le nombre de connexions à l'aide d'un CloudWatch graphique, vous pouvez déployer ce AWS CloudFormation modèle disponible GitHub dans le référentiel de modèles Amazon Keyspaces.

Comment gérer les erreurs de connexion dans Amazon Keyspaces

Lorsque le quota de 3 000 demandes par connexion est dépassé, Amazon Keyspaces renvoie un `PerConnectionRequestExceeded` événement et le pilote Cassandra reçoit une `WriteTimeout` exception ou. `ReadTimeout` Vous devez réessayer cette exception avec un retard exponentiel dans votre politique de réessai de Cassandra ou dans votre application. Vous devez prévoir un délai de traitement exponentiel pour éviter d'envoyer des demandes supplémentaires.

La politique de nouvelles tentatives par défaut tente d'intégrer `try next host` le plan de requête. Comme Amazon Keyspaces peut avoir un à trois points de terminaison disponibles lors de la connexion au point de terminaison VPC, vous pouvez également voir les `ReadTimeout` exceptions `NoHostAvailableException` en plus `WriteTimeout` et dans les journaux de vos applications. Vous pouvez utiliser les politiques de réessai fournies par Amazon Keyspaces, qui permettent de réessayer sur le même point de terminaison mais sur des connexions différentes.

Vous trouverez des exemples de politiques de nouvelles tentatives exponentielles pour Java GitHub dans le référentiel d'exemples de code Java d'Amazon Keyspaces. Vous pouvez trouver des exemples de langage supplémentaires sur Github dans le référentiel d'exemples de code Amazon Keyspaces.

Modélisation des données dans Amazon Keyspaces (pour Apache Cassandra)

Cette rubrique présente les concepts de modélisation des données dans Amazon Keyspaces (pour Apache Cassandra). Utilisez cette section pour trouver des recommandations pour concevoir des modèles de données conformes aux modèles d'accès aux données de votre application. La mise en

œuvre des meilleures pratiques de modélisation des données améliore les performances et minimise les coûts de débit lorsque vous travaillez avec Amazon Keyspaces.

Pour visualiser et concevoir des modèles de données plus facilement, vous pouvez utiliser le NoSQL Workbench.

Rubriques

- Comment utiliser efficacement les clés de partition dans Amazon Keyspaces

Comment utiliser efficacement les clés de partition dans Amazon Keyspaces

La clé primaire qui identifie de manière unique chaque ligne d'une table Amazon Keyspaces peut être composée d'une ou de plusieurs colonnes de clé de partition, qui déterminent les partitions dans lesquelles les données sont stockées, et d'une ou plusieurs colonnes de clustering facultatives, qui définissent la manière dont les données sont regroupées et triées au sein d'une partition.

Étant donné que la clé de partition définit le nombre de partitions dans lesquelles vos données sont stockées et la manière dont les données sont réparties entre ces partitions, le choix de votre clé de partition peut avoir un impact significatif sur les performances de vos requêtes. En général, vous devez concevoir votre application pour une activité uniforme sur toutes les partitions du disque.

La répartition uniforme des activités de lecture et d'écriture de votre application sur toutes les partitions permet de minimiser les coûts de débit, et cela s'applique aux modes de capacité de lecture/écriture à la demande ainsi qu'aux modes de capacité de lecture/écriture provisionnée. Par exemple, si vous utilisez le mode capacité provisionnée, vous pouvez déterminer les modèles d'accès dont votre application a besoin et estimer le nombre total d'unités de capacité de lecture (RCU) et d'unités de capacité d'écriture (WCU) requises par chaque table. Amazon Keyspaces prend en charge vos modèles d'accès en utilisant le débit que vous avez fourni, à condition que le trafic sur une partition donnée ne dépasse pas 3 000 RCU et 1 000 WCU.

Amazon Keyspaces offre une flexibilité supplémentaire dans le provisionnement du débit par partition en fournissant une capacité de rafale. Pour plus d'informations, consultez. the section called “Capacité de débordement”

Rubriques

- Utilisation du partitionnement d'écriture pour répartir les charges de travail de manière uniforme dans Amazon Keyspaces

Utilisation du partitionnement d'écriture pour répartir les charges de travail de manière uniforme dans Amazon Keyspaces

L'un des moyens de mieux répartir les écritures sur une partition dans Amazon Keyspaces consiste à étendre l'espace. Vous pouvez effectuer cette opération de plusieurs manières. Vous pouvez ajouter une colonne de clé de partition supplémentaire dans laquelle vous écrivez des nombres aléatoires pour répartir les lignes entre les partitions. Ou vous pouvez utiliser un nombre qui est calculé en fonction d'une information sur laquelle porte la requête.

Sharding à l'aide de clés de partition composées et de valeurs aléatoires

Une stratégie pour répartir les charges de manière plus uniforme sur une partition consiste à ajouter une colonne de clé de partition supplémentaire dans laquelle vous écrivez des nombres aléatoires. Vous pouvez alors randomiser les écritures sur l'espace plus important.

Par exemple, considérez le tableau suivant qui contient une clé de partition unique représentant une date.

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   title text,
   description int,
   PRIMARY KEY (publish_date));
```

Pour répartir plus uniformément ce tableau entre les partitions, vous pouvez inclure une colonne de clé de partition supplémentaire `shard` qui stocke des nombres aléatoires. Par exemple :

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   shard int,
   title text,
   description int,
   PRIMARY KEY ((publish_date, shard)));
```

Lorsque vous insérez des données, vous pouvez choisir un nombre aléatoire entre `1` et `200` pour la `shard` colonne. Cela produit des valeurs de clé de partition composées telles que `(2020-07-09, 1)(2020-07-09, 2)`,, et ainsi de suite`(2020-07-09, 200)`. Puisque vous randomisez la clé de partition, les écritures quotidiennes sur la table sont réparties uniformément entre plusieurs partitions. Cela permet un meilleur parallélisme et un débit général plus élevé.

Toutefois, pour lire toutes les lignes d'un jour donné, vous devez rechercher toutes les partitions dans les lignes, puis fusionner les résultats. Par exemple, vous devez d'abord émettre une `SELECT` instruction pour la valeur de la clé de partition`(2020-07-09, 1)`. Émettez ensuite une autre `SELECT` déclaration pour`(2020-07-09, 2)`, et ainsi de suite, par`(2020-07-09, 200)`. Enfin, votre application devra fusionner les résultats de toutes ces `SELECT` instructions.

Sharding à l'aide de clés de partition composées et de valeurs calculées

Une stratégie de randomisation peut considérablement améliorer le débit d'écriture. Mais il est difficile de lire une ligne spécifique car vous ne savez pas quelle valeur a été écrite dans la `shard` colonne au moment où la ligne a été écrite. Pour faciliter la lecture des lignes individuelles, vous pouvez utiliser une autre stratégie. Au lieu d'utiliser un nombre aléatoire pour répartir les lignes entre les partitions, utilisez un nombre que vous pouvez calculer en fonction de l'élément sur lequel vous souhaitez effectuer une requête.

Considérons l'exemple précédent, dans lequel une table utilise la date du jour dans la clé de partition. Supposons maintenant que chaque ligne possède une `title` colonne accessible et que vous ayez le plus souvent besoin de rechercher les lignes par titre en plus de leur date. Avant que votre application n'écrive la ligne dans la table, elle peut calculer une valeur de hachage en fonction du titre et l'utiliser pour remplir la `shard` colonne. Le calcul peut se traduire par un nombre compris entre 1 et 200 assez uniformément distribué, à l'image de ce que la stratégie de randomisation produit.

Un simple calcul suffirait probablement, tel que le produit des valeurs des points de code UTF-8 pour les caractères du titre, modulo 200, + 1. La valeur de la clé de partition composée serait alors la combinaison de la date et du résultat du calcul.

Avec cette stratégie, les écritures sont réparties uniformément entre les valeurs de clé de partition, et de ce fait entre les partitions physiques. Vous pouvez facilement exécuter une `SELECT` instruction pour une ligne et une date spécifiques, car vous pouvez calculer la valeur de la clé de partition pour une `title` valeur spécifique.

Pour lire toutes les lignes d'un jour donné, vous devez toujours `SELECT` chacune des `(2020-07-09, N)` clés (1 `N` à 200), et votre application doit ensuite fusionner tous les résultats. L'avantage est que vous évitez qu'une valeur de clé de partition « critique » ne prenne l'ensemble de la charge de travail.

Optimisation des coûts des tables Amazon Keyspaces

Cette section décrit les meilleures pratiques pour optimiser les coûts de vos tables Amazon Keyspaces existantes. Vous devez examiner les stratégies suivantes pour déterminer la stratégie d'optimisation des coûts qui répond le mieux à vos besoins et les aborder de manière itérative. Chaque stratégie fournit une vue d'ensemble de ce qui pourrait avoir un impact sur vos coûts, la manière de rechercher des opportunités d'optimisation des coûts et des conseils prescriptifs sur la manière de mettre en œuvre ces meilleures pratiques pour vous aider à économiser.

Rubriques

Évaluer les coûts au niveau de la table

L'outil Cost Explorer qui se trouve dans le AWS Management Console permet de visualiser les coûts ventilés par type, par exemple les frais de lecture, d'écriture, de stockage et de sauvegarde. Vous pouvez également voir ces coûts résumés par période, par mois ou par jour.

L'un des problèmes courants de Cost Explorer est que vous ne pouvez pas facilement consulter les coûts d'une seule table en particulier, car Cost Explorer ne vous permet pas de filtrer ou de regrouper les coûts d'une table spécifique. Vous pouvez consulter la taille métrique de la table facturable (octets) de chaque table dans l'onglet Monitor de la console Amazon Keyspaces. Si vous avez besoin de plus d'informations relatives aux coûts par table, cette section explique comment utiliser le balisage pour effectuer une analyse des coûts de table individuelle dans Cost Explorer.

Rubriques

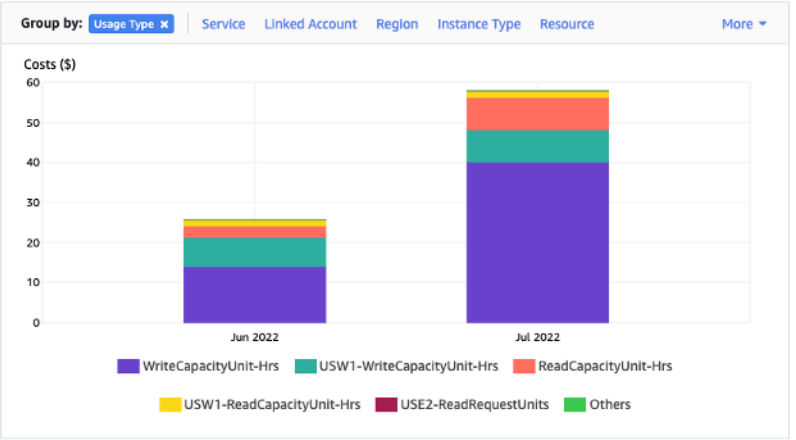
Comment consulter les coûts d'un seul tableau Amazon Keyspaces

Vous pouvez consulter des informations de base sur une table Amazon Keyspaces dans la console, notamment le schéma de clé primaire, la taille de la table facturable et les statistiques relatives à la capacité. Vous pouvez utiliser la taille de la table pour calculer le coût de stockage mensuel de la table. Par exemple, 0,25$ par Go dans l'est des États-Unis (Virginie du Nord) Région AWS.

Si la table utilise le mode capacité provisionnée, les paramètres actuels de l'unité de capacité de lecture (RCU) et de l'unité de capacité d'écriture (WCU) sont également renvoyés. Vous pouvez utiliser ces informations pour calculer les coûts actuels de lecture et d'écriture de la table. Notez que ces coûts peuvent changer, en particulier si vous avez configuré le tableau avec le dimensionnement automatique d'Amazon Keyspaces.

Vue par défaut de Cost Explorer

La vue par défaut de Cost Explorer fournit des graphiques indiquant le coût des ressources consommées, par exemple le débit et le stockage. Vous pouvez choisir de regrouper ces coûts par période, par exemple les totaux par mois ou par jour. Les coûts de stockage, de lecture, d'écriture et d'autres catégories peuvent également être ventilés et comparés.



Comment utiliser et appliquer des balises de table dans Cost Explorer

Par défaut, Cost Explorer ne fournit pas de résumé des coûts pour une table spécifique, car il combine les coûts de plusieurs tables pour obtenir un total. Vous pouvez toutefois utiliser le balisage des ressources AWS pour identifier chaque table à l'aide d'une balise de métadonnées. Les balises sont des paires clé-valeur que vous pouvez utiliser à diverses fins, par exemple pour identifier toutes les ressources appartenant à un projet ou à un département. Pour plus d’informations, consultez the section called “Utilisation des tags”.

Pour cet exemple, nous utilisons une table portant le nom MyTable.

1. Définissez une balise avec la clé table_name et la valeur de. MyTable
2. Activez la balise dans Cost Explorer, puis filtrez sur la valeur de la balise pour obtenir une meilleure visibilité sur les coûts de chaque table.

> **Note**
>
> Cela peut prendre un ou deux jours pour que la balise commence à apparaître dans Cost Explorer

Vous pouvez définir vous-même les balises de métadonnées dans la console ou par programmation avec CQL AWS CLI, le SDK ou le SDK. AWS Envisagez d'exiger la définition d'une balise table_name dans le cadre du nouveau processus de création de tables de votre organisation. Pour plus d'informations, consultez the section called "Rapports de répartition des coûts pour Amazon Keyspaces".

Évaluez le mode capacité de votre table

Cette section explique comment sélectionner le mode de capacité approprié pour votre table Amazon Keyspaces. Chaque mode est réglé de façon à répondre aux besoins d'une charge de travail différente en termes de réactivité face à l'évolution du débit, ainsi que de facturation de cette utilisation. Vous devez tenir compte de ces facteurs au moment de prendre votre décision.

Rubriques

- Quels sont les modes de capacité de table disponibles ?
- Quand sélectionner le mode de capacité à la demande ?
- Quand sélectionner le mode de capacité provisionnée ?
- Autres facteurs à prendre en compte lors du choix d'un mode de capacité de table

Quels sont les modes de capacité de table disponibles ?

Lorsque vous créez une table Amazon Keyspaces, vous devez sélectionner le mode de capacité à la demande ou le mode de capacité provisionnée. Pour plus d'informations, consultez the section called "Modes de capacité de lecture/écriture".

Mode de capacité à la demande

Le mode de capacité à la demande est conçu pour éliminer le besoin de planifier ou de provisionner la capacité de votre table Amazon Keyspaces. Dans ce mode, votre table répond instantanément aux demandes sans qu'il soit nécessaire d'augmenter ou de diminuer les ressources (jusqu'à deux fois le débit maximal précédent de la table).

Les tables à la demande sont facturées en comptant le nombre de demandes réelles par rapport à la table, de sorte que vous ne payez que pour ce que vous utilisez plutôt que pour ce qui a été provisionné.

Mode de capacité provisionnée

Le mode de capacité provisionnée est un modèle plus traditionnel dans lequel vous pouvez définir la capacité dont la table dispose pour les demandes, soit directement, soit à l'aide d'Application Auto Scaling. Dans la mesure où une capacité spécifique est mise en service pour la table à tout moment, la facturation est basée sur la capacité fournie plutôt que sur le nombre de demandes. Le dépassement de la capacité allouée peut également entraîner le rejet des demandes par la table et réduire l'expérience des utilisateurs de votre application.

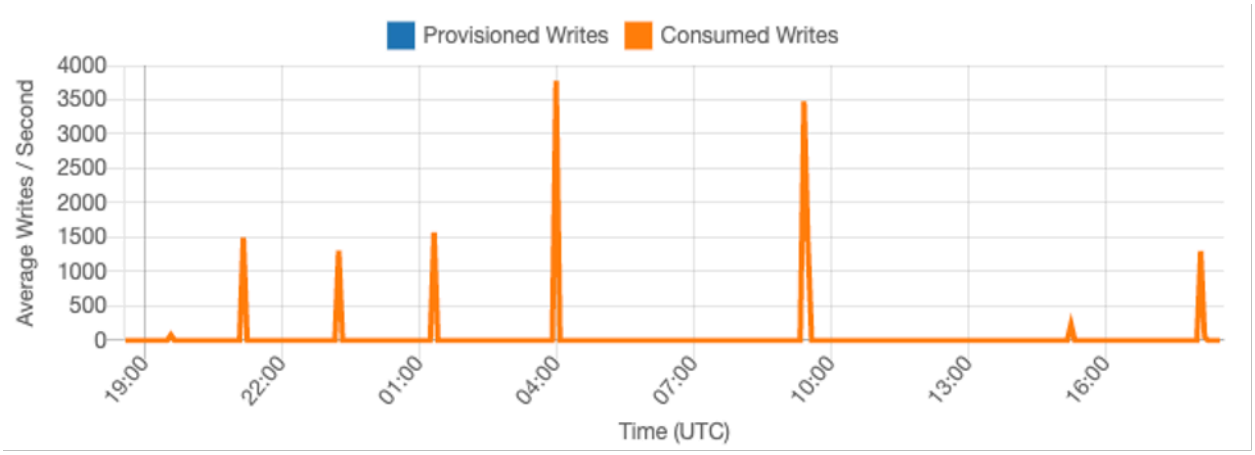
Le mode de capacité provisionnée nécessite un équilibre entre le fait de ne pas surprovisionner ou sous-provisionner la table pour atteindre les deux objectifs, la faible occurrence d'erreurs de capacité de débit insuffisante et l'optimisation des coûts.

Quand sélectionner le mode de capacité à la demande ?

Lorsque vous optimisez en fonction des coûts, le mode à la demande est votre meilleur choix lorsque vous avez une charge de travail imprévisible similaire à celle illustrée dans le graphique suivant.

Les facteurs suivants contribuent à ce type de charge de travail :

- Caractère imprévisible des demandes (entraînant des pics de trafic)
- Volume variable des demandes (en raison des charges de travail par lots)
- Baisse à zéro ou en dessous de 18 % du pic pendant une heure donnée (en raison des environnements de développement ou de test)



Pour les charges de travail présentant les caractéristiques ci-dessus, l'utilisation d'Application Auto Scaling pour maintenir une capacité suffisante pour que la table puisse répondre aux pics de trafic peut entraîner des résultats indésirables. Il se peut que la table soit surprovisionnée et coûte plus cher que nécessaire, soit qu'elle soit sous-approvisionnée et que les demandes entraînent des erreurs de débit inutiles liées à une faible capacité. Dans de tels cas, les tables à la demande sont le meilleur choix.

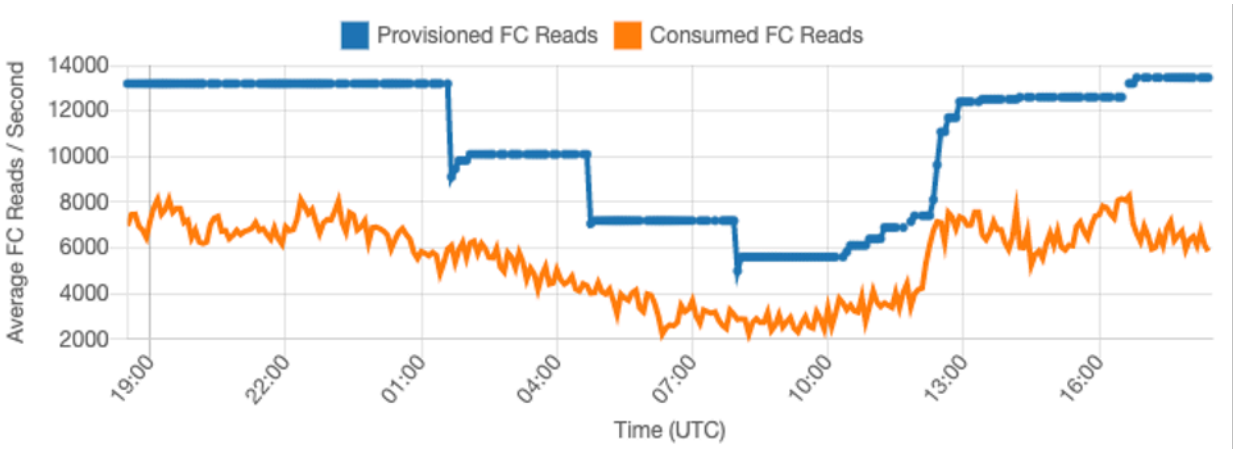
Les tables à la demande étant facturées sur demande, vous n'avez rien d'autre à faire au niveau des tables pour optimiser les coûts. Vous devez régulièrement évaluer vos tables à la demande pour vérifier que la charge de travail présente toujours les caractéristiques ci-dessus. Si la charge de travail s'est stabilisée, envisagez de passer en mode provisionné afin de maintenir l'optimisation des coûts.

Quand sélectionner le mode de capacité provisionnée ?

Une charge de travail idéale pour le mode de capacité allouée est une charge de travail dont le modèle d'utilisation est plus prévisible, comme le montre le graphique ci-dessous.

Les facteurs suivants contribuent à une charge de travail prévisible :

- Trafic prévisible/cyclique pour une heure ou une journée donnée
- Rafales de trafic limitées à court terme



Étant donné que les volumes de trafic sont plus stables à une heure ou à une journée donnée, vous pouvez définir la capacité allouée de manière relativement proche de la capacité réellement consommée de la table. L'optimisation des coûts d'un tableau des capacités provisionnées consiste en fin de compte à faire en sorte que la capacité provisionnée (ligne bleue) soit aussi proche que possible de la capacité consommée (ligne orange) sans augmenter le nombre d'`ThrottledRequests`événements associés au tableau. L'espace entre les deux lignes représente à la fois une perte de capacité et une assurance contre une mauvaise expérience utilisateur due à des erreurs de capacité de débit insuffisante.

Amazon Keyspaces fournit Application Auto Scaling pour les tables de capacité provisionnées, qui équilibre automatiquement ces données en votre nom. Vous pouvez suivre votre capacité consommée tout au long de la journée et configurer la capacité provisionnée de la table en fonction de quelques variables.

Unités de capacité minimale

Vous pouvez définir la capacité minimale d'une table pour limiter les erreurs de capacité de débit insuffisante, mais cela ne réduit pas le coût de la table. Si votre table connaît des périodes de faible utilisation suivies d'une augmentation soudaine d'utilisation, le fait de définir la valeur minimale peut empêcher Application Auto Scaling de définir une capacité trop faible de la table.

Unité de capacité maximale

Vous pouvez définir la capacité maximale d'une table afin de limiter la mise à l'échelle d'une table en utilisant une valeur supérieure à celle prévue. Envisagez d'appliquer un maximum pour les tables de développement ou de test, où les tests de charge à grande échelle ne sont pas souhaités. Vous pouvez définir un maximum pour n'importe quelle table, mais veillez à évaluer régulièrement ce

paramètre par rapport à la base de référence du tableau lorsque vous l'utilisez en production, afin d'éviter les erreurs accidentelles liées à une capacité de débit insuffisante.

Utilisation cible

La définition de l'utilisation cible de la table est le principal moyen d'optimiser les coûts pour une table à capacité provisionnée. La définition d'une valeur en pourcentage inférieure augmente le surprovisionnement de la table, ce qui augmente les coûts, mais réduit le risque d'erreurs liées à une capacité de débit insuffisante. La définition d'une valeur de pourcentage plus élevée réduit le surprovisionnement de la table, mais augmente le risque d'erreurs liées à une capacité de débit insuffisante.

Autres facteurs à prendre en compte lors du choix d'un mode de capacité de table

Au moment de choisir entre les deux modes de capacité, certains facteurs supplémentaires méritent d'être pris en compte.

Lorsque vous choisissez entre les deux modes de table, considérez dans quelle mesure cette réduction supplémentaire affecte le coût de la table. Dans de nombreux cas, même une charge de travail relativement imprévisible peut être plus rentable à exécuter sur une table de capacité provisionnée surdimensionnée avec des capacités réservées.

Améliorer la prévisibilité de votre charge de travail

Dans certaines situations, une charge de travail peut apparemment présenter à la fois un schéma prévisible et un schéma imprévisible. Bien que cela puisse être facilement pris en charge par un tableau à la demande, les coûts seront probablement inférieurs si les modèles imprévisibles de la charge de travail peuvent être améliorés.

Les importations par lots sont l'une des causes les plus fréquentes de ces tendances. Ce type de trafic peut souvent dépasser la capacité de base de la table à un point tel que des erreurs de capacité de débit insuffisante se produiraient en cas d'exécution. Pour qu'une charge de travail comme celle-ci soit exécutée sur une table à capacité provisionnée, considérez les options suivantes :

- Si le lot se produit à des heures planifiées, vous pouvez planifier une augmentation de la capacité d'autodimensionnement de votre application avant son exécution.
- Si le lot se produit de manière aléatoire, pensez à essayer de prolonger le temps d'exécution plutôt que de l'exécuter le plus rapidement possible.

- Ajoutez une période d'accélération à l'importation, au cours de laquelle la vitesse de l'importation commence lentement mais augmente lentement en quelques minutes jusqu'à ce qu'Application Auto Scaling ait eu l'opportunité de commencer à ajuster la capacité de la table.

Évaluez les paramètres Application Auto Scaling de votre table

Cette section explique comment évaluer les paramètres Application Auto Scaling sur vos tables Amazon Keyspaces. Amazon Keyspaces Application Auto Scaling est une fonctionnalité qui gère le débit des tables en fonction du trafic de votre application et de votre indicateur d'utilisation cible. Cela garantit que vos tables disposent de la capacité requise pour vos modèles d'application.

Le service Application Auto Scaling surveille l'utilisation actuelle de votre table et la compare à la valeur d'utilisation cible :`TargetValue`. Il vous indique s'il est temps d'augmenter ou de diminuer la capacité allouée.

Rubriques

- Comprendre les paramètres de votre Application Auto Scaling
- Comment identifier les tables présentant une faible cible d'utilisation (<= 50 %)
- Comment gérer les charges de travail liées aux variations saisonnières
- Comment gérer les pics de charge de travail imprévisibles
- Comment gérer les charges de travail avec des applications liées

Comprendre les paramètres de votre Application Auto Scaling

Pour définir la valeur correcte correspondant à l'utilisation cible, ainsi que l'étape initiale et les valeurs finales, vous devez faire appel à l'équipe des opérations. Cela vous permet de définir correctement les valeurs en fonction de l'utilisation historique de l'application, qui est utilisée pour déclencher les politiques Application Auto Scaling. L'objectif d'utilisation est le pourcentage de votre capacité totale qui doit être atteint pendant un certain temps avant que les règles d'Application Auto Scaling ne s'appliquent.

Lorsque vous définissez un objectif d'utilisation élevé (un objectif d'environ 90 %), cela signifie que votre trafic doit être supérieur à 90 % pendant un certain temps avant que l'Application Auto Scaling ne soit activée. Il est préférable de ne pas utiliser une cible d'utilisation élevée, sauf si votre application est très constante et ne fait l'objet d'aucun pic de trafic.

Lorsque vous définissez un taux d'utilisation très faible (un objectif inférieur à 50 %), cela signifie que votre application doit atteindre 50 % de la capacité allouée avant de déclencher une politique Application Auto Scaling. À moins que le trafic de vos applications n'augmente à un rythme très soutenu, cela se traduit généralement par une capacité inutilisée et un gaspillage de ressources.

Comment identifier les tables présentant une faible cible d'utilisation (<= 50 %)

Vous pouvez utiliser le AWS CLI ou AWS Management Console pour surveiller et identifier les `TargetValues` politiques de votre Application Auto Scaling dans vos ressources Amazon Keyspaces :

AWS CLI

1. Pour afficher la liste complète des ressources, exécutez la commande suivante :

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra
   ```

   Cette commande renverra la liste complète des politiques Application Auto Scaling émises pour n'importe quelle ressource Amazon Keyspaces. Pour récupérer uniquement les ressources d'une table particulière, vous pouvez ajouter le `–resource-id parameter`. Par exemple :

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra --resource-id "keyspace/keyspace-name/table/table-name”
   ```

2. Renvoie uniquement les politiques de dimensionnement automatique pour une table particulière en exécutant la commande suivante

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra --resource-id "keyspace/keyspace-name/table/table-name”
   ```

   Les valeurs des politiques Application Auto Scaling sont mises en évidence ci-dessous. Vous devez vous assurer que la valeur cible est supérieure à 50 % pour éviter le surprovisionnement. Le résultat doit ressembler à ce qui suit :

   ```
   {
       "ScalingPolicies": [
           {
   ```

```
            "PolicyARN": "arn:aws:autoscaling:<region>:<account-
id>:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-scaling-policy",
            "PolicyName": $<full-gsi-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:index:WriteCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "KeyspacesWriteCapacityUtilization"
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:48.641000+10:00"
        },
        {
            "PolicyARN": "arn:aws:autoscaling:<region>:<account-
id>:scalingPolicy:<uuid>:resource/keyspaces/table/table-name/index/<index-
name>:policyName/$<full-gsi-name>-scaling-policy",
            "PolicyName":$<full-table-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:index:ReadCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "CassandraReadCapacityUtilization"
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:47.820000+10:00"
        }
    ]
}
```

AWS Management Console

1. Connectez-vous au AWS Management Console et accédez à la page de CloudWatch service sur Getting Started with the AWS Management Console. Sélectionnez la solution appropriée Région AWS si nécessaire.
2. Dans le volet de navigation, sélectionnez Tables. Sur la page Tables, sélectionnez le nom de la table.
3. Sur la page Détails de la table de l'onglet Capacity, passez en revue les paramètres Application Auto Scaling de votre table.

Si vos valeurs d'utilisation cibles sont inférieures ou égales à 50 %, explorez les métriques d'utilisation de vos tables pour déterminer si elles sont sous-provisionnées ou surprovisionnées.

Comment gérer les charges de travail liées aux variations saisonnières

Envisagez le scénario suivant : votre application fonctionne en dessous d'une valeur moyenne minimale la plupart du temps, mais la cible d'utilisation est faible. Votre application peut donc réagir rapidement aux événements qui se produisent à certaines heures de la journée et vous disposez d'une capacité suffisante pour éviter les ralentissements. Ce scénario est courant avec les applications qui sont très actives pendant les heures normales de bureau (de 9 h à 17 h), mais qui fonctionnent ensuite à un niveau de base en dehors de cette plage horaire. Étant donné que certains utilisateurs commencent à se connecter avant 9 heures du matin, l'application utilise ce seuil bas pour augmenter rapidement afin d'atteindre la capacité requise aux heures de pointe.

Ce scénario peut se présenter comme suit :

- Entre 17 h et 9 h, les unités `ConsumedWriteCapacityUnits` restent entre 90 et 100.
- Les utilisateurs commencent à se connecter à l'application avant 9 heures du matin et les unités de capacité augmentent considérablement (la valeur maximale que vous avez vue est de 1 500 WCU).
- En moyenne, l'utilisation de vos applications varie entre 800 et 1 200 pendant les heures de travail.

Si le scénario précédent s'applique à votre application, envisagez d'utiliser le dimensionnement automatique des applications planifié, dans lequel une règle Application Auto Scaling peut toujours être configurée sur votre table, mais avec une utilisation cible moins agressive qui ne fournit la capacité supplémentaire qu'aux intervalles spécifiques dont vous avez besoin.

Vous pouvez utiliser le AWS CLI pour exécuter les étapes suivantes afin de créer une règle de dimensionnement automatique planifiée qui s'exécute en fonction de l'heure du jour et du jour de la semaine.

1. Enregistrez votre table Amazon Keyspaces en tant que cible évolutive avec. Application Auto Scaling Une cible pouvant être mise à l'échelle avec est une ressource dont Application Auto Scaling peut augmenter ou réduire la capacité.

```
aws application-autoscaling register-scalable-target \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
    --resource-id keyspace/keyspace-name/table/table-name \
    --min-capacity 90 \
    --max-capacity 1500
```

2. Configurez les actions planifiées en fonction de vos besoins.

   Vous avez besoin de deux règles pour couvrir le scénario : l'une pour augmenter la taille et l'autre pour la réduire. La première règle permettant d'étendre l'action planifiée est illustrée dans l'exemple suivant.

```
aws application-autoscaling put-scheduled-action \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
    --resource-id keyspace/keyspace-name/table/table-name \
    --scheduled-action-name my-8-5-scheduled-action \
    --scalable-target-action MinCapacity=800,MaxCapacity=1500 \
    --schedule "cron(45 8 ? * MON-FRI *)" \
    --timezone "Australia/Brisbane"
```

   La deuxième règle permettant de réduire l'action planifiée est illustrée dans cet exemple.

```
aws application-autoscaling put-scheduled-action \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
    --resource-id keyspace/keyspace-name/table/table-name \
    --scheduled-action-name my-5-8-scheduled-down-action \
    --scalable-target-action MinCapacity=90,MaxCapacity=1500 \
    --schedule "cron(15 17 ? * MON-FRI *)" \
    --timezone "Australia/Brisbane"
```

3. Exécutez la commande suivante pour confirmer que les deux règles ont été activées :

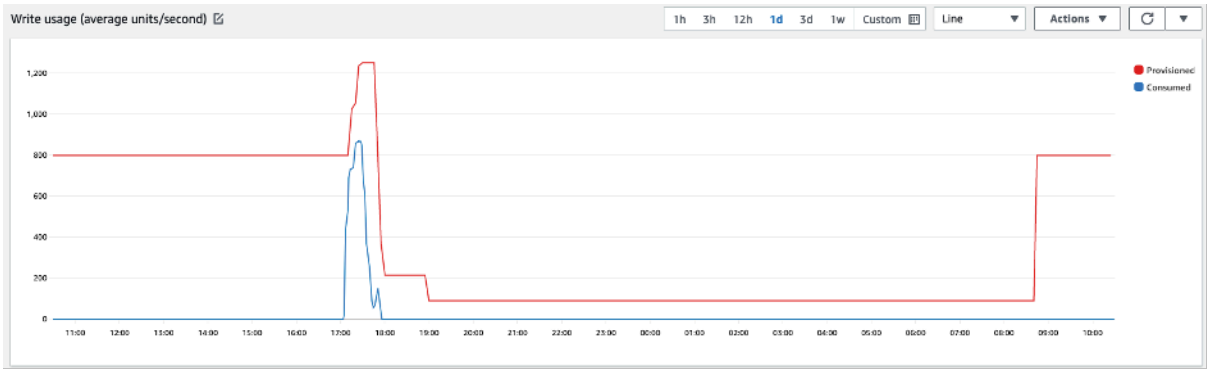
```
aws application-autoscaling describe-scheduled-actions --service-namespace
 cassandra
```

Vous devriez obtenir le résultat suivant :

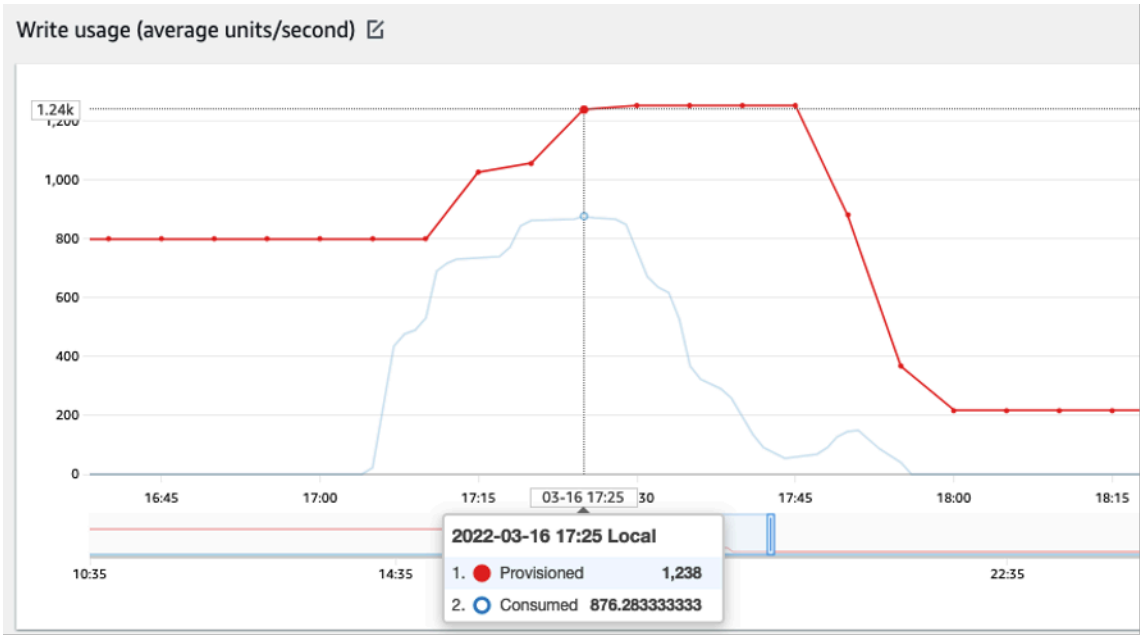
```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-5-8-scheduled-down-action",
            "ScheduledActionARN":
 "arn:aws:autoscaling:<region>:<account>:scheduledAction:<uuid>:resource/keyspaces/
table/table-name:scheduledActionName/my-5-8-scheduled-down-action",
            "ServiceNamespace": "cassandra",
            "Schedule": "cron(15 17 ? * MON-FRI *)",
            "Timezone": "Australia/Brisbane",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:WriteCapacityUnits",
            "ScalableTargetAction": {
                "MinCapacity": 90,
                "MaxCapacity": 1500
            },
            "CreationTime": "2022-03-15T17:30:25.100000+10:00"
        },
        {
            "ScheduledActionName": "my-8-5-scheduled-action",
            "ScheduledActionARN":
 "arn:aws:autoscaling:<region>:<account>:scheduledAction:<uuid>:resource/keyspaces/
table/table-name:scheduledActionName/my-8-5-scheduled-action",
            "ServiceNamespace": "cassandra",
            "Schedule": "cron(45 8 ? * MON-FRI *)",
            "Timezone": "Australia/Brisbane",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:WriteCapacityUnits",
            "ScalableTargetAction": {
                "MinCapacity": 800,
                "MaxCapacity": 1500
            },
            "CreationTime": "2022-03-15T17:28:57.816000+10:00"
        }
    ]
```

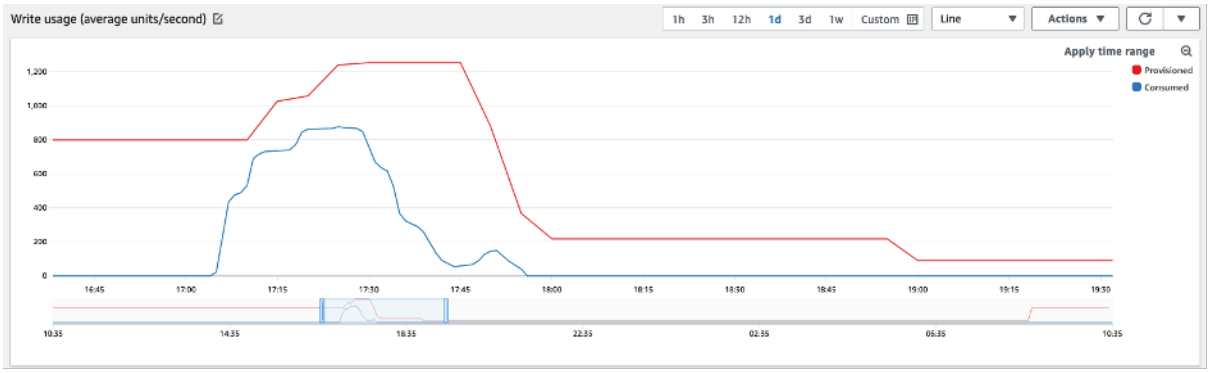
```
}
```

L'image suivante montre un exemple de charge de travail qui maintient en permanence une cible d'utilisation de 70 %. Remarquez que les règles de dimensionnement automatique s'appliquent toujours et que le débit ne diminue pas.



En zoomant, nous pouvons constater qu'un pic dans l'application a déclenché le seuil de 70 %, forçant ainsi la mise à l'échelle automatique à démarrer et à fournir la capacité supplémentaire requise pour la table. L'action de mise à l'échelle automatique planifiée affectera les valeurs maximales et minimales, et il est de votre responsabilité de les configurer.





Comment gérer les pics de charge de travail imprévisibles

Dans ce scénario, l'application utilise un objectif d'utilisation très faible, car vous ne connaissez pas encore les modèles d'application et vous voulez vous assurer que votre charge de travail ne rencontre pas d'erreurs de débit liées à une faible capacité.

Vous pouvez ici envisager d'utiliser le mode de capacité à la demande. Les tables à la demande sont idéales pour les charges de travail exigeantes pour lesquelles vous ne connaissez pas les tendances de trafic. Avec le mode de capacité à la demande, vous payez à la demande les lectures et écritures de données que votre application effectue sur vos tables. Vous n'avez pas besoin de spécifier le débit de lecture et d'écriture que vous souhaitez que votre application atteigne, car Amazon Keyspaces s'adapte instantanément à vos charges de travail à mesure qu'elles augmentent ou diminuent.

Comment gérer les charges de travail avec des applications liées

Dans ce scénario, l'application dépend d'autres systèmes, tels que les scénarios de traitement par lots dans lesquels vous pouvez avoir de forts pics de trafic en fonction des événements dans la logique de l'application.

Envisagez de développer une logique d'auto-scaling des applications personnalisée qui réagit aux événements susceptibles d'augmenter la capacité des tables et `TargetValues` en fonction de vos besoins spécifiques. Vous pourriez bénéficier Amazon EventBridge et utiliser une combinaison de AWS services tels que λ et Step Functions pour répondre aux besoins spécifiques de votre application.

Identifiez vos ressources inutilisées

Cette section explique comment évaluer régulièrement vos ressources inutilisées. Au fur et à mesure que les exigences de votre application évoluent, vous devez vous assurer qu'aucune ressource n'est inutilisée et qu'elle ne contribue à des coûts inutiles liés à Amazon Keyspaces. Les procédures

décrites ci-dessous utilisent CloudWatch les métriques Amazon pour identifier les ressources inutilisées et prendre des mesures pour réduire les coûts.

Vous pouvez surveiller Amazon Keyspaces à l'aide d'Amazon Keyspaces CloudWatch, qui collecte et traite les données brutes d'Amazon Keyspaces pour en faire des indicateurs lisibles en temps quasi réel. Ces statistiques étant conservées pendant un certain temps, vous pouvez accéder aux informations historiques pour acquérir une meilleure compréhension de votre utilisation. Par défaut, les données métriques d'Amazon Keyspaces sont envoyées automatiquement à CloudWatch . Pour plus d'informations, consultez Qu'est-ce qu'Amazon CloudWatch ? et conservation des métriques dans le guide de CloudWatch l'utilisateur Amazon.

Rubriques

Comment identifier les ressources inutilisées

Pour identifier les tables inutilisées, vous pouvez examiner les CloudWatch indicateurs suivants sur une période de 30 jours afin de déterminer s'il existe des lectures ou des écritures actives sur une table spécifique :

`ConsumedReadCapacityUnits`

Nombre d'unités de capacité de lecture consommées sur la période spécifiée, de sorte que vous puissiez suivre la capacité consommée. Vous pouvez récupérer la capacité de lecture totale consommée pour une table.

`ConsumedWriteCapacityUnits`

Nombre d'unités de capacité d'écriture consommées sur la période spécifiée, de sorte que vous puissiez suivre la capacité consommée. Vous pouvez récupérer la capacité d'écriture totale consommée pour une table.

Identification des ressources de table inutilisées

Amazon CloudWatch est un service de surveillance et d'observabilité qui fournit les statistiques du tableau Amazon Keyspaces que vous pouvez utiliser pour identifier les ressources inutilisées.

CloudWatch les métriques peuvent être consultées à AWS Management Console la fois par le biais du AWS Command Line Interface.

AWS Command Line Interface

Pour afficher les statistiques de vos tables via le AWS Command Line Interface, vous pouvez utiliser les commandes suivantes.

1. Tout d'abord, évaluez les lectures de votre table :

> **Note**
>
> Si le nom de la table n'est pas unique dans votre compte, vous devez également spécifier le nom du keyspace.

```
aws cloudwatch get-metric-statistics --metric-name
ConsumedReadCapacityUnits --start-time <start-time> --end-time <end-
time> --period <period> --namespace AWS/Cassandra --statistics Sum --
dimensions Name=TableName,Value=<table-name>
```

Pour éviter de faussement identifier une table comme étant inutilisée, évaluez les indicateurs sur une période plus longue. Choisissez une plage d'heures de début et de fin appropriées, par exemple 30 jours, et une période appropriée, telle que 86400.

Dans les données renvoyées, toute somme supérieure à 0 indique que la table que vous évaluez a reçu du trafic de lecture pendant cette période.

Le résultat suivant montre une table recevant du trafic de lecture au cours de la période évaluée :

```
        {
            "Timestamp": "2022-08-25T19:40:00Z",
            "Sum": 36023355.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-12T19:40:00Z",
            "Sum": 38025777.5,
            "Unit": "Count"
```

```
        },
```

Le résultat suivant montre qu'une table n'a pas reçu de trafic de lecture au cours de la période évaluée :

```
        {
            "Timestamp": "2022-08-01T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-20T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

2. Ensuite, évaluez les écritures dans votre table :

```
aws cloudwatch get-metric-statistics --metric-name
ConsumedWriteCapacityUnits --start-time <start-time> --end-time <end-
time> --period <period> --namespace AWS/Cassandra --statistics Sum --
dimensions Name=TableName,Value=<table-name>
```

Pour éviter de faussement identifier une table comme étant inutilisée, vous devez évaluer les indicateurs sur une période plus longue. Choisissez une plage d'heure de début et de fin appropriée, par exemple 30 jours, et une période appropriée, telle que 86400.

Dans les données renvoyées, toute somme supérieure à 0 indique que la table que vous évaluez a reçu du trafic de lecture pendant cette période.

Le résultat suivant montre une table recevant du trafic d'écriture au cours de la période évaluée :

```
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 41014457.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-18T20:15:00Z",
            "Sum": 40048531.0,
```

```
            "Unit": "Count"
        },
```

Le résultat suivant montre qu'une table n'a pas reçu de trafic d'écriture au cours de la période évaluée :

```
        {
            "Timestamp": "2022-07-31T20:15:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

AWS Management Console

Les étapes suivantes vous permettent d'évaluer l'utilisation de vos ressources à l'aide du AWS Management Console.

1. Connectez-vous au AWS Management Console et accédez à la page de CloudWatch service à l'adresse https://console.aws.amazon.com/cloudwatch/. Sélectionnez l'option appropriée Région AWS en haut à droite de la console, si nécessaire.
2. Dans la barre de navigation de gauche, recherchez la section Mesures et choisissez Toutes les mesures.
3. L'action ci-dessus ouvre un tableau de bord avec deux panneaux. Dans le panneau supérieur, vous pouvez voir les statistiques actuellement représentées graphiquement. En bas, vous pouvez sélectionner les indicateurs disponibles pour le graphe. Choisissez Amazon Keyspaces dans le panneau inférieur.
4. Dans le panneau de sélection des statistiques d'Amazon Keyspaces, choisissez la catégorie Tableau Metrics pour afficher les statistiques relatives à vos tables dans la région actuelle.
5. Identifiez le nom de votre table en faisant défiler le menu vers le bas, puis choisissez les métriques `ConsumedReadCapacityUnits` et `ConsumedWriteCapacityUnits` pour votre tableau.
6. Choisissez l'onglet Mesures graphiques (2) et réglez la colonne Statistiques sur Somme.

7. Pour éviter d'identifier à tort une table comme étant inutilisée, évaluez les métriques de la table sur une période plus longue. En haut du panneau graphique, choisissez une période appropriée, par exemple un mois, pour évaluer votre tableau. Choisissez Personnalisé, choisissez 1 mois dans le menu déroulant, puis choisissez Appliquer.
8. Évaluez les métriques représentées graphiquement pour votre table afin de déterminer si elle est utilisée. Les métriques supérieures à 0 indiquent qu'une table a été utilisée pendant la période évaluée. Un graphique plat à 0 en lecture et en écriture indique qu'une table n'est pas utilisée.

Nettoyage des ressources de table inutilisées

Si vous avez identifié des ressources de table inutilisées, vous pouvez réduire leurs coûts permanents de la manière suivante.

> **Note**
>
> Si vous avez identifié une table non utilisée mais que vous souhaitez la garder à disposition pour un accès futur, pensez à la passer en mode à la demande. Sinon, vous pouvez envisager de supprimer le tableau.

Modes de capacité

Amazon Keyspaces facture la lecture, l'écriture et le stockage de données dans vos tables Amazon Keyspaces.

Amazon Keyspaces propose deux modes de capacité, dotés d'options de facturation spécifiques pour le traitement des lectures et des écritures sur vos tables : à la demande et provisionné. Le mode de capacité en lecture/écriture détermine la façon dont le débit de lecture et écriture vous est facturé et votre façon de gérer la capacité.

Pour les tables en mode à la demande, vous devez spécifier le débit de lecture et d'écriture que votre application est supposée atteindre. Amazon Keyspaces vous facture les opérations de lecture et d'écriture effectuées par votre application sur vos tables en termes d'unités de demande de lecture et d'unités de demande d'écriture. S'il n'y a aucune activité sur votre table, vous ne payez pas pour le débit, mais vous devez tout de même payer des frais de stockage.

Suppression des tables

Si vous avez découvert une table inutilisée et que vous souhaitez la supprimer, pensez d'abord à effectuer une sauvegarde ou à exporter les données.

Les sauvegardes effectuées AWS Backup peuvent tirer parti de la hiérarchisation du stockage à froid, ce qui permet de réduire encore les coûts. Reportez-vous à la documentation sur la gestion des plans de sauvegarde pour savoir comment utiliser un cycle de vie pour transférer votre sauvegarde vers un stockage à froid.

Une fois votre table sauvegardée, vous pouvez choisir de la supprimer via la AWS Management Console ou via l' AWS Command Line Interface.

Nettoyage des sauvegardes de point-in-time restauration non utilisées (PITR)

Amazon Keyspaces propose P oint-in-time Recovery, qui fournit des sauvegardes continues pendant 35 jours afin de vous protéger contre les écritures ou les suppressions accidentelles. Les sauvegardes PITR sont associées à des coûts.

Reportez-vous à la documentation oint-in-time Récupération de données à l'adresse suivante pour déterminer si des sauvegardes sont activées sur vos tables et qu'elles ne sont peut-être plus nécessaires.

Évaluer les modèles d'utilisation d'une table

Cette section explique comment évaluer si vous utilisez efficacement vos tables Amazon Keyspaces. Certains modèles d'utilisation ne sont pas optimaux pour Amazon Keyspaces, et ils peuvent être optimisés tant du point de vue des performances que du point de vue des coûts.

Rubriques

- Effectuer moins d'opérations de lecture fortement cohérente
- Activer la durée de vie (TTL)

Effectuer moins d'opérations de lecture fortement cohérente

Amazon Keyspaces vous permet de configurer la cohérence de lecture pour chaque demande. Les demandes de lecture sont « éventuellement cohérentes » par défaut. Finalement, les lectures cohérentes sont facturées à 0,5 RCU pour un maximum de 4 Ko de données.

La plupart des éléments des charges de travail distribuées sont flexibles et peuvent tolérer une éventuelle cohérence. Cependant, certains modèles d'accès peuvent nécessiter des lectures

fortement cohérentes. Les lectures très cohérentes sont facturées à 1 RCU pour un maximum de 4 Ko de données, ce qui double essentiellement vos coûts de lecture. Amazon Keyspaces vous offre la possibilité d'utiliser les deux modèles de cohérence sur la même table.

Vous pouvez évaluer votre charge de travail et le code de votre application pour vérifier si les lectures fortement cohérentes ne sont utilisées que lorsque cela est nécessaire.

Activer la durée de vie (TTL)

Time to Live (TTL) vous aide à simplifier la logique de votre application et à optimiser le prix du stockage en expirant automatiquement les données des tables. Les données dont vous n'avez plus besoin sont automatiquement supprimées de votre tableau en fonction de la valeur Time to Live que vous avez définie.

Évaluer la capacité allouée pour un dimensionnement approprié

Cette section explique comment évaluer si vous disposez d'un provisionnement de taille appropriée sur vos tables Amazon Keyspaces. À mesure que votre charge de travail évolue, vous devez modifier vos procédures opérationnelles de manière appropriée, en particulier lorsque votre table Amazon Keyspaces est configurée en mode provisionné et que vous courez le risque de surprovisionner ou de sous-approvisionner vos tables.

Les procédures décrites dans cette section nécessitent des informations statistiques qui doivent être capturées à partir des tables Amazon Keyspaces qui prennent en charge votre application de production. Pour comprendre le comportement de votre application, vous devez définir une période suffisamment importante pour saisir le caractère saisonnier des données de votre application. Par exemple, si votre application repose sur des cycles hebdomadaires, spécifier une période de trois semaines devrait vous laisser suffisamment de marge pour analyser ses besoins en matière de débit.

Si vous ne savez pas par où commencer, utilisez au moins un mois de données pour les calculs ci-dessous.

Lors de l'évaluation de la capacité, pour les tables Amazon Keyspaces, vous pouvez configurer les unités de capacité de lecture (RCU) et les unités de capacité d'écriture (WCU) indépendamment.

Rubriques

- Comment récupérer les statistiques de consommation à partir de vos tableaux Amazon Keyspaces
- Comment identifier les tables Amazon Keyspaces sous-approvisionnées
- Comment identifier les tables Amazon Keyspaces surapprovisionnées

Comment récupérer les statistiques de consommation à partir de vos tableaux Amazon Keyspaces

Pour évaluer la capacité de la table, surveillez les CloudWatch mesures suivantes et sélectionnez la dimension appropriée pour récupérer les informations de la table :

Unités de capacité de lecture	Unités de capacité d'écriture
`ConsumedReadCapacityUnits`	`ConsumedWriteCapacityUnits`
`ProvisionedReadCapacityUnits`	`ProvisionedWriteCapacityUnits`
`ReadThrottleEvents`	`WriteThrottleEvents`

Vous pouvez le faire via le AWS CLI ou le AWS Management Console.

AWS CLI

Avant de récupérer les métriques de consommation du tableau, vous devez commencer par capturer certains points de données historiques à l'aide de l' CloudWatch API.

Commencez par créer deux fichiers : `write-calc.json` et `read-calc.json`. Ces fichiers représentent les calculs de la table. Vous devez mettre à jour certains champs, comme indiqué dans le tableau ci-dessous, pour les adapter à votre environnement.

> **Note**
>
> Si le nom de la table n'est pas unique dans votre compte, vous devez également spécifier le nom du keyspace.

Nom de champ	Définition	Exemple
`<table-name>`	Le nom de la table que vous analysez	SampleTable

Nom de champ	Définition	Exemple
`<period>`	Période que vous utilisez pour évaluer l'objectif d'utilisation, en secondes	Pour une période d'une heure, vous devez spécifier : 3 600
`<start-time>`	Début de votre intervalle d'évaluation, spécifié au format ISO8601	2022-02-21T23:00:00
`<end-time>`	Fin de votre intervalle d'évaluation, spécifié au format ISO8601	2022-02-22T06:00:00

Le fichier de calculs d'écriture récupère le nombre de WCU provisionnés et consommés au cours de la période correspondant à la plage de dates spécifiée. Il génère également un pourcentage d'utilisation qui peut être utilisé pour l'analyse. Le contenu complet du `write-calc.json` fichier doit ressembler à celui de l'exemple suivant.

```
{
  "MetricDataQueries": [
    {
      "Id": "provisionedWCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ProvisionedWriteCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
              "Value": "<table-name>"
            }
          ]
        },
        "Period": <period>,
        "Stat": "Average"
      },
      "Label": "Provisioned",
      "ReturnData": false
    },
```

```
    {
      "Id": "consumedWCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ConsumedWriteCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
              "Value": "<table-name>""
            }
          ]
        },
        "Period": <period>,
        "Stat": "Sum"
      },
      "Label": "",
      "ReturnData": false
    },
    {
      "Id": "m1",
      "Expression": "consumedWCU/PERIOD(consumedWCU)",
      "Label": "Consumed WCUs",
      "ReturnData": false
    },
    {
      "Id": "utilizationPercentage",
      "Expression": "100*(m1/provisionedWCU)",
      "Label": "Utilization Percentage",
      "ReturnData": true
    }
  ],
  "StartTime": "<start-time>",
  "EndTime": "<end-time>",
  "ScanBy": "TimestampDescending",
  "MaxDatapoints": 24
}
```

Le fichier de calculs de lecture utilise une métrique similaire. Ce fichier récupère le nombre de RCU qui ont été provisionnées et consommées pendant la période correspondant à la plage de dates spécifiée. Le contenu du `read-calc.json` fichier doit ressembler à celui de cet exemple.

```
{
```

```
  "MetricDataQueries": [
    {
      "Id": "provisionedRCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ProvisionedReadCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
              "Value": "<table-name>"
            }
          ]
        },
        "Period": <period>,
        "Stat": "Average"
      },
      "Label": "Provisioned",
      "ReturnData": false
    },
    {
      "Id": "consumedRCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ConsumedReadCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
              "Value": "<table-name>"
            }
          ]
        },
        "Period": <period>,
        "Stat": "Sum"
      },
      "Label": "",
      "ReturnData": false
    },
    {
      "Id": "m1",
      "Expression": "consumedRCU/PERIOD(consumedRCU)",
      "Label": "Consumed RCUs",
      "ReturnData": false
```

```
    },
    {
      "Id": "utilizationPercentage",
      "Expression": "100*(m1/provisionedRCU)",
      "Label": "Utilization Percentage",
      "ReturnData": true
    }
  ],
  "StartTime": "<start-time>",
  "EndTime": "<end-time>",
  "ScanBy": "TimestampDescending",
  "MaxDatapoints": 24
}
```

Une fois les fichiers créés, vous pouvez commencer à récupérer les données d'utilisation.

1. Pour récupérer les données d'utilisation de l'écriture, exécutez la commande suivante :

   ```
   aws cloudwatch get-metric-data --cli-input-json file://write-calc.json
   ```

2. Pour récupérer les données d'utilisation en lecture, exécutez la commande suivante :

   ```
   aws cloudwatch get-metric-data --cli-input-json file://read-calc.json
   ```

Le résultat des deux requêtes est une série de points de données au format JSON qui peuvent être utilisés pour l'analyse. Vos résultats dépendent du nombre de points de données que vous avez spécifiés, de la période et de vos propres données de charge de travail. Cela pourrait ressembler à l'exemple suivant.

```
{
    "MetricDataResults": [
        {
            "Id": "utilizationPercentage",
            "Label": "Utilization Percentage",
            "Timestamps": [
                "2022-02-22T05:00:00+00:00",
                "2022-02-22T04:00:00+00:00",
                "2022-02-22T03:00:00+00:00",
                "2022-02-22T02:00:00+00:00",
                "2022-02-22T01:00:00+00:00",
                "2022-02-22T00:00:00+00:00",
```

```
                "2022-02-21T23:00:00+00:00"
            ],
            "Values": [
                91.55364583333333,
                55.066631944444445,
                2.6114930555555556,
                24.9496875,
                40.94725694444445,
                25.61819444444444,
                0.0
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

> **Note**
>
> Si vous spécifiez une courte période et une longue période, vous devrez peut-être modifier la `MaxDatapoints` valeur, qui est définie par défaut sur 24 dans le script. Cela représente un seul point de données par heure et donc 24 points de données par jour.

AWS Management Console

1. Connectez-vous au AWS Management Console et accédez à la page de CloudWatch service sur Getting Started with the AWS Management Console. Sélectionnez la solution appropriée Région AWS si nécessaire.
2. Localisez la section Mesures dans la barre de navigation de gauche et choisissez Toutes les mesures.
3. Cela ouvre un tableau de bord avec deux panneaux. Le panneau supérieur affiche le graphique, tandis que le panneau inférieur contient les indicateurs que vous souhaitez représenter graphiquement. Choisissez le panneau Amazon Keyspaces.
4. Choisissez la catégorie Table Metrics dans les sous-panneaux. Cela vous montre les tables de votre compte actuel Région AWS.
5. Identifiez le nom de votre table en faisant défiler le menu vers le bas, puis sélectionnez les métriques des opérations d'écriture : `ConsumedWriteCapacityUnits` et `ProvisionedWriteCapacityUnits`.

> **Note**
>
> Cet exemple décrit les métriques des opérations d'écriture, mais vous pouvez également suivre ces étapes pour représenter graphiquement les métriques des opérations de lecture.

6. Sélectionnez l'onglet Graphed metrics (2) (Indicateurs graphiques (2)) pour modifier les formules. Par défaut, CloudWatch choisit la fonction statistique Average pour les graphiques.
7. Lorsque les deux mesures représentées dans le graphique sont sélectionnées (case à cocher sur la gauche), sélectionnez le menu Add math (Ajouter une formule), suivi de Common (Commun), puis sélectionnez la fonction Percentage (Pourcentage). Répétez cette procédure deux fois.

   Sélection de la fonction Pourcentage pour la première fois.

   Sélection de la fonction Pourcentage pour la deuxième fois.
8. À ce stade, vous devriez avoir quatre métriques dans le menu inférieur. Travaillons sur le calcul de `ConsumedWriteCapacityUnits`. Par souci de cohérence, vous devez faire correspondre les noms à ceux que vous avez utilisés dans la AWS CLI section. Cliquez sur l'ID m1 et remplacez cette valeur par consumedWCU.
9. Remplacez la statistique Average (Moyenne) par Sum (Somme). Cette action crée automatiquement une autre métrique appelée ANOMALY_DETECTION_BAND. Dans le cadre de cette procédure, vous pouvez l'ignorer en décochant la case sur la métrique ad1 nouvellement générée.
10. Répétez l'étape 8 pour remplacer le nom de l'ID m2 par provisionedWCU. Conservez la statistique Average (Moyenne).
11. Choisissez l'étiquette Expression1 et mettez à jour la valeur en m1 et l'étiquette en WCU consommées.

> **Note**
>
> Assurez-vous de n'avoir sélectionné que m1 (case à cocher sur la gauche) et provisionedWCU pour visualiser correctement les données. Mettez à jour la formule en cliquant sur Details (Détails) et en remplaçant la formule par consumedWCU/PERIOD(consumedWCU). Cette étape peut également générer une autre métrique

ANOMALY_DETECTION_BAND, mais dans le cadre de cette procédure, vous pouvez l'ignorer.

12. Vous devriez maintenant avoir deux graphiques : l'un qui indique vos WCU provisionnés dans le tableau et l'autre qui indique les WCU consommés.
13. Mettez à jour la formule du pourcentage en sélectionnant le graphique Expression2 (e2). Remplacez le nom des libellés et des identifiants par utilizationPercent. Remplacez le nom de la formule par 100*(m1/provisionedWCU).
14. Supprimez la case à cocher de toutes les métriques, à l'exception de UtilizationPercentage, pour visualiser vos modèles d'utilisation. L'intervalle par défaut est fixé à 1 minute, mais n'hésitez pas à le modifier selon vos besoins.

Les résultats que vous obtenez dépendent des données réelles de votre charge de travail. Les intervalles d'utilisation supérieurs à 100 % sont sujets à des événements d'erreur liés à une faible capacité de débit. Amazon Keyspaces propose une capacité en rafale, mais dès qu'elle est épuisée, toute valeur supérieure à 100 % est confrontée à des événements d'erreur liés à une faible capacité de débit.

Comment identifier les tables Amazon Keyspaces sous-approvisionnées

Pour la plupart des charges de travail, une table est considérée comme sous-provisionnée lorsqu'elle consomme constamment plus de 80 % de sa capacité provisionnée.

La capacité en rafale est une fonctionnalité d'Amazon Keyspaces qui permet aux clients de consommer temporairement plus de RCU/WCU que ce qui était initialement prévu (plus que le débit provisionné par seconde défini dans le tableau). La capacité de débordement a été créée pour absorber les augmentations soudaines du trafic dues à des événements spéciaux ou à des pics d'utilisation. Cette capacité de rafale est limitée. Pour plus d'informations, voirthe section called “Capacité de débordement”. Dès que les RCU et WCU inutilisés sont épuisés, vous pouvez rencontrer des erreurs de débit en cas de faible capacité si vous essayez de consommer plus de capacité que celle allouée. Lorsque le trafic de votre application atteint un taux d'utilisation proche de 80 %, le risque de rencontrer des erreurs liées au débit de faible capacité est nettement plus élevé.

La règle du taux d'utilisation de 80 % varie en fonction de la saisonnalité de vos données et de la croissance du trafic. Réfléchissez aux scénarios suivants :

- Si le trafic est resté stable à un taux d'utilisation d'environ 90 % au cours des 12 derniers mois, votre table dispose de la capacité idéale
- Si le trafic de vos applications augmente à un rythme de 8 % par mois en moins de 3 mois, vous allez atteindre une utilisation de 100 %
- Si le trafic de vos applications augmente à un rythme de 5 % en un peu plus de 4 mois, vous allez tout de même atteindre une utilisation de 100 %

Les résultats des requêtes ci-dessus donnent une idée de votre taux d'utilisation. Utilisez-les comme guide pour évaluer plus en détail d'autres métriques qui pourront vous aider à choisir d'augmenter la capacité de votre table selon vos besoins (par exemple, à un taux de croissance mensuel ou hebdomadaire). Travaillez avec l'équipe des opérations pour définir le pourcentage approprié pour votre charge de travail et vos tables.

Il existe des scénarios spéciaux dans lesquels les données sont faussées lorsque vous les analysez sur une base quotidienne ou hebdomadaire. Par exemple, dans le cas des applications saisonnières dont l'utilisation augmente pendant les heures de travail (mais qui tombent ensuite à presque zéro en dehors des heures de travail), vous pourriez tirer parti de la planification de l'auto-scaling des applications, dans laquelle vous spécifiez les heures de la journée (et les jours de la semaine) pour augmenter la capacité allouée, ainsi que le moment de la réduire. Au lieu de viser une capacité accrue afin de couvrir les heures de pointe, vous pouvez également bénéficier des configurations d'auto-scaling des tables Amazon Keyspaces si votre saisonnalité est moins prononcée.

Comment identifier les tables Amazon Keyspaces surapprovisionnées

Les résultats de requête obtenus à partir des scripts ci-dessus fournissent les points de données nécessaires pour effectuer une analyse initiale. Si votre ensemble de données présente des valeurs d'utilisation inférieures à 20 % pendant plusieurs intervalles, votre table est peut-être surprovisionnée. Pour déterminer plus précisément si vous devez réduire le nombre de WCU et de RCU, réexaminez les autres lectures dans les intervalles.

Lorsque votre table contient plusieurs intervalles d'utilisation réduits, vous pouvez tirer parti des politiques Application Auto Scaling, soit en planifiant Application Auto Scaling, soit en configurant simplement les politiques Application Auto Scaling par défaut pour la table, en fonction de l'utilisation.

Si votre charge de travail présente un faible rapport utilisation/accélération élevé (Max () /Min (ThrottleEvents) dans l'intervalleThrottleEvents), cela peut se produire lorsque votre charge de travail est très élevée et que le trafic augmente de manière significative certains jours (ou à certaines

heures de la journée), mais qu'il est par ailleurs constamment faible. Dans ces scénarios, il peut être avantageux d'utiliser Application Auto Scaling programmé.

Utilisation NoSQL Cassandra) Keyspaces NoSQL Cassandra)

NoSQL Workbench est une application côté client qui vous permet de concevoir et de visualiser plus facilement des modèles de données non relationnels pour Amazon Keyspaces. Les clients NoSQL Workbench sont disponibles pour Windows, macOS et Linux.

Conception de modèles de données et création automatique de ressources

NoSQL Workbench fournit une point-and-click interface pour concevoir et créer des modèles de données Amazon Keyspaces. Vous pouvez facilement créer de nouveaux modèles de données à partir de zéro en définissant des espaces clés, des tables et des colonnes. Vous pouvez également importer des modèles de données existants et apporter des modifications (par exemple, ajouter, modifier ou supprimer des colonnes) afin d'adapter les modèles de données aux nouvelles applications. NoSQL Workbench vous permet ensuite de valider les modèles de données dans Amazon Keyspaces ou Apache Cassandra, et de créer automatiquement les espaces clés et les tables. Pour savoir comment créer des modèles de données, consultezthe section called “Modélisateur de données”.

Visualisation de de de de de de de

À l'aide de NoSQL Workbench, vous pouvez visualiser vos modèles de données afin de vous assurer qu'ils peuvent prendre en charge les requêtes et les modèles d'accès de votre application. Vous pouvez également enregistrer et exporter vos modèles de données dans différents formats à des fins de collaboration, de documentation et de présentations. Pour plus d'informations, veuillez consulter the section called “Visualiseur de données”.

Rubriques

Téléchargement de NoSQL Workbench

Suivez ces instructions Workbench pour télécharger et installer NoSQL Workbench.

Pour télécharger et installer NoSQL Workbench

1. Utilisez l'un des liens suivants pour télécharger gratuitement NoSQL Workbench.

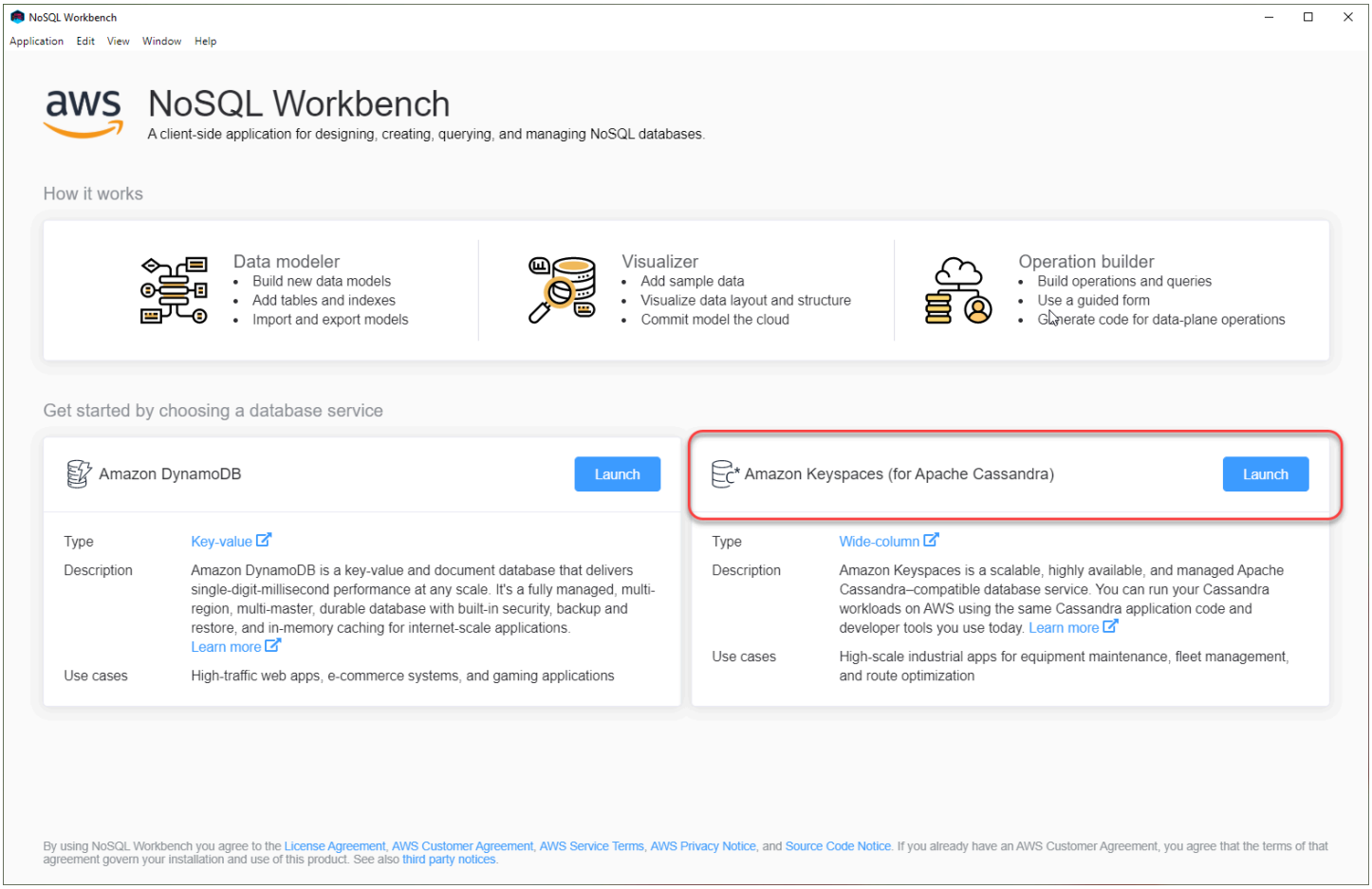
Système d'exploitation	Lien de téléchargement
macOS	Télécharger pour macOS
Linux*	Téléchargement pour Linux
Windows	Télécharger pour Windows

   * NoSQL Workbench prend en charge Ubuntu 12.04, Fedora 21 et Debian 8, ou toute version plus récente de ces distributions Linux.

2. Une fois le téléchargement terminé, démarrez l'application et suivez les instructions qui s'affichent à l'écran pour terminer l'installation.

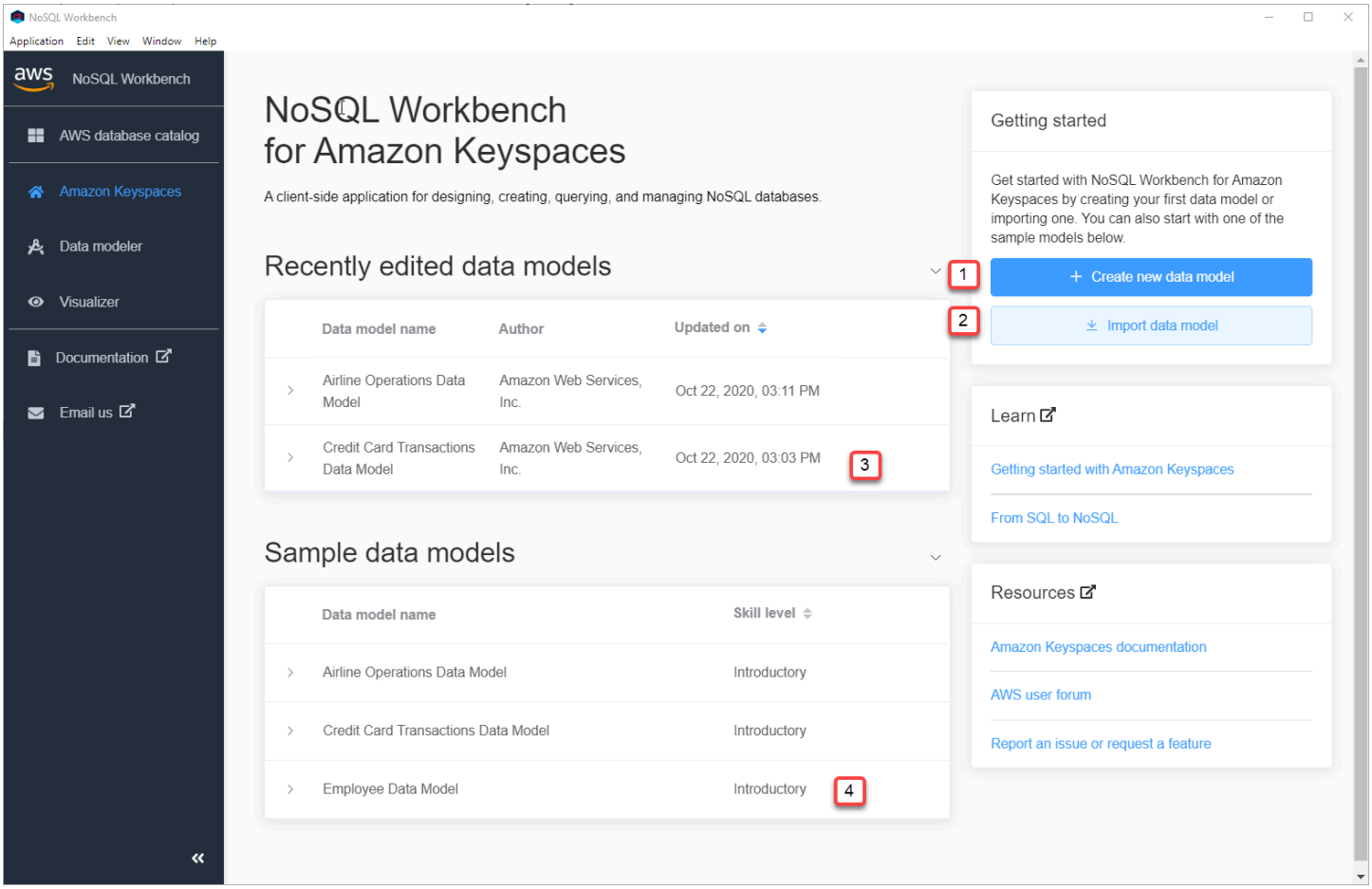
Débuter avec NoSQL Workbench avec NoSQL Workbench de de

Pour démarrer avec NoSQL Workbench, sur la page du catalogue de bases de données de NoSQL Workbench, choisissez Amazon Keyspaces, puis choisissez Launch.



La page d'accueil de NoSQL Workbench pour Amazon Keyspaces s'ouvre. Vous y trouverez les options suivantes pour commencer :

1. Créez d'un modèle de de de de de de de de de
2. Importez un modèle de de de de de de de de de de de de de de de
3. Ouvrez un modèle de données récemment modifié.
4. Ouvrez l'un des exemples de modèles disponibles.



Chacune des options ouvre le modeleur de données NoSQL Workbench. Pour continuer à créer un nouveau modèle de données, consultezthe section called “Création d'un modèle de données”. Pour modifier un modèle de données existant, consultezthe section called “Modification d'un modèle de données”.

Comment créer des modèles de données

Vous pouvez utiliser le modélisateur de données NoSQL Workbench pour concevoir de nouveaux modèles de données en fonction des modèles d'accès aux données de votre application. Vous pouvez utiliser le modélisateur de données pour concevoir de nouveaux modèles de données ou importer et modifier des modèles de données existants créés à l'aide de NoSQL Workbench. Le modélisateur de données inclut également quelques exemples de modèles de données pour vous aider à démarrer avec la modélisation des données.

Rubriques

- Création de nouveaux modèles de données avec NoSQL Workbench

- Modification de modèles de données existants avec NoSQL Workbench

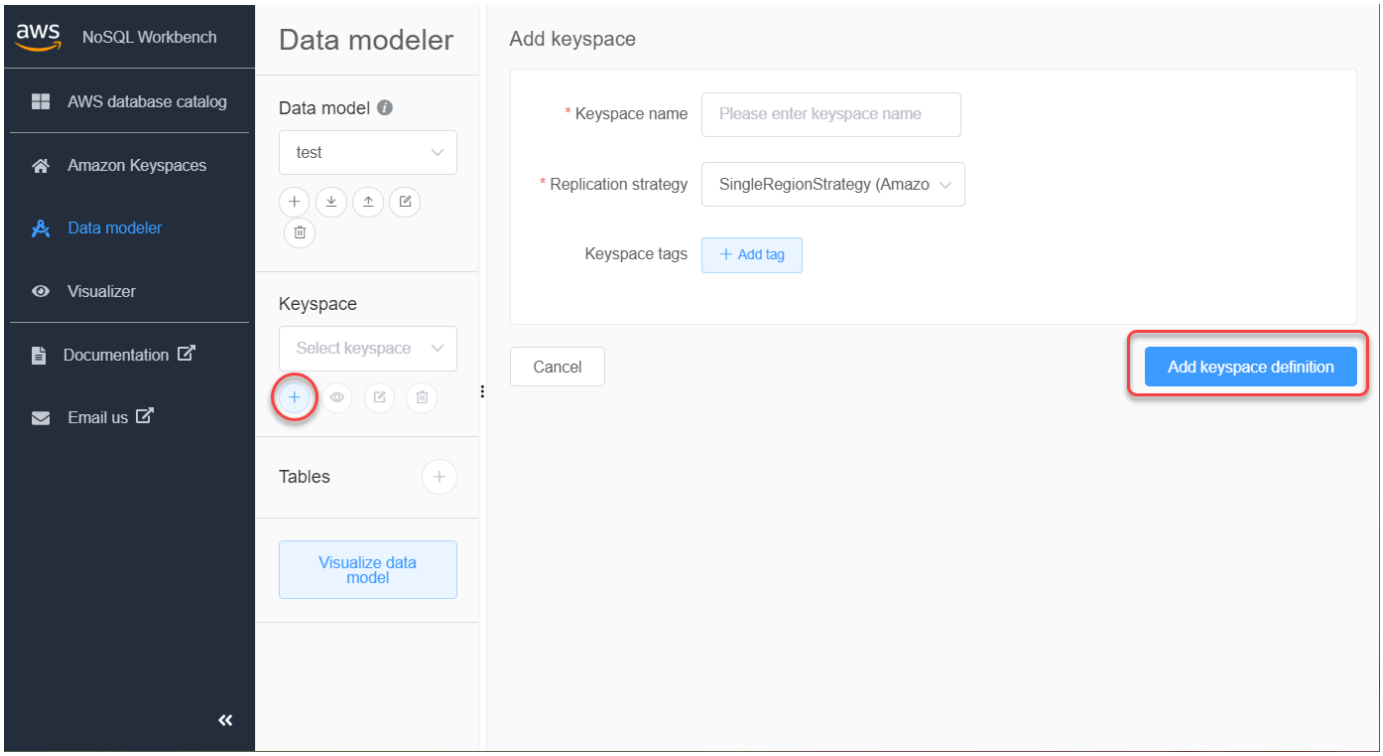
Création de nouveaux modèles de données avec NoSQL Workbench

Pour créer un nouveau modèle de données pour Amazon Keyspaces, vous pouvez utiliser le modeleur de données NoSQL Workbench pour créer des espaces clés, des tables et des colonnes. Procédez comme suit pour créer un modèle de données.

1. Pour créer un nouvel espace de touches, choisissez le signe plus sous Keyspace.

   Au cours de cette étape, choisissez les propriétés et les paramètres suivants.

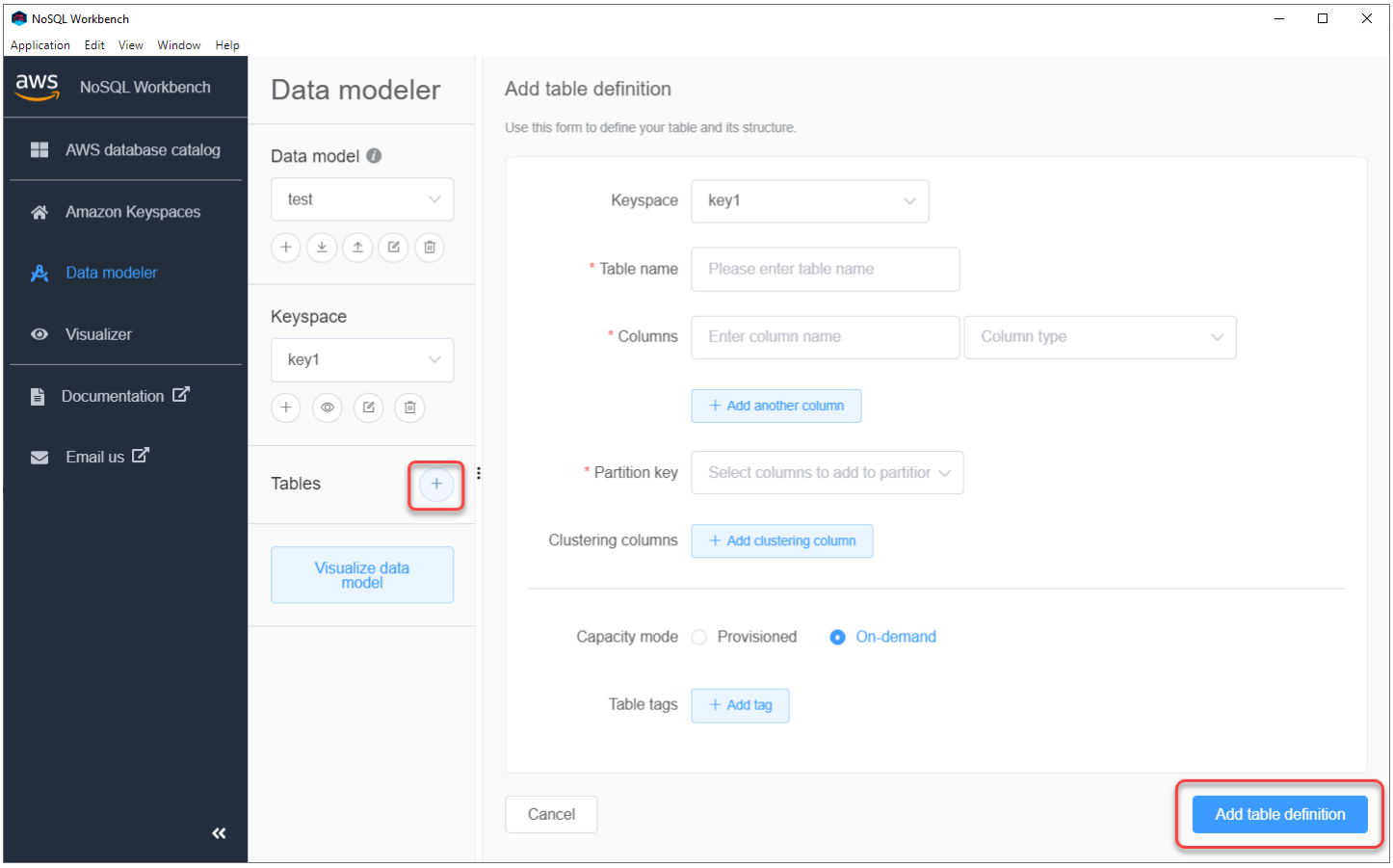
   - Nom de l'espace clé : entrez le nom du nouvel espace de touches.
   - Stratégie de réplication : choisissez la stratégie de réplication pour l'espace clé. Amazon Keyspaces utilise le SingleRegionStrategypour répliquer les données trois fois automatiquement dans plusieurs zones deAWS disponibilité. Si vous envisagez de valider le modèle de données dans un cluster Apache Cassandra, vous pouvez choisir SimpleStrategyou NetworkTopologyStrategy.
   - Balises d'Keyspaces : les balises de ressources sont facultatives et vous permettent de classer vos ressources de différentes manières, par exemple, par objectif, par propriétaire, par environnement ou selon d'autres critères. Pour en savoir plus sur les balises des ressources Amazon Keyspaces, consultezthe section called “Utilisation des tags”.
2. Choisissez Ajouter une définition d'espace clé pour créer l'espace clé.



3. Pour créer un nouveau tableau, cliquez sur le signe plus à côté de Tables. Au cours de cette étape, vous définissez les propriétés et les paramètres suivants.

   - Nom de la table : nom de la nouvelle table.
   - Colonnes : ajoutez un nom de colonne et choisissez le type de données. Répétez ces étapes pour chaque colonne de votre schéma.
   - Clé de partition : choisissez les colonnes pour la clé de partition.
   - Colonnes de regroupement : choisissez de regrouper les colonnes (facultatif).
   - Mode de capacité : choisissez le mode de capacité de lecture/écriture de la table. Vous pouvez choisir une capacité provisionnée ou à la demande. Pour en savoir plus sur les modes de capacité, veuillez consulterthe section called “Modes de capacité de lecture/écriture”.
   - Balises de tableau : les balises de ressources sont facultatives et vous permettent de classer vos ressources de différentes manières, par exemple, par objectif, par propriétaire, par environnement ou selon d'autres critères. Pour en savoir plus sur les balises des ressources Amazon Keyspaces, consultezthe section called “Utilisation des tags”.

4. Choisissez Ajouter une définition de table pour créer la nouvelle table.
5. Répétez ces étapes pour créer des tables supplémentaires.
6. Continuezthe section called “Visualisation d'un modèle de données” à visualiser le modèle de données que vous avez créé.



Modification de modèles de données existants avec NoSQL Workbench

Le modélisateur de données NoSQL Workbench vous permet de modifier des modèles de données existants dans Amazon Keyspaces. Il peut s'agir de modèles de données importés à partir d'un fichier, des exemples de modèles de données fournis ou de modèles de données que vous avez créés précédemment.

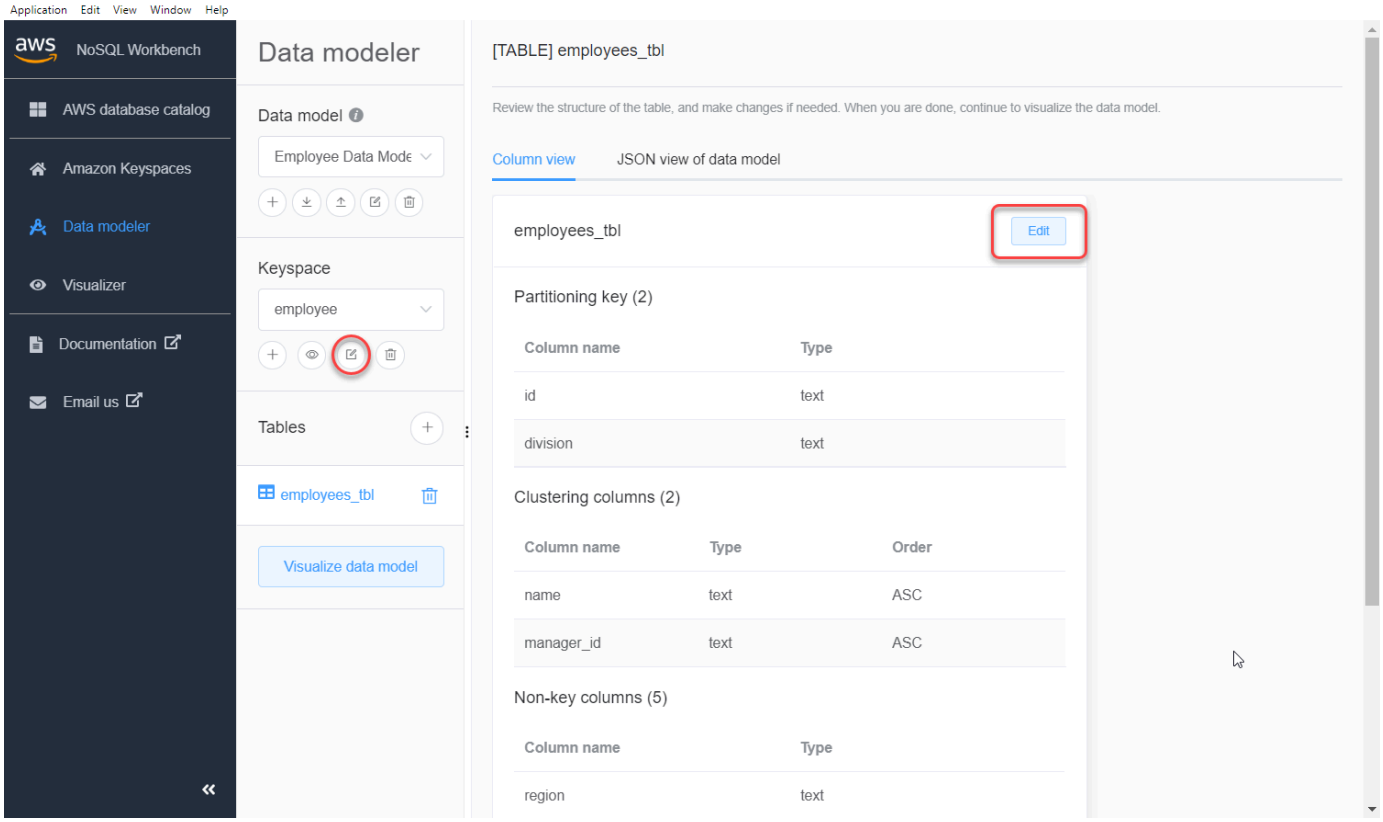
1. Pour modifier un espace clé, choisissez le symbole d'édition sous Keyspace.

   Au cours de cette étape, vous pouvez modifier les propriétés et les paramètres suivants.

   - Nom de l'espace clé : entrez le nom du nouvel espace de touches.
   - Stratégie de réplication : choisissez la stratégie de réplication pour l'espace clé. Amazon Keyspaces utilise le SingleRegionStrategypour répliquer les données trois fois automatiquement dans plusieurs zones deAWS disponibilité. Si vous envisagez de valider le modèle de données dans un cluster Apache Cassandra, vous pouvez choisir SimpleStrategyou NetworkTopologyStrategy.

- Balises d'Keyspaces : les balises de ressources sont facultatives et vous permettent de classer vos ressources de différentes manières, par exemple, par objectif, par propriétaire, par environnement ou selon d'autres critères. Pour en savoir plus sur les balises des ressources Amazon Keyspaces, consultezthe section called “Utilisation des tags”.

2. Choisissez Enregistrer les modifications pour mettre à jour l'espace de touches.



3. Pour modifier une table, choisissez Modifier à côté du nom de la table. Au cours de cette étape, vous pouvez mettre à jour les propriétés et les paramètres suivants.

- Nom de la table : nom de la nouvelle table.
- Colonnes : ajoutez un nom de colonne et choisissez le type de données. Répétez ces étapes pour chaque colonne de votre schéma.
- Clé de partition : choisissez les colonnes pour la clé de partition.
- Colonnes de regroupement : choisissez de regrouper les colonnes (facultatif).
- Mode de capacité : choisissez le mode de capacité de lecture/écriture de la table. Vous pouvez choisir une capacité provisionnée ou à la demande. Pour en savoir plus sur les modes de capacité, veuillez consulterthe section called “Modes de capacité de lecture/écriture”.
- Balises de tableau : les balises de ressources sont facultatives et vous permettent de classer vos ressources de différentes manières, par exemple, par objectif, par propriétaire, par

environnement ou selon d'autres critères. Pour en savoir plus sur les balises des ressources Amazon Keyspaces, consultezthe section called “Utilisation des tags”.

4. Choisissez Enregistrer les modifications pour mettre à jour le tableau.
5. Continuezthe section called “Visualisation d'un modèle de données” à visualiser le modèle de données que vous avez mis à jour.

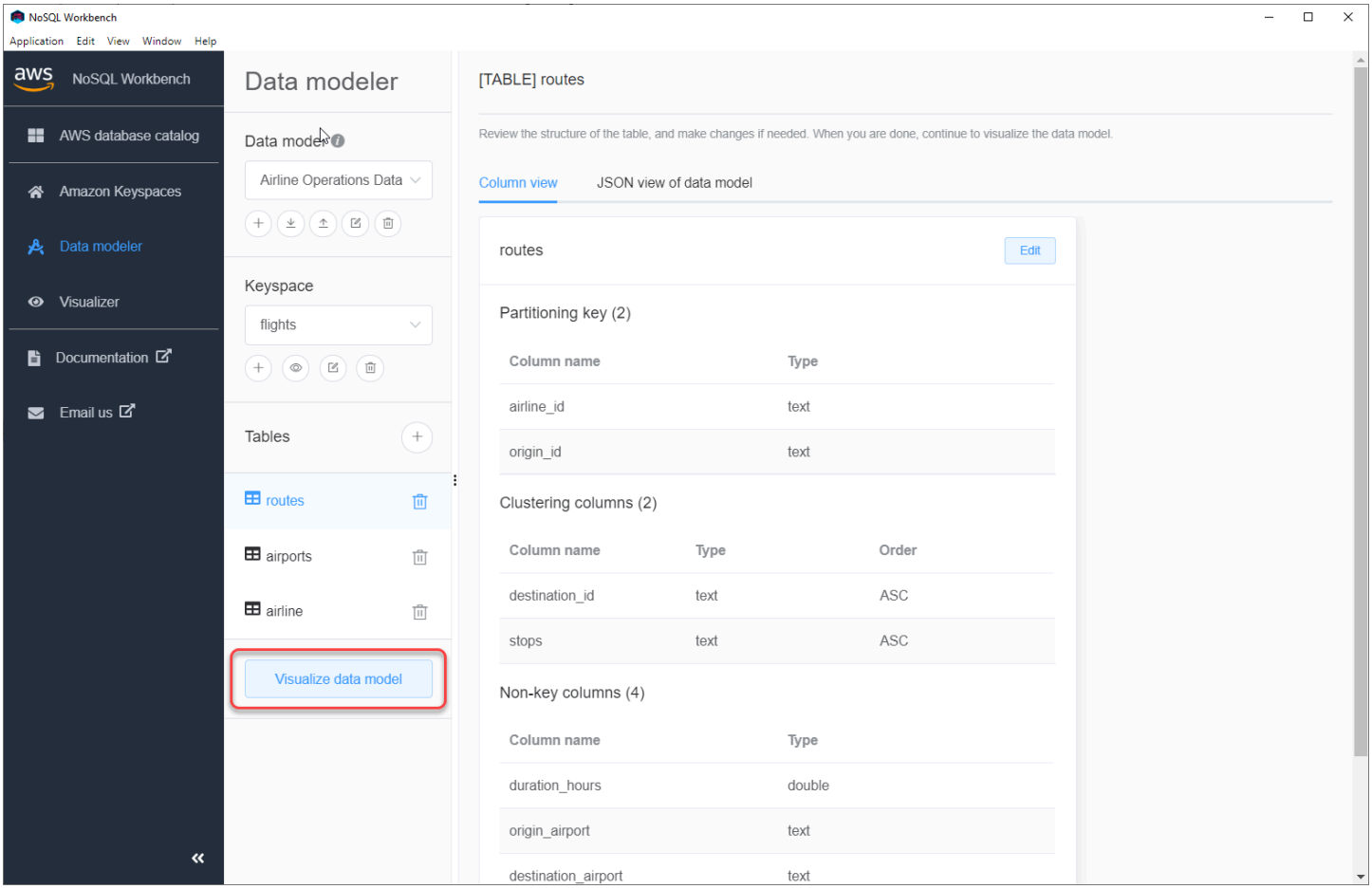
Comment visualiser les modèles de données

À l'aide de NoSQL Workbench, vous pouvez visualiser vos modèles de données pour vous assurer qu'ils peuvent prendre en charge les requêtes et les modèles d'accès de votre application. Vous pouvez également enregistrer et exporter vos modèles de données dans différents formats pour la collaboration, la documentation et les présentations.

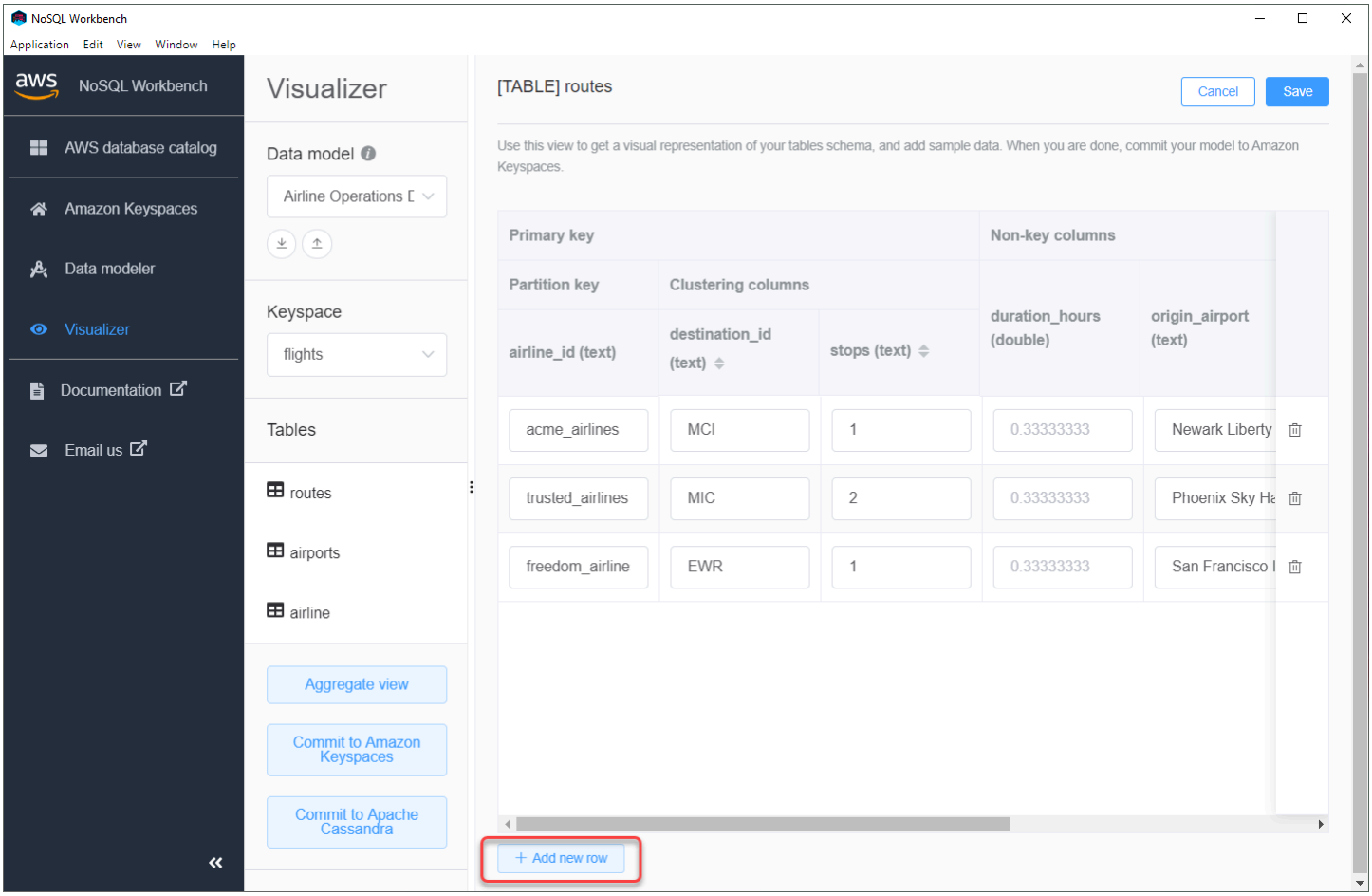
Après avoir créé un nouveau modèle de données ou modifié un modèle de données existant, vous pouvez visualiser le modèle.

Visualisation de modèles de données avec NoSQL Workbench

Lorsque vous avez terminé le modèle de données dans le modeleur de données, choisissez Visualize data model.

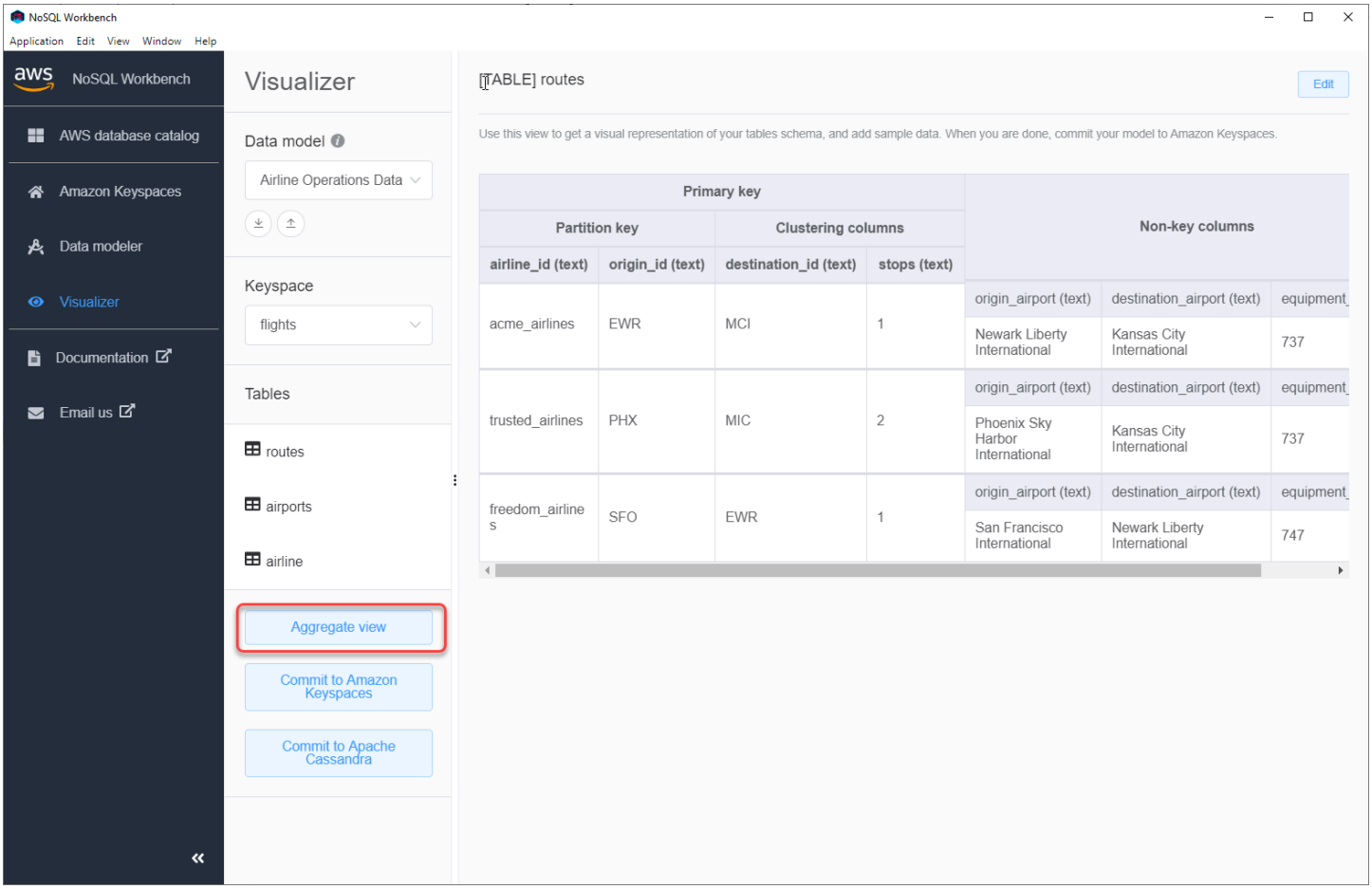


Vous accédez ainsi au visualiseur de données de NoSQL Workbench. Le visualiseur de données fournit une représentation visuelle du schéma de la table et vous permet d'ajouter des exemples de données. Pour ajouter des exemples de données à une table, choisissez une table dans le modèle, puis choisissez Modifier. Pour ajouter une nouvelle ligne de données, choisissez Ajouter une nouvelle ligne en bas de l'écran. Choisissez Save lorsque vous avez terminé.



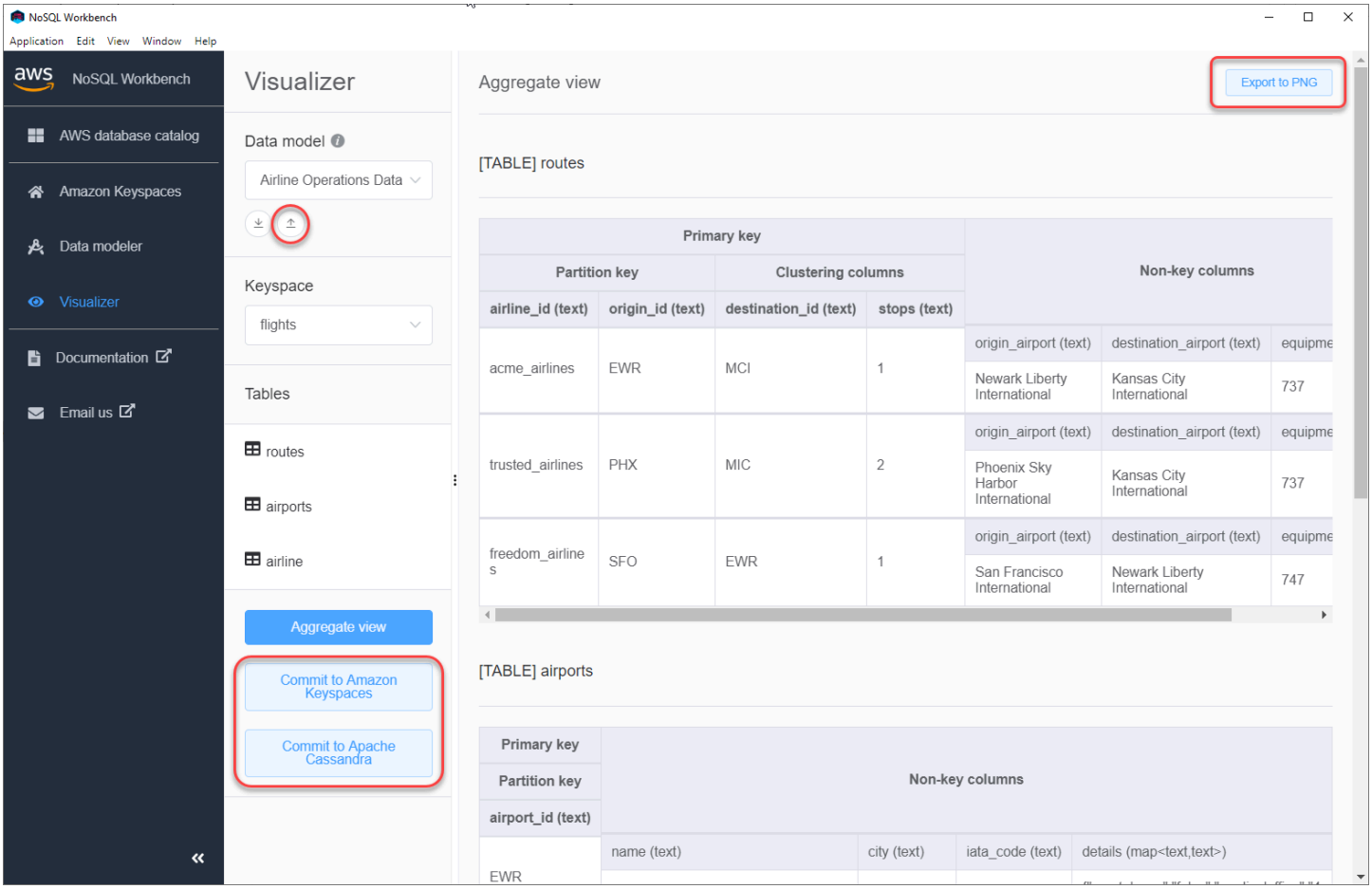
Vue d'agrégation

Après avoir confirmé le schéma de la table, vous pouvez agréger les visualisations des modèles de données.



Après avoir agrégé la vue du modèle de données, vous pouvez exporter la vue dans un fichier PNG. Pour exporter le modèle de données vers un fichier JSON, choisissez le signe de téléchargement sous le nom du modèle de données.

> **Note**
>
> Vous pouvez exporter le modèle de données au format JSON à tout moment au cours du processus de conception.



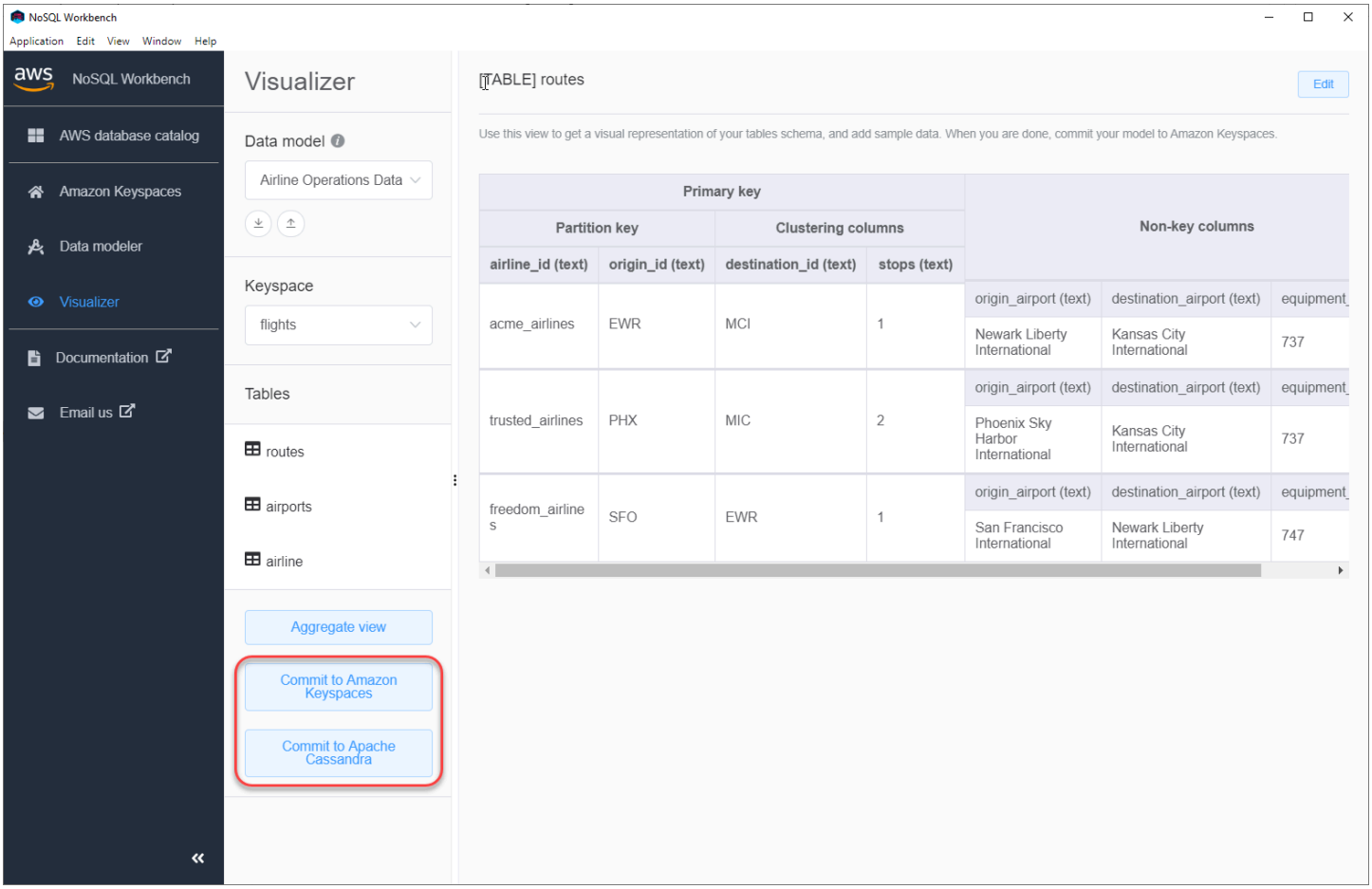
Vous disposez des options suivantes pour valider les modifications :

- Adhérez à Amazon Keyspaces
- Valider dans un cluster Apache Cassandra

Pour en savoir plus sur la façon de valider les modifications, consultezthe section called “Validation d'un modèle de données”.

Comment valider des modèles de données dans Amazon Keyspaces et Apache Cassandra

Cette section explique comment valider des modèles de données complets dans des clusters Amazon Keyspaces et Apache Cassandra. Ce processus crée automatiquement les ressources côté serveur pour les espaces de touches et les tables en fonction des paramètres que vous avez définis dans le modèle de données.



Rubriques

- Avant de commencer
- Connexion à Amazon Keyspaces à l'aide d'informations d'identification spécifiques au service
- Connexion à Amazon Keyspaces à l'aide d'informations d'AWS Identity and Access Managementidentification (IAM)
- Utilisation d'une connexion enregistrée
- S'engager dans Apache Cassandra

Avant de commencer

Amazon Keyspaces nécessite l'utilisation du protocole TLS (Transport Layer Security) pour sécuriser les connexions avec les clients. Pour vous connecter à Amazon Keyspaces à l'aide du protocole TLS, vous devez effectuer la tâche suivante avant de commencer.

- Téléchargez le certificat numérique Starfield à l'aide de la commande suivante et enregistrez-le `sf-class2-root.crt` localement ou dans votre répertoire personnel.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> Vous pouvez également utiliser le certificat numérique Amazon pour vous connecter à Amazon Keyspaces et continuer à le faire si votre client se connecte correctement à Amazon Keyspaces. Le certificat Starfield fournit une rétrocompatibilité supplémentaire pour les clients utilisant d'anciennes autorités de certification.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```


Après avoir enregistré le fichier de certificat, vous pouvez vous connecter à Amazon Keyspaces. L'une des options consiste à se connecter à l'aide d'informations d'identification spécifiques au service. Les informations d'identification spécifiques au service sont un nom d'utilisateur et un mot de passe associés à un utilisateur IAM spécifique et ne peuvent être utilisés qu'avec le service spécifié. La deuxième option consiste à se connecter avec des informations d'identification IAM utilisant le processus AWS Signature Version 4 (SigV4). Pour en savoir plus sur ces deux options, consultezthe section called “Création d'informations d'identification”.

Pour vous connecter à l'aide d'informations d'identification spécifiques au service, consultez. the section called “Connexion à l'aide d'informations d'identification spécifiques au service”

Pour vous connecter à l'aide des informations d'identification IAM, consultezthe section called “Connexion à l'aide des informations d'identification IAM”.

Connexion à Amazon Keyspaces à l'aide d'informations d'identification spécifiques au service

Cette section explique comment utiliser les informations d'identification spécifiques au service pour valider le modèle de données que vous avez créé ou modifié avec NoSQL Workbench.

1. Pour créer une nouvelle connexion à l'aide d'informations d'identification spécifiques au service, choisissez l'onglet Se connecter à l'aide d'un nom d'utilisateur et d'un mot de passe.

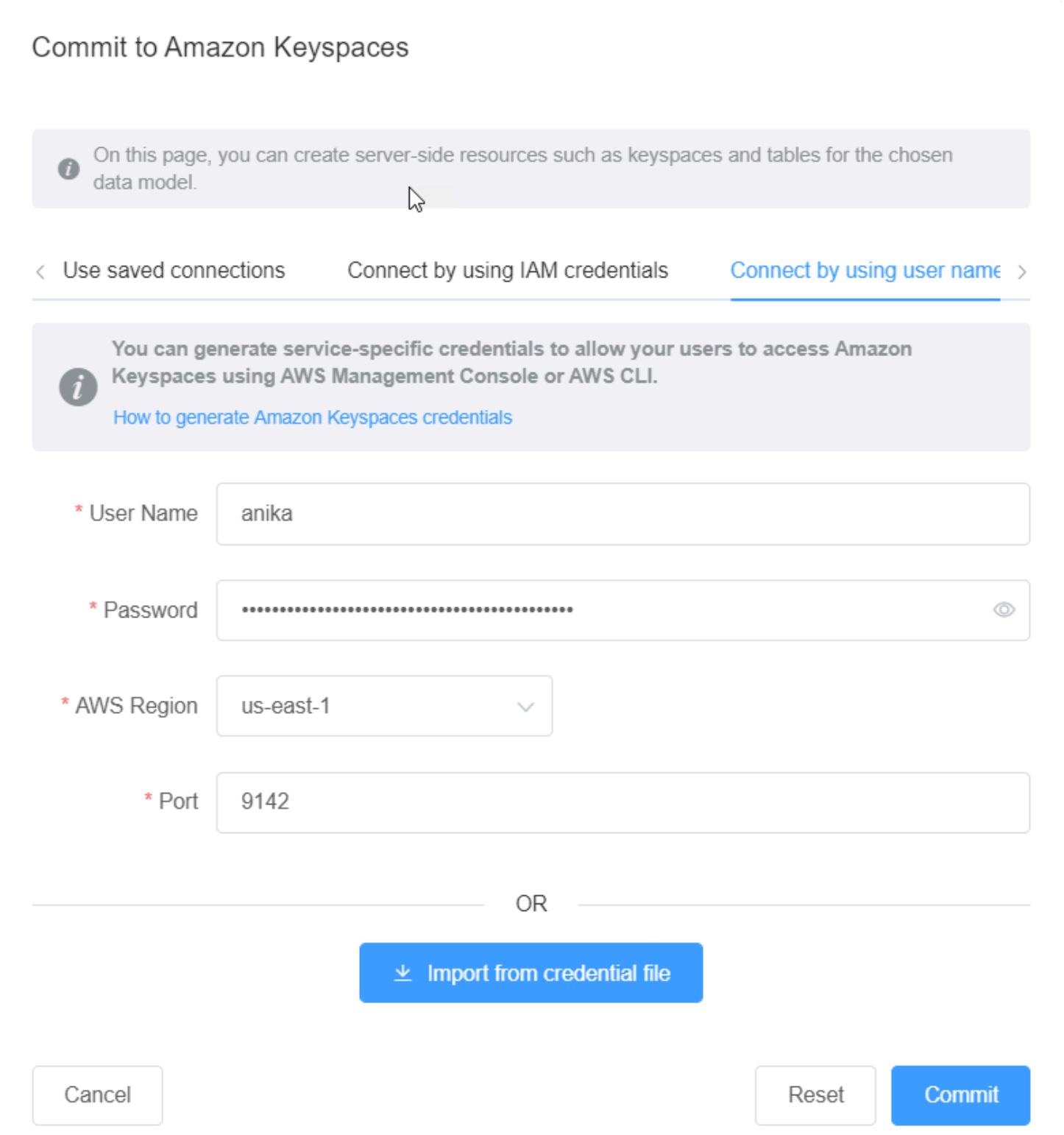
- Avant de commencer, vous devez créer des informations d'identification spécifiques au service à l'aide du processus décrit à l'adresse. the section called “Informations d'identification spécifiques au service”

Après avoir obtenu les informations d'identification spécifiques au service, vous pouvez continuer à configurer la connexion. Continuez avec l'une des méthodes suivantes :

- Nom d'utilisateur — Entrez le nom d'utilisateur.
- Mot de passe — Entrez le mot de passe.
- Région AWS— Pour les régions disponibles, voirthe section called “Points de terminaison de service”.
- Port : Amazon Keyspaces utilise le port 9142.

Vous pouvez également importer des informations d'identification enregistrées à partir d'un fichier.

2. Choisissez Commit pour mettre à jour Amazon Keyspaces avec le modèle de données.


Commit to Amazon Keyspaces
On this page, you can create server-side resources such as keyspaces and tables for the chosen data model.
Use saved connections
Connect by using IAM credentials
Connect by using user name
You can generate service-specific credentials to allow your users to access Amazon Keyspaces using AWS Management Console or AWS CLI.
How to generate Amazon Keyspaces credentials
* User Name
anika
* Password
* AWS Region
us-east-1
* Port
9142
OR
Import from credential file
Cancel
Reset
Commit

Connexion à Amazon Keyspaces à l'aide d'informations d'AWS Identity and Access Managementidentification (IAM)

Cette section explique comment utiliser les informations d'identification IAM pour valider le modèle de données créé ou modifié avec NoSQL Workbench.

1. Pour créer une nouvelle connexion à l'aide des informations d'identification IAM, choisissez l'onglet Se connecter à l'aide des informations d'identification IAM.

   - Avant de commencer, vous devez créer des informations d'identification IAM à l'aide de l'une des méthodes suivantes.

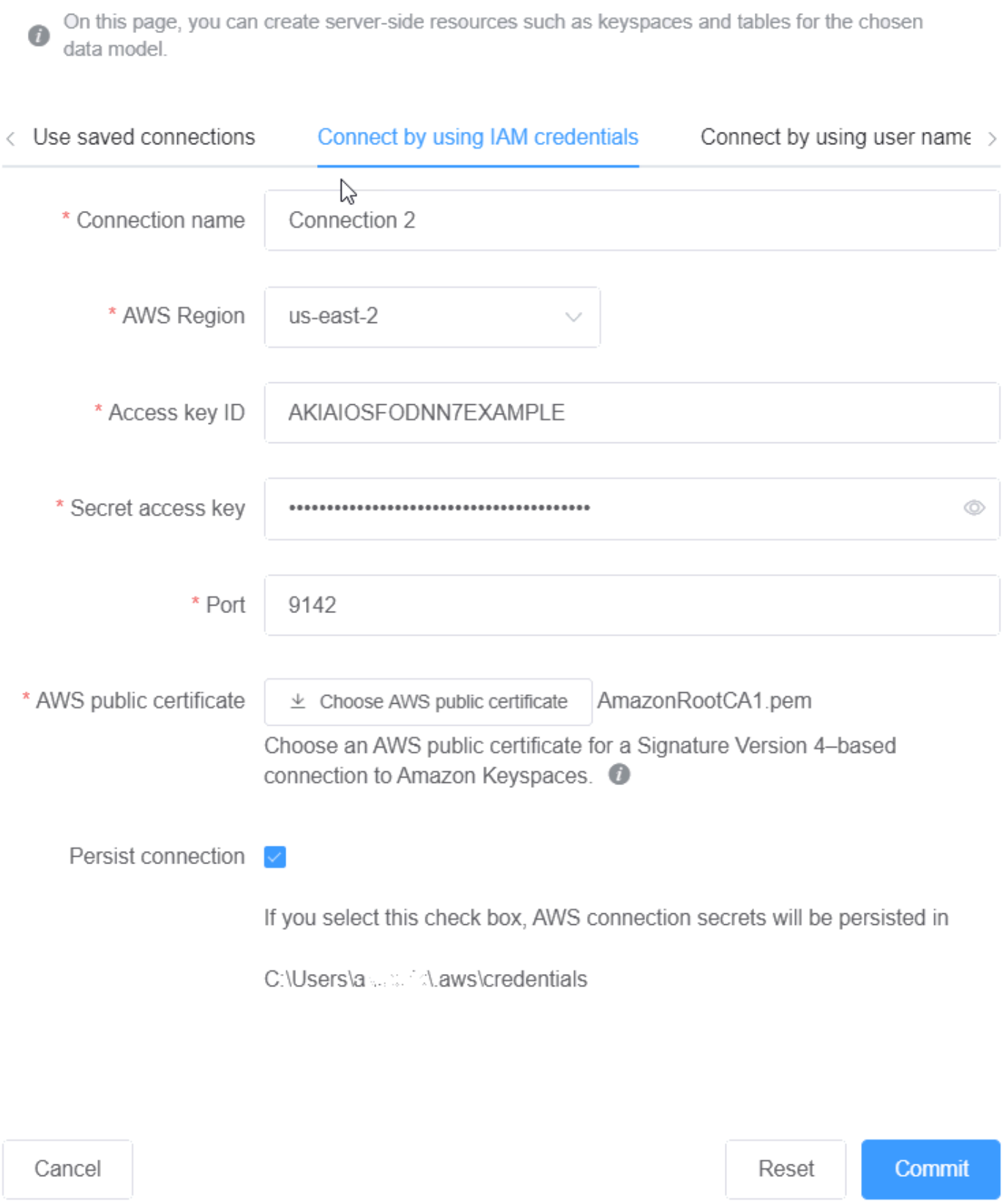
     - Pour accéder à la console, utilisez votre nom d'utilisateur et votre mot de passe IAM pour vous connecter à AWS Management Consolepartir de la page de connexion IAM. Pour plus d'informations sur les informations d'identification AWS de sécurité, y compris l'accès par programmation et les alternatives aux informations d'identification à long terme, consultez les informations d'identification AWS de sécurité dans le Guide de l'utilisateur IAM. Pour plus d'informations sur la connexion à votreCompte AWS, consultez Comment vous connecter AWS dans le Guide de Connexion à AWS l'utilisateur.
     - Pour accéder à la CLI, vous avez besoin d'un ID de clé d'accès et d'une clé d'accès secrète. Utilisation des informations d'identification temporaires au lieu des clés d'accès à long terme si possible. Les informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton de sécurité qui indique la date d'expiration des informations d'identification. Pour plus d'informations, consultez la section Utilisation d'informations d'identification temporaires avec AWS des ressources dans le Guide de l'utilisateur IAM.
     - Pour accéder à l'API, vous avez besoin d'un ID de clé d'accès et d'une clé d'accès secrète. Utilisez des clés d'accès utilisateur IAM plutôt que des clés d'accès Utilisateur racine d'un compte AWS. Pour de plus amples informations sur la création de clés d'accès, veuillez consulter Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

     Pour plus d'informations, consultez la section Gestion des clés d'accès pour les utilisateurs IAM.

Après avoir obtenu les informations d'identification IAM, vous pouvez continuer à configurer la connexion.

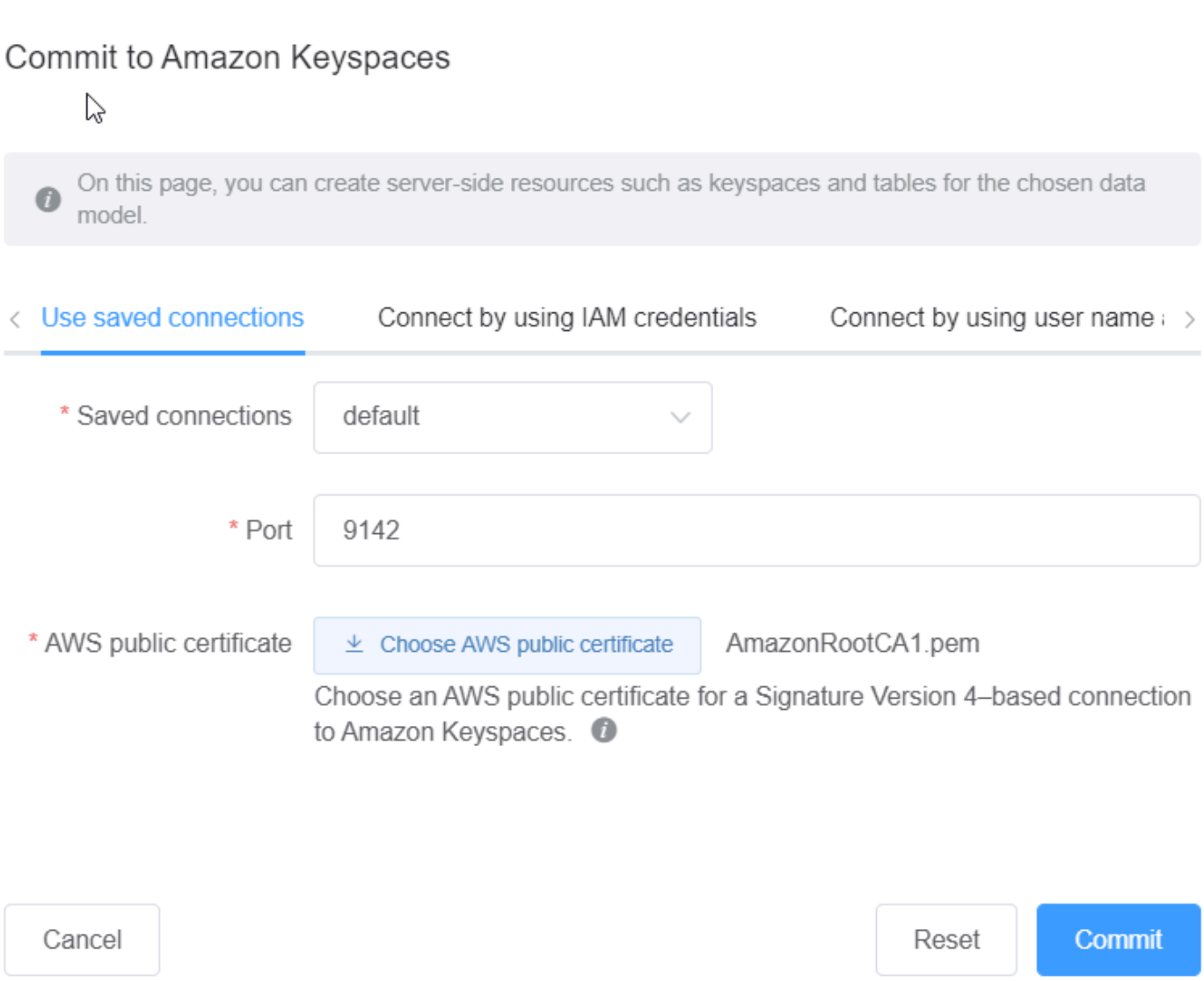
- Nom de la connexion : nom de la connexion.
- Région AWS— Pour les régions disponibles, voirthe section called “Points de terminaison de service”.
- ID de clé d'accès — Entrez l'ID de clé d'accès.
- Clé d'accès secrète — Entrez la clé d'accès secrète.
- Port : Amazon Keyspaces utilise le port 9142.
- AWScertificat public : pointez sur le AWS certificat qui a été téléchargé lors de la première étape.
- Connexion persistante : cochez cette case si vous souhaitez enregistrer les secrets de AWS connexion localement.

2. Choisissez Commit pour mettre à jour Amazon Keyspaces avec le modèle de données.


On this page, you can create server-side resources such as keyspaces and tables for the chosen data model.
Use saved connections
Connect by using IAM credentials
Connect by using user name
* Connection name
Connection 2
* AWS Region
us-east-2
* Access key ID
AKIAIOSFODNN7EXAMPLE
* Secret access key
* Port
9142
* AWS public certificate
Choose AWS public certificate
AmazonRootCA1.pem
Choose an AWS public certificate for a Signature Version 4–based connection to Amazon Keyspaces.
Persist connection
If you select this check box, AWS connection secrets will be persisted in
C:\Users\a
\.aws\credentials
Cancel
Reset
Commit

Utilisation d'une connexion enregistrée

Si vous avez déjà configuré une connexion à Amazon Keyspaces, vous pouvez l'utiliser comme connexion par défaut pour valider les modifications du modèle de données. Choisissez l'onglet Utiliser les connexions enregistrées et continuez à valider les mises à jour.

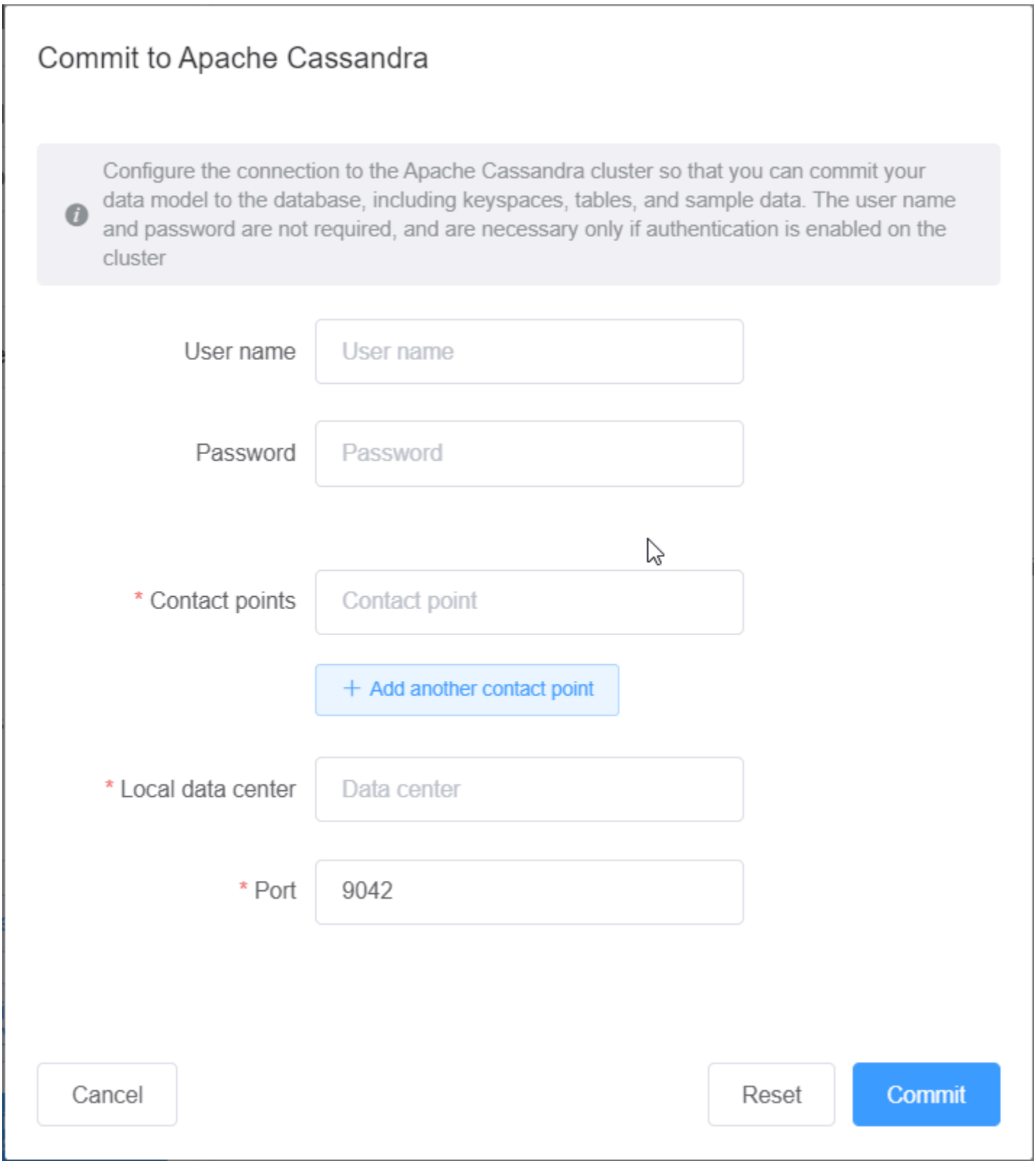


S'engager dans Apache Cassandra

Cette section explique comment établir les connexions à un cluster Apache Cassandra pour valider le modèle de données créé ou modifié avec NoSQL Workbench.

> **Note**
>
> Uniquement les modèles de données qui ont été créés avec des clusters Apache Cassandra `SimpleStrategy` ou qui `NetworkTopologyStrategy` peuvent être validés dans ces clusters. Pour modifier la stratégie de réplication, modifiez l'espace clavier dans le modeleur de données.

1.
   - Nom d'utilisateur — Entrez le nom d'utilisateur si l'authentification est activée sur le cluster.
   - Mot de passe : entrez le mot de passe si l'authentification est activée sur le cluster.
   - Points de contact — Entrez les points de contact.
   - Centre de données local — Entrez le nom du centre de données local.
   - Port — La connexion utilise le port 9042.
2. Choisissez Commit pour mettre à jour le cluster Apache Cassandra avec le modèle de données.


Commit to Apache Cassandra
Configure the connection to the Apache Cassandra cluster so that you can commit your data model to the database, including keyspaces, tables, and sample data. The user name and password are not required, and are necessary only if authentication is enabled on the cluster
User name
User name
Password
Password
* Contact points
Contact point
+ Add another contact point
* Local data center
Data center
* Port
9042
Cancel
Reset
Commit

Exemples de modèle de données dans NoSQL Workbench

La page d'accueil du modeleur et du visualiseur affiche un certain nombre d'exemples de modèles fournis avec NoSQL Workbench. Cette section décrit ces modèles et leurs utilisations potentielles.

Rubriques

Modèle de données d'employé

Ce modèle de données représente un schéma Amazon Keyspaces pour une application de base de données destinée aux employés.

Les applications qui accèdent aux informations sur les employés d'une entreprise donnée peuvent utiliser ce modèle de données.

Les modèles d'accès pris en charge par ce modèle de données sont les suivants :

- Récupération du dossier d'un employé avec un identifiant donné.
- Récupération du dossier d'un employé avec un identifiant et une division donnés.
- Récupération du dossier d'un employé avec un identifiant et un nom donnés.

Modèle de données de transaction de carte de crédit

Ce modèle de données représente un schéma Amazon Keyspaces pour les transactions par carte de crédit dans les magasins de détail.

Le stockage des transactions par carte de crédit aide non seulement les magasins à tenir leurs livres, mais aide également les directeurs de magasin à analyser les tendances d'achat, ce qui peut les aider à établir des prévisions et à planifier.

Les modèles d'accès pris en charge par ce modèle de données sont les suivants :

- Récupération des transactions par numéro de carte de crédit, mois, année et date.

- Récupération des transactions par numéro de carte de crédit, catégorie et date.
- Récupération des transactions par catégorie, lieu et numéro de carte de crédit.
- Récupération des transactions par numéro de carte de crédit et statut du litige.

Modèle de données d'exploitation d'avion

Ce modèle de données affiche des données sur les vols en avion, y compris les aéroports, les compagnies aériennes et les itinéraires de vol.

Les principaux composants de la modélisation Amazon Keyspaces qui sont présentés sont les paires clé-valeur, les magasins de données à colonnes larges, les clés composites et les types de données complexes tels que des cartes pour illustrer les modèles d'accès aux données NoSQL courants.

Les modèles d'accès pris en charge par ce modèle de données sont les suivants :

- Récupération des itinéraires en provenance d'une compagnie aérienne donnée à un aéroport donné.
- Récupération des itinéraires avec un aéroport de destination donné.
- Récupération des aéroports avec des vols directs.
- Récupération des informations relatives à l'aéroport et à la compagnie aérienne.

Historique de version pour NoSQL Workbench

Le tableau ci-après décrit les modifications importantes apportées à chaque version de l'application côté client NoSQL Workbench.

Modification	Description	Date
Prise en charge de NoSQL Workbench pour Amazon Keyspaces — GA.	NoSQL Workbench pour Amazon Keyspaces est généralement disponible.	28 octobre 2020
Version préliminaire de NoSQL Workbench.	NoSQL Workbench est une application côté client qui vous permet de concevoir et de visualiser plus facilemen	5 octobre 2020

Modification	Description	Date
	t des modèles de données non relationnels pour Amazon Keyspaces. Les clients NoSQL Workbench sont disponibles pour Windows, macOS et Linux. Pour plus d'informations, veuillez consulter NoSQL Workbench pour Amazon Keyspaces.	

Réplication multirégionale pour Amazon Keyspaces (pour Apache Cassandra)

Vous pouvez utiliser Amazon Keyspaces Multi-Region Replication pour répliquer vos données grâce à une réplication active-active, automatisée et entièrement gérée sur le site de votre choix. Régions AWS Avec la réplication active-active, chaque région est capable d'effectuer des lectures et des écritures de manière isolée. Vous pouvez améliorer à la fois la disponibilité et la résilience face à la dégradation régionale, tout en bénéficiant de lectures et d'écritures locales à faible latence pour les applications globales.

Avec la réplication multirégionale, Amazon Keyspaces réplique les données de manière asynchrone entre les régions, et les données sont généralement propagées entre les régions en une seconde. De plus, grâce à la réplication multirégionale, vous n'avez plus à résoudre les conflits et les problèmes de divergence des données. Vous pouvez donc vous concentrer sur votre application.

Par défaut, Amazon Keyspaces réplique les données dans trois zones de disponibilité au sein d'une même zone Région AWS pour garantir la durabilité et la haute disponibilité. Avec la réplication multirégionale, vous pouvez créer des espaces clés multirégionaux qui répliquent vos tables dans un maximum de six zones géographiques Régions AWS différentes de votre choix.

Rubriques

Avantages de l'utilisation de la réplication multirégionale

La réplication multirégionale offre les avantages suivants.

- Lectures et écritures globales avec une latence d'un chiffre en millisecondes — Dans Amazon Keyspaces, la réplication est active-active. Vous pouvez effectuer des opérations de lecture et d'écriture localement depuis les régions les plus proches de vos clients avec une latence d'un chiffre en millisecondes, quelle que soit l'échelle. Vous pouvez utiliser les tables multirégionales

d'Amazon Keyspaces pour les applications internationales qui ont besoin d'un temps de réponse rapide partout dans le monde.

- Continuité d'activité améliorée et protection contre la dégradation dans une seule région — Grâce à la réplication multirégionale, vous pouvez remédier à la dégradation en une seule fois Région AWS en redirigeant votre application vers une autre région de votre espace clé multirégional. Amazon Keyspaces proposant une réplication active-active, cela n'a aucun impact sur vos lectures et vos écritures.

  Amazon Keyspaces garde une trace de toutes les écritures effectuées sur votre espace de touches multirégional mais qui n'ont pas été propagées à toutes les régions répliques. Une fois que la région est de nouveau en ligne, Amazon Keyspaces synchronise automatiquement toutes les modifications manquantes afin que vous puissiez les récupérer sans aucun impact sur l'application.
- Réplication à haut débit entre les régions : la réplication multirégionale utilise une réplication physique rapide basée sur le stockage des données entre les régions, avec un délai de réplication généralement inférieur à une seconde.

  La réplication dans Amazon Keyspaces n'a que peu ou pas d'impact sur vos requêtes de base de données car elle ne partage pas les ressources de calcul avec votre application. Cela signifie que vous pouvez traiter les cas d'utilisation à haut débit d'écriture ou les cas d'utilisation présentant des pics ou des pics de débit soudains sans aucun impact sur les applications.
- Cohérence et résolution des conflits — Toute modification apportée aux données d'une région est reproduite dans les autres régions dans un espace clé multirégional. Si les applications mettent à jour les mêmes données dans différentes régions en même temps, des conflits peuvent survenir.

  Pour garantir une certaine cohérence, Amazon Keyspaces utilise des horodatages au niveau des cellules et le dernier rédacteur obtient le rapprochement entre les mises à jour simultanées. La résolution des conflits est entièrement gérée et s'effectue en arrière-plan sans aucun impact sur les applications.

Pour plus d'informations sur les configurations et fonctionnalités prises en charge, consultezthe section called “Notes d’utilisation”.

Modes de capacité et tarification

Pour un espace clé multirégional, vous pouvez utiliser le mode capacité à la demande ou le mode capacité provisionnée. Pour de plus amples informations, veuillez consulter the section called “Modes de capacité de lecture/écriture”.

Pour le mode à la demande, 1,25 unités de demande d'écriture (WRU) vous sont facturées pour écrire jusqu'à 1 Ko de données par ligne. Les écritures vous sont facturées dans chaque région de votre espace de saisie multirégional. Par exemple, l'écriture d'une ligne de 3 Ko de données dans un espace clé multirégional avec deux régions nécessite 7,5 WRU : 3 * 1,25 * 2 = 7,5 WRU. En outre, les écritures qui incluent à la fois des données statiques et non statiques nécessitent des opérations d'écriture supplémentaires.

Pour le mode provisionné, 1,25 unités de capacité d'écriture (WCU) vous sont facturées pour écrire jusqu'à 1 Ko de données par ligne. Les écritures vous sont facturées dans chaque région de votre espace de saisie multirégional. Par exemple, l'écriture d'une ligne de 3 Ko de données par seconde dans un espace clé multirégional avec deux régions nécessite 7,5 WCU : 3 * 1,25 * 2 = 7,5 WCU. En outre, les écritures qui incluent à la fois des données statiques et non statiques nécessitent des opérations d'écriture supplémentaires.

Pour plus d'informations sur les tarifs, consultez les tarifs d'Amazon Keyspaces (pour Apache Cassandra).

Comment fonctionne la réplication multirégionale dans Amazon Keyspaces

Cette section fournit un aperçu du fonctionnement de la réplication multirégionale d'Amazon Keyspaces. Pour plus d'informations sur les tarifs, consultez les tarifs d'Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Comment fonctionne la réplication multirégionale dans Amazon Keyspaces

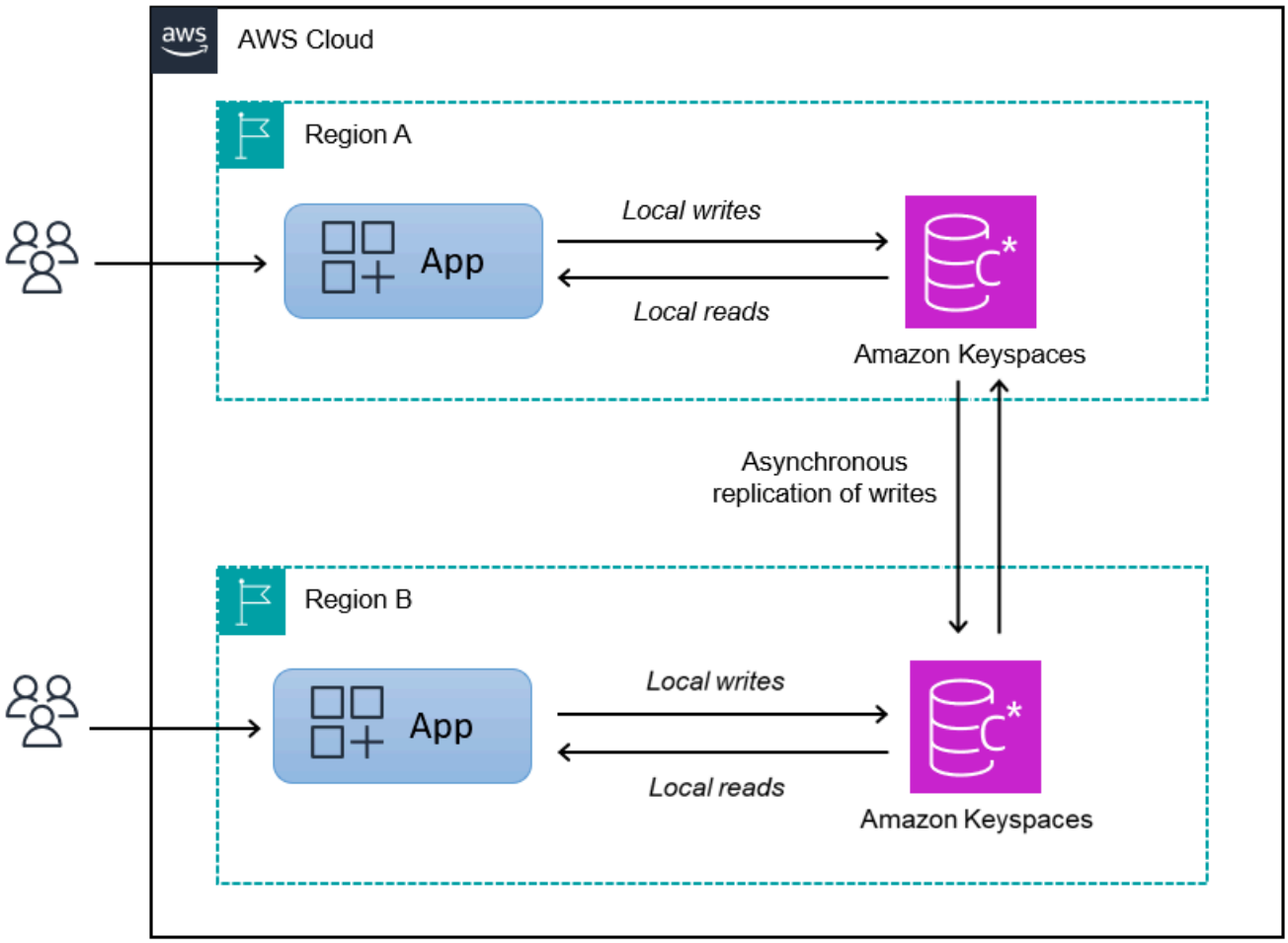
Amazon Keyspaces Multi-Region Replication met en œuvre une architecture de résilience des données qui distribue vos données de manière indépendante et géographiquement distribuée.

Régions AWS Il utilise la réplication active-active, qui fournit une faible latence locale, chaque région étant capable d'effectuer des lectures et des écritures de manière isolée.

Lorsque vous créez un espace clé multirégional Amazon Keyspaces, vous pouvez sélectionner jusqu'à cinq régions supplémentaires dans lesquelles les données seront répliquées. Chaque table que vous créez dans un espace clé multirégional se compose de plusieurs répliques de tables (une par région) qu'Amazon Keyspaces considère comme une seule unité.

Les réplicas a les mêmes nom de table et schéma de clé primaire. Lorsqu'une application écrit des données dans une table locale d'une région, les données sont écrites de manière durable en utilisant le niveau de `LOCAL_QUORUM` cohérence. Amazon Keyspaces réplique automatiquement les données de manière asynchrone vers les autres régions de réplication. Le délai de réplication entre les régions est généralement inférieur à une seconde et n'a aucun impact sur les performances ou le débit de votre application.

Une fois les données écrites, vous pouvez les lire dans la table multi-régions d'une autre région de réplication avec les niveaux de `LOCAL_ONE/LOCAL_QUORUM` cohérence. Pour plus d'informations sur les configurations et fonctionnalités prises en charge, consultezthe section called “Notes d’utilisation”.



Résolution des conflits de réplication multirégionale

La réplication multirégionale d'Amazon Keyspaces est entièrement gérée, ce qui signifie que vous n'avez pas à effectuer de tâches de réplication telles que l'exécution régulière d'opérations de réparation pour résoudre les problèmes de synchronisation des données. Amazon Keyspaces surveille la cohérence des données entre les tables de différentes Régions AWS tables en détectant et en réparant les conflits, et synchronise automatiquement les répliques.

Amazon Keyspaces utilise la méthode du dernier rédacteur gagnant pour le rapprochement des données. Grâce à ce mécanisme de résolution des conflits, toutes les régions d'un espace clé multirégional s'accordent sur la dernière mise à jour et convergent vers un état dans lequel elles disposent toutes de données identiques. Le processus de rapprochement n'a aucun impact sur les performances de l'application. Pour faciliter la résolution des conflits, les horodatages côté client sont automatiquement activés pour les tables multirégionales et ne peuvent pas être désactivés. Pour plus d'informations, consultez Horodatages côté client.

Reprise après sinistre par réplication multirégionale

Avec Amazon Keyspaces Multi-Region Replication, les écritures sont répliquées de manière asynchrone dans chaque région. Dans les rares cas de dégradation ou de défaillance d'une seule région, la réplication multirégionale vous aide à récupérer après un sinistre avec peu ou pas d'impact sur votre application. La reprise après sinistre est généralement mesurée à l'aide des valeurs de l'objectif de temps de restauration (RTO) et de l'objectif du point de restauration (RPO).

Objectif de temps de reprise : temps nécessaire à un système pour revenir en état de fonctionnement après un sinistre. Le RTO mesure le temps d'arrêt que votre charge de travail peut tolérer, mesuré dans le temps. Pour les plans de reprise après sinistre qui utilisent la réplication multirégionale pour basculer vers une région non affectée, le RTO peut être proche de zéro. Le RTO est limité par la rapidité avec laquelle votre application peut détecter la défaillance et rediriger le trafic vers une autre région.

Objectif du point de récupération : quantité de données pouvant être perdue (mesurée dans le temps). Pour les plans de reprise après sinistre qui utilisent la réplication multirégionale pour basculer vers une région non affectée, le RPO est généralement de quelques secondes à un chiffre. Le RPO est limité par la latence de réplication vers la réplique cible de basculement.

En cas de défaillance ou de dégradation régionale, vous n'avez pas besoin de promouvoir une région secondaire ou d'exécuter des procédures de basculement de base de données, car la réplication dans Amazon Keyspaces est active-active. Vous pouvez plutôt utiliser Amazon Route 53 pour acheminer votre application vers la région saine la plus proche. Pour en savoir plus sur la Route 53, consultez Qu'est-ce qu'Amazon Route 53 ? .

Si une seule Région AWS est isolée ou dégradée, votre application peut rediriger le trafic vers une autre région à l'aide de la Route 53 pour effectuer des lectures et des écritures sur une autre table de réplication. Vous pouvez également appliquer une logique métier personnalisée pour déterminer quand rediriger les demandes vers d'autres régions. Par exemple, vous pouvez informer votre application des multiples points de terminaison disponibles.

Lorsque la région est de nouveau en ligne, Amazon Keyspaces recommence à propager toutes les écritures en attente depuis cette région vers les répliques de tables des autres régions. Il reprend également la propagation des écritures des autres tables de réplique vers la région revenue en ligne.

Autorisations IAM requises pour créer des espaces clés et des tables multirégionaux

Pour créer correctement des espaces clés et des tables multirégionaux, le principal IAM doit être en mesure de créer un rôle lié à un service. Ce rôle lié à un service est un type unique de rôle IAM prédéfini par Amazon Keyspaces. Il inclut toutes les autorisations dont Amazon Keyspaces a besoin pour effectuer des actions en votre nom. Pour de plus amples informations sur le rôle lié à un service, veuillez consulter the section called “Réplication multirégionale”.

Pour créer le rôle lié au service requis par la réplication multirégionale, la politique du principal IAM nécessite les éléments suivants :

- `iam:CreateServiceLinkedRole`— L'action que le principal peut effectuer.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`— La ressource sur laquelle l'action peut être exécutée.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`— Le seul AWS service auquel ce rôle peut être associé est Amazon Keyspaces.

Voici un exemple de politique qui accorde les autorisations minimales requises à un directeur pour créer des espaces clés et des tables multirégionaux.

```
{
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication",
            "Condition": {"StringLike": {"iam:AWSServiceName":
 "replication.cassandra.amazonaws.com"}}
}
```

Pour obtenir des autorisations IAM supplémentaires pour les espaces clés et les tables multirégionaux, consultez les actions, les ressources et les clés de condition pour Amazon Keyspaces (pour Apache Cassandra) dans le Service Authorization Reference.

Réplication multirégionale et intégration avec point-in-time restauration (PITR)

La oint-in-time restauration P est prise en charge dans les tables multirégionales. Pour restaurer correctement une table multirégionale avec PITR, les conditions suivantes doivent être remplies.

- La table source et la table cible doivent être configurées en tant que tables multirégionales.
- Les régions de réplication pour l'espace clé de la table source et pour l'espace clé de la table cible doivent être identiques.

Vous pouvez exécuter l'instruction de restauration à partir de n'importe quelle région dans laquelle la table source est disponible. Amazon Keyspaces restaure automatiquement la table cible dans chaque région. Pour plus d'informations sur le PITR, consultez the section called “Comment ça marche”.

Réplication multirégionale et intégration aux services AWS

Vous pouvez surveiller les performances de réplication entre les tables de différentes Régions AWS manières en utilisant CloudWatch les métriques Amazon. La métrique suivante fournit une surveillance continue des espaces clés multirégionaux.

- `ReplicationLatency`— Cette métrique mesure le temps nécessaire à la réplication `updates` ou au passage d'une table `deletes` de réplication à une autre table de réplication dans un espace clé multirégional. `inserts`

Pour plus d'informations sur la façon de surveiller CloudWatch les métriques, consultezthe section called “Surveillance avec CloudWatch”.

Remarques d'utilisation de la réplication multirégionale Amazon Keyspaces

Tenez compte des points suivants lorsque vous utilisez la réplication multirégionale avec Amazon Keyspaces.

- Vous pouvez sélectionner jusqu'à six des publics disponibles Régions AWS. AWS GovCloud (US) Regions, les régions chinoises et celles Régions AWS qui sont désactivées par défaut ne sont pas prises en charge.

- Sélectionnez soigneusement les régions de réplication pour le keyspace, car vous ne pourrez ni les ajouter ni les supprimer ultérieurement.
- Finalisez le schéma de table avant de créer une table multirégionale, car vous ne pourrez pas ajouter de nouvelles colonnes ultérieurement.
- Pour le chiffrement au repos, utilisez une clé AWS détenue. Les clés gérées par le client ne sont pas prises en charge pour les tables multirégionales. Pour plus d'informations, veuillez consulter la rubrique

  the section called “Comment ça marche”.
- Lorsque vous utilisez la gestion des capacités allouées avec le dimensionnement automatique d'Amazon Keyspaces, veillez à utiliser les opérations de l'API Amazon Keyspaces pour créer et configurer vos tables multirégionales. Les opérations d'API Application Auto Scaling sous-jacentes qu'Amazon Keyspaces appelle en votre nom ne disposent pas de fonctionnalités multirégionales.

  Pour plus d’informations, consultez the section called “Comment utiliser la réplication multirégionale”. Pour plus d'informations sur la manière d'estimer le débit de capacité d'écriture des tables multirégionales provisionnées, consultez. the section called “Tableaux multirégionaux”
- Décidez si la table a besoin de Time to Live (TTL). Vous ne pourrez pas l'activer plus tard. Pour plus d’informations, consultez Expiration de données avec.
- Bien que les données soient automatiquement répliquées dans les régions sélectionnées d'une table multirégionale, lorsqu'un client se connecte à un point de terminaison dans une région et interroge la `system.peers` table, la requête renvoie uniquement des informations locales. Le résultat de la requête apparaît comme un cluster de centre de données unique pour le client.
- La réplication multirégionale d'Amazon Keyspaces est asynchrone et garantit la cohérence des écritures. `LOCAL_QUORUM` `LOCAL_QUORUM`la cohérence nécessite qu'une mise à jour d'une ligne soit maintenue de manière durable sur deux répliques de la région locale avant de renvoyer le succès au client. La propagation des écritures vers la région (ou les régions) répliquée est ensuite effectuée de manière asynchrone.

  La réplication multirégionale d'Amazon Keyspaces ne prend pas en charge la réplication synchrone ni la cohérence. `QUORUM`
- Lorsque vous créez un espace de touches ou un tableau multirégional, toutes les balises que vous définissez au cours du processus de création sont automatiquement appliquées à tous les espaces clés et tables de toutes les régions. Lorsque vous modifiez les balises existantes à l'aide de `ALTER KEYSPACE` ou`ALTER TABLE`, la mise à jour ne s'applique qu'au keyspace ou au tableau de la région dans laquelle vous effectuez la modification.

- Amazon CloudWatch fournit une `ReplicationLatency` métrique pour chaque région répliquée. Il calcule cette métrique en suivant les lignes qui arrivent, en comparant leur heure d'arrivée avec leur temps d'écriture initial et en calculant une moyenne. Les horaires sont enregistrés CloudWatch dans la région source. Pour plus d'informations, consultez the section called “Surveillance avec CloudWatch”.

  Il peut être utile de consulter les durées moyennes et maximales afin de déterminer le délai de réplication moyen et le plus défavorable. Il n'existe aucun SLA sur cette latence.

- Lorsque vous utilisez une table multirégionale en mode à la demande, vous pouvez observer une augmentation de la latence pour la réplication asynchrone des écritures si une réplique de table connaît un nouveau pic de trafic. Tout comme Amazon Keyspaces adapte automatiquement la capacité d'une table à la demande composée d'une seule région au trafic d'application qu'elle reçoit, Amazon Keyspaces adapte automatiquement la capacité d'une réplique de table à la demande multirégionale au trafic qu'elle reçoit. L'augmentation de la latence de réplication est transitoire car Amazon Keyspaces alloue automatiquement davantage de capacité à mesure que le volume de trafic augmente. Une fois que toutes les répliques se sont adaptées à votre volume de trafic, la latence de réplication devrait revenir à la normale. Pour plus d'informations, consultez the section called “Trafic de pointe et propriétés de scalabilité”.
- Lorsque vous utilisez une table multirégionale en mode provisionné, si votre application dépasse la capacité de débit allouée, vous pouvez observer des erreurs de capacité insuffisantes et une augmentation de la latence de réplication. Pour garantir une capacité de lecture et d'écriture suffisante pour toutes les répliques de table dans l'ensemble Régions AWS d'une table multirégionale, nous vous recommandons de configurer le dimensionnement automatique d'Amazon Keyspaces. La mise à l'échelle automatique d'Amazon Keyspaces vous aide à fournir une capacité de débit efficace pour des charges de travail variables en ajustant automatiquement la capacité de débit en fonction du trafic réel des applications. Pour plus d'informations, voir the section called “Comment fonctionne la mise à l'échelle automatique pour les tables multirégionales”.

Comment utiliser la réplication multirégionale

Vous pouvez créer et gérer des espaces clés et des tables multirégionaux à l'aide de la console Amazon Keyspaces (pour Apache Cassandra), du langage de requête Cassandra (CQL), du SDK et du AWS (). AWS Command Line Interface AWS CLI

Cette section fournit des exemples de création d'espaces clés et de tables multirégionaux avec la console, avec CQL et avec le, en utilisant à la fois le mode à la AWS CLI demande et le mode de capacité provisionnée. Toutes les tables créées dans un espace de touches multirégional héritent automatiquement des paramètres multirégionaux de l'espace de touches.

Cette section inclut également des exemples d'utilisation de la console, du CQL, et de gestion des paramètres AWS CLI de dimensionnement automatique d'Amazon Keyspaces pour les tables multirégionales provisionnées. Pour plus d'informations sur les options générales de configuration de l'autoscaling et sur leur fonctionnement, consultezthe section called “Gérez la capacité de débit grâce à la mise à l'échelle automatique”.

Notez que si vous utilisez le mode capacité provisionnée pour les tables multirégionales, vous devez toujours utiliser les appels d'API Amazon Keyspaces pour configurer le dimensionnement automatique. Cela est dû au fait que les opérations sous-jacentes de l'API Application Auto Scaling ne sont pas adaptées aux régions.

Pour plus d'informations sur la manière d'estimer le débit de capacité d'écriture des tables multirégionales provisionnées, consultez. the section called “Tableaux multirégionaux”

Pour plus d'informations sur l'API Amazon Keyspaces, consultez le manuel Amazon Keyspaces API Reference.

Pour plus d'informations sur les configurations prises en charge et les fonctionnalités de réplication multirégionale, consultezthe section called “Notes d’utilisation”.

Rubriques

- Utilisation de la console pour créer et gérer des tables multirégionales
- Utilisation de CQL pour créer et gérer des tables multirégionales
- Utilisation du AWS CLI pour créer et gérer des tables multirégionales

Utilisation de la console pour créer et gérer des tables multirégionales

Cette section fournit des exemples de création d'espaces clés et de tables multirégionaux en mode de capacité à la demande et provisionnée à l'aide de la console Amazon Keyspaces (pour Apache Cassandra). Toutes les tables que vous créez dans un espace de touches multirégional héritent automatiquement des paramètres multirégionaux de l'espace de touches.

Pour des exemples de CQL, consultezthe section called “Utilisation de CQL”. Pour AWS CLI des exemples, voirthe section called “En utilisant le AWS CLI”.

Rubriques

Création d'un espace de touches multirégional (console)

Suivez ces étapes pour créer un nouvel espace de touches multirégional à l'aide de la console Amazon Keyspaces.

Pour créer un espace de touches multirégional (console)

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Keyspaces, puis Créer un keyspace.
3. Dans le champ Nom de l'espace clé, entrez le nom de l'espace clé.
4. Dans la section Réplication multirégionale, vous pouvez ajouter jusqu'à cinq régions supplémentaires disponibles dans la liste.
5. Pour terminer, choisissez Create keyspace.

> **Note**
>
> Lorsque vous créez un espace de saisie multirégional, Amazon Keyspaces crée un rôle lié à un service dont le nom figure dans votre compte. `AWSServiceRoleForAmazonKeyspacesReplication` Ce rôle permet à Amazon Keyspaces de répliquer les écritures sur toutes les répliques d'une table multirégionale en votre nom. Pour en savoir plus, veuillez consulter la section the section called “Réplication multirégionale”.

Création d'une table multirégionale avec des paramètres par défaut (console)

Suivez ces étapes pour créer une table multirégionale à l'aide de la console Amazon Keyspaces.

Pour créer une table multirégionale (console)

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez un espace de touches multirégional.
3. Dans l'onglet Tables, choisissez Créer une table.
4. Dans Nom de la table, entrez le nom de la table. Le Régions AWS fichier dans lequel cette table est répliquée est indiqué dans la boîte d'information.
5. Continuez avec le schéma de table.
6. Sous Paramètres du tableau, passez à l'option Paramètres par défaut. Notez les paramètres par défaut suivants pour les tables multirégionales.

   - Mode capacité : le mode de capacité par défaut est à la demande. Pour plus d'informations sur la configuration du mode provisionné, consultezthe section called “Création d'une table multirégionale en mode provisionné avec le dimensionnement automatique activé (console)”.
   - Gestion des clés de chiffrement : seule l'Clé détenue par AWSoption est prise en charge.
   - Horodatages côté client : cette fonctionnalité est requise pour les tables multirégionales.

   - Choisissez Personnaliser les paramètres si vous devez activer Time to Live (TTL) pour le tableau et toutes ses répliques.

     > **Note**
     >
     > Vous ne pourrez pas modifier les paramètres TTL sur une table multirégionale existante.

7. Pour terminer, choisissez Créer une table.

Création d'une table multirégionale en mode provisionné avec le dimensionnement automatique activé (console)

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour créer une nouvelle table multirégionale avec la mise à l'échelle automatique activée

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez un espace de touches multirégional.
3. Dans l'onglet Tables, choisissez Créer une table.
4. Sur la page Créer une table, dans la section Détails de la table, sélectionnez un espace de touche et attribuez un nom à la nouvelle table.
5. Dans la section Colonnes, créez le schéma de votre table.
6. Dans la section Clé primaire, définissez la clé primaire de la table et sélectionnez les colonnes de clustering facultatives.
7. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
8. Continuez jusqu'à Paramètres de capacité en lecture/écriture.
9. Pour le Mode de capacité, choisissez Provisioned (Alloué).
10. Dans la section Capacité de lecture confirmez que l'option Mettre à l'échelle automatiquement est sélectionnée.

    Vous pouvez choisir de configurer les mêmes unités de capacité de lecture pour tous les éléments dans Régions AWS lesquels la table est répliquée. Vous pouvez également décocher la case et configurer différemment la capacité de lecture pour chaque région.

    Si vous choisissez de configurer chaque région différemment, vous sélectionnez les unités de capacité de lecture minimale et maximale pour chaque réplique de table, ainsi que l'utilisation cible.

- Unités de capacité minimale : entrez la valeur du niveau de débit minimal que le tableau doit toujours être prêt à prendre en charge. La valeur doit être comprise entre 1 et le quota par seconde du débit maximal pour votre compte (40 000 par défaut).
- Unités de capacité maximale : entrez le débit maximal que vous souhaitez allouer pour la table. La valeur doit être comprise entre 1 et le quota par seconde du débit maximal pour votre compte (40 000 par défaut).
- Utilisation cible — Entrez un taux d'utilisation cible compris entre 20 % et 90 %. Lorsque le trafic dépasse le taux d'utilisation cible défini, la capacité est automatiquement mise à l'échelle. Lorsque le trafic tombe en dessous de la cible définie, il est automatiquement réduit de nouveau.

- Désactivez la case à cocher Echelle automatique si vous souhaitez configurer manuellement la capacité de lecture du tableau. Ce paramètre s'applique à toutes les répliques de la table.

> **Note**
>
> Pour garantir une capacité de lecture suffisante pour toutes les répliques, nous recommandons le dimensionnement automatique d'Amazon Keyspaces pour les tables multirégionales provisionnées.

> **Note**
>
> Pour en savoir plus sur les quotas par défaut de votre compte et sur la façon de les augmenter, reportez-vous à la section Quotas.

11. Dans la section Capacité d'écriture, vérifiez que l'option Échelle automatique est sélectionnée. Configurez ensuite les unités de capacité de la table. Les unités de capacité d'écriture restent synchronisées entre toutes les régions Régions AWS afin de garantir une capacité suffisante pour répliquer les événements d'écriture dans les régions.
    - Désélectionnez Scale automatiquement si vous souhaitez provisionner manuellement la capacité d'écriture de la table. Ce paramètre s'applique à toutes les répliques de la table.

> **Note**
>
> Pour garantir une capacité d'écriture suffisante pour toutes les répliques, nous recommandons le dimensionnement automatique d'Amazon Keyspaces pour les tables multirégionales provisionnées.

12. Choisissez Créer un tableau. Votre table est créée avec les paramètres de mise à l'échelle automatique spécifiés.

Activation de la mise à l'échelle automatique pour une table multirégionale existante (console)

Suivez ces étapes pour activer le dimensionnement automatique pour une table multirégionale en mode provisionné avec la console Amazon Keyspaces.

> **Note**
>
> Le dimensionnement automatique d'Amazon Keyspaces nécessite la présence d'un rôle lié à un service (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) qui exécute des actions de dimensionnement automatique en votre nom. Ce rôle est créé automatiquement pour vous. Pour de plus amples informations, veuillez consulter the section called “Utilisation des rôles liés à un service”.

Pour activer le dimensionnement automatique d'Amazon Keyspaces pour une table multirégionale existante

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la table avec laquelle vous souhaitez travailler, puis accédez à l'onglet Capacité.
3. Dans la section Paramètres de capacité, choisissez Modifier.
4. En mode Capacité, assurez-vous que la table utilise le mode Capacité provisionnée.
5. Sélectionnez Mettre à l'échelle automatiquement et reportez-vous à l'étape 9 Création d'une table multirégionale en mode provisionné avec le dimensionnement automatique activé (console) pour modifier la capacité de lecture et d'écriture.

6. Lorsque les paramètres de mise à l'échelle automatique sont définis, choisissez Enregistrer.

Désactivation de la mise à l'échelle automatique pour une table multirégionale (console)

Suivez ces étapes pour désactiver le dimensionnement automatique pour une table multirégionale en mode provisionné avec la console Amazon Keyspaces.

Pour désactiver le dimensionnement automatique d'Amazon Keyspaces pour une table multirégionale existante

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la table avec laquelle vous souhaitez travailler, puis cliquez sur l'onglet Capacité.
3. Dans la section Paramètres de capacité, choisissez Modifier.
4. Pour désactiver le dimensionnement automatique d'Amazon Keyspaces, décochez la case Scale automatically. La désactivation du dimensionnement automatique annule l'enregistrement de la table en tant que cible évolutive avec Application Auto Scaling. Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling pour accéder à votre table Amazon Keyspaces, suivez les étapes décrites dans. the section called “Supprimer un rôle lié à un service pour Amazon Keyspaces”

   > **Note**
   >
   > Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

5. Lorsque les paramètres de mise à l'échelle automatique sont définis, choisissez Enregistrer.

Afficher les activités de dimensionnement automatique d'Amazon Keyspaces sur la console

Vous pouvez surveiller la manière dont le dimensionnement automatique d'Amazon Keyspaces utilise les ressources en utilisant Amazon CloudWatch, qui génère des statistiques relatives à votre utilisation et à vos performances. Suivez les étapes du guide de l'Application Auto Scaling utilisateur pour créer un CloudWatch tableau de bord.

Utilisation de CQL pour créer et gérer des tables multirégionales

Vous pouvez utiliser Cassandra Query Language (CQL) pour créer et gérer des espaces clés et des tables multirégionaux dans Amazon Keyspaces.

Cette section fournit des exemples de création et de gestion de tables multirégionales avec CQL. Toutes les tables que vous créez dans un espace de touches multirégional héritent automatiquement des paramètres multirégionaux de l'espace de touches. Pour plus d'informations sur le CQL, consultez le manuel de référence du langage CQL d'Amazon Keyspaces.

Pour plus d'informations sur les configurations et fonctionnalités prises en charge, consultezthe section called “Notes d’utilisation”.

Rubriques

- Création d'un keyspace multirégional (CQL)
- Création d'une table multirégionale avec des paramètres par défaut (CQL)
- Création d'une table multirégionale avec mode capacité provisionnée et mise à l'échelle automatique (CQL)
- Mise à jour de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CQL)
- Affichage de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CQL)
- Désactivation de la mise à l'échelle automatique pour une table multirégionale (CQL)
- Configuration manuelle de la capacité provisionnée d'une table multirégionale (CQL)

Création d'un keyspace multirégional (CQL)

Pour créer un espace de touches multirégional, utilisez cette option `NetworkTopologyStrategy` pour spécifier l'espace de touches dans Régions AWS lequel l'espace de touches doit être répliqué. Vous devez inclure votre région actuelle et au moins une région supplémentaire. L'instruction CQL suivante en est un exemple.

```
CREATE KEYSPACE mykeyspace
WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3' };
```

Toutes les tables du keyspace utilisent la même stratégie de réplication que le keyspace. Vous ne pouvez pas modifier la stratégie de réplication au niveau de la table.

`NetworkTopologyStrategy`— Le facteur de réplication pour chaque région est de trois, car Amazon Keyspaces réplique les données dans trois zones de disponibilité au sein d'une même région Région AWS, par défaut.

> **Note**
>
> Lorsque vous créez un espace de saisie multirégional, Amazon Keyspaces crée un rôle lié à un service dont le nom figure dans votre compte. `AWSServiceRoleForAmazonKeyspacesReplication` Ce rôle permet à Amazon Keyspaces de répliquer les écritures sur toutes les répliques d'une table multirégionale en votre nom. Pour en savoir plus, veuillez consulter la section the section called “Réplication multirégionale”.

Vous pouvez utiliser une instruction CQL pour interroger la `tables` table dans l'`system_multiregion_info`espace de touches afin de répertorier par programmation les régions et le statut de la table multirégionale que vous spécifiez. Le code suivant en est un exemple.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

Le résultat de l'instruction ressemble à ce qui suit :

```
 keyspace_name  | table_name     | region         | status
----------------+----------------+----------------+--------
 mykeyspace     | mytable        | us-east-1      | ACTIVE
 mykeyspace     | mytable        | ap-southeast-1 | ACTIVE
 mykeyspace     | mytable        | eu-west-1      | ACTIVE
```

Création d'une table multirégionale avec des paramètres par défaut (CQL)

Pour créer une table multirégionale avec des paramètres par défaut, vous pouvez utiliser l'exemple suivant.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
```

```
    WITH CUSTOM_PROPERTIES = {
  'capacity_mode':{
   'throughput_mode':'PAY_PER_REQUEST'
  },
  'point_in_time_recovery':{
   'status':'enabled'
  },
  'encryption_specification':{
   'encryption_type':'AWS_OWNED_KMS_KEY'
  },
  'client_side_timestamps':{
   'status':'enabled'
  }
};
```

Création d'une table multirégionale avec mode capacité provisionnée et mise à l'échelle automatique (CQL)

Pour créer une table multirégionale en mode provisionné avec mise à l'échelle automatique, vous devez d'abord spécifier le mode de capacité en le définissant `CUSTOM_PROPERTIES` pour la table. Après avoir spécifié le mode de capacité provisionnée, vous pouvez configurer les paramètres de dimensionnement automatique de la table à l'aide `AUTOSCALING_SETTINGS` de.

Pour des informations détaillées sur les paramètres de mise à l'échelle automatique, la politique de suivi des cibles, la valeur cible et les paramètres facultatifs, consultezthe section called “Création d'une nouvelle table avec mise à l'échelle automatique à l'aide de CQL”.

Lorsque vous créez une table multirégionale, vous pouvez également spécifier des paramètres de capacité de lecture et de mise à l'échelle automatique de lecture différents pour chaque réplique de la table. Les paramètres que vous spécifiez remplacent les paramètres généraux de la table pour les paramètres spécifiés Région AWS. La capacité d'écriture reste toutefois synchronisée entre toutes les répliques afin de garantir une capacité suffisante pour répliquer les écritures dans toutes les régions.

Pour définir la capacité de lecture d'une réplique de table dans une région spécifique, vous pouvez configurer les paramètres suivants dans le cadre de la table `replica_updates` :

- La région
- Les unités de capacité de lecture provisionnées (facultatif)
- Paramètres de mise à l'échelle automatique pour la capacité de lecture (facultatif)

L'exemple suivant montre une `CREATE TABLE` instruction pour une table multirégionale en mode provisionné. Les paramètres généraux de mise à l'échelle automatique des capacités d'écriture et de lecture sont les mêmes. Toutefois, les paramètres de mise à l'échelle automatique de lecture spécifient des périodes de recharge supplémentaires de 60 secondes avant d'augmenter ou de diminuer la capacité de lecture du tableau. En outre, les paramètres de mise à l'échelle automatique de la capacité de lecture pour la région USA Est (Virginie du Nord) sont supérieurs à ceux des autres répliques. De plus, la valeur cible est fixée à 70 % au lieu de 50 %.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 5,
        'write_capacity_units': 5
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    },
    'replica_updates': {
        'us-east-1': {
            'provisioned_read_capacity_autoscaling_update': {
                'maximum_units': 20,
                'minimum_units': 5,
                'scaling_policy': {
```

```
                    'target_tracking_scaling_policy_configuration': {
                        'target_value': 70
                    }
                }
            }
        }
    }
};
```

Mise à jour de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CQL)

Vous pouvez l'utiliser `ALTER TABLE` pour mettre à jour le mode de capacité et les paramètres de mise à l'échelle automatique d'une table existante. Si vous mettez à jour une table actuellement en mode capacité à la demande, `capacity_mode` c'est obligatoire. Si votre table est déjà en mode capacité provisionnée, ce champ peut être omis.

Pour des informations détaillées sur les paramètres de mise à l'échelle automatique, la politique de suivi des cibles, la valeur cible et les paramètres facultatifs, consultezthe section called “Création d'une nouvelle table avec mise à l'échelle automatique à l'aide de CQL”.

Dans la même déclaration, vous pouvez également mettre à jour les paramètres de capacité de lecture et de mise à l'échelle automatique des répliques de tables dans des régions spécifiques en mettant à jour les `replica_updates` propriétés de la table. La déclaration suivante en est un exemple.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
```

```
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    },
    'replica_updates': {
        'us-east-1': {
            'provisioned_read_capacity_autoscaling_update': {
                'maximum_units': 20,
                'minimum_units': 5,
                'scaling_policy': {
                    'target_tracking_scaling_policy_configuration': {
                        'target_value': 70
                    }
                }
            }
        }
    }
};
```

Affichage de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CQL)

Pour afficher la configuration de mise à l'échelle automatique d'une table multirégionale, utilisez la commande suivante.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name = 'mykeyspace'
 AND table_name = 'mytable';
```

Le résultat de cette commande ressemble à ce qui suit :

```
 keyspace_name  | table_name | region         |
 provisioned_read_capacity_autoscaling_update

                                     | provisioned_write_capacity_autoscaling_update
```

```
---------------+------------+---------------
+-------------------------------------------------------------------------------------------------
+-------------------------------------------------------------------------------------------------
  mykeyspace    |  mytable   | ap-southeast-1 | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 60, 'disable_scale_in': false, 'target_value':
 50, 'scale_in_cooldown': 60}}} | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 0, 'disable_scale_in': false, 'target_value': 50,
 'scale_in_cooldown': 0}}}
  mykeyspace    |  mytable   | us-east-1      | {'minimum_units': 5, 'maximum_units':
 20, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 60, 'disable_scale_in': false, 'target_value':
 70, 'scale_in_cooldown': 60}}} | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 0, 'disable_scale_in': false, 'target_value': 50,
 'scale_in_cooldown': 0}}}
  mykeyspace    |  mytable   | eu-west-1      | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 60, 'disable_scale_in': false, 'target_value':
 50, 'scale_in_cooldown': 60}}} | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 0, 'disable_scale_in': false, 'target_value': 50,
 'scale_in_cooldown': 0}}}
```

Désactivation de la mise à l'échelle automatique pour une table multirégionale (CQL)

Vous pouvez l'utiliser `ALTER TABLE` pour désactiver la mise à l'échelle automatique pour une table existante. Notez que vous ne pouvez pas désactiver le redimensionnement automatique pour une réplique de table individuelle.

Dans l'exemple suivant, la mise à l'échelle automatique est désactivée pour la capacité de lecture de la table.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_read_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

Configuration manuelle de la capacité provisionnée d'une table multirégionale (CQL)

Si vous devez désactiver le dimensionnement automatique pour une table multirégionale, vous pouvez l'utiliser pour `ALTER TABLE` provisionner manuellement la capacité de lecture de la table pour une table de réplique.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    },
    'replica_updates': {
        'us-east-1': {
            'read_capacity_units': 2
         }
    }
};
```

> **Note**
>
> Nous recommandons d'utiliser le dimensionnement automatique pour les tables multirégionales qui utilisent de la capacité allouée. Pour de plus amples informations, veuillez consulter the section called “Tableaux multirégionaux”.

Utilisation du AWS CLI pour créer et gérer des tables multirégionales

Vous pouvez utiliser le AWS Command Line Interface (AWS CLI) pour créer et gérer des espaces de touches et des tables multirégionaux dans Amazon Keyspaces.

Cette section fournit des exemples de création et de gestion de tables multirégionales à l'aide du AWS CLI. Toutes les tables que vous créez dans un espace de touches multirégional héritent automatiquement des paramètres multirégionaux de l'espace de touches.

Pour plus d'informations sur les AWS CLI commandes Amazon Keyspaces décrites dans cette rubrique, consultez le AWS CLI Command Reference for Amazon Keyspaces.

Rubriques

Création d'un nouvel espace de touches multirégional (CLI)

Pour créer un espace de touches multirégional, vous pouvez utiliser l'instruction CLI suivante. Spécifiez votre région actuelle et au moins une région supplémentaire dans le`regionList`.

```
aws keyspaces create-keyspace --keyspace-name mykeyspace
             \ --replication-specification
 replicationStrategy=MULTI_REGION,regionList=us-east-1,eu-west-1
```

> **Note**
>
> Lorsque vous créez un espace de saisie multirégional, Amazon Keyspaces crée un rôle lié à un service dont le nom figure dans votre compte. `AWSServiceRoleForAmazonKeyspacesReplication` Ce rôle permet à Amazon Keyspaces de répliquer les écritures sur toutes les répliques d'une table multirégionale en votre nom. Pour en savoir plus, veuillez consulter la section the section called “Réplication multirégionale”.

Création d'une nouvelle table multirégionale avec des paramètres par défaut (CLI)

Pour créer une table multirégionale avec les paramètres par défaut, il suffit de spécifier le schéma. Vous pouvez utiliser l'exemple suivant.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
            \ --schema-definition 'allColumns=[{name=pk,type=int}],partitionKeys={name=
 pk}'
```

Le résultat de la commande est le suivant :

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable"
}
```

Pour confirmer les paramètres du tableau, vous pouvez utiliser l'instruction suivante.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

La sortie affiche tous les paramètres par défaut d'une table multirégionale.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable",
    "creationTimestamp": "2023-12-19T16:50:37.639000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "pk",
                "type": "int"
            }
        ],
        "partitionKeys": [
            {
                "name": "pk"
            }
        ],
        "clusteringKeys": [],
```

```
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2023-12-19T16:50:37.639000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
    },
    "replicaSpecifications": [
        {
            "region": "us-east-1",
            "status": "ACTIVE",
            "capacitySpecification": {
                "throughputMode": "PAY_PER_REQUEST",
                "lastUpdateToPayPerRequestTimestamp": 1702895811.469
            }
        },
        {
            "region": "eu-north-1",
            "status": "ACTIVE",
            "capacitySpecification": {
                "throughputMode": "PAY_PER_REQUEST",
                "lastUpdateToPayPerRequestTimestamp": 1702895811.121
            }
        }
    ]
}
```

Création d'une nouvelle table multirégionale en mode provisionné avec mise à l'échelle automatique (CLI)

Pour créer une table multirégionale en mode provisionné avec une configuration de dimensionnement automatique, vous pouvez utiliser le. AWS CLI Notez que vous devez utiliser la `create-table` commande Amazon Keyspaces CLI pour configurer les paramètres de mise à l'échelle automatique multirégionale. Cela est dû au fait qu'Application Auto Scaling, le service qu'Amazon Keyspaces utilise pour effectuer le dimensionnement automatique en votre nom, ne prend pas en charge plusieurs régions.

Pour plus d'informations sur les paramètres de dimensionnement automatique, la politique de suivi des cibles, la valeur cible et les paramètres facultatifs, consultezthe section called “Créez un nouveau tableau avec mise à l'échelle automatique à l'aide du AWS CLI”.

Lorsque vous créez une nouvelle table multirégionale en mode provisionné avec des paramètres de dimensionnement automatique, vous pouvez spécifier les paramètres généraux de la table qui sont valides pour tous les Régions AWS éléments dans lesquels la table est répliquée. Vous pouvez ensuite remplacer les paramètres de capacité de lecture et lire les paramètres de mise à l'échelle automatique pour chaque réplique. La capacité d'écriture reste toutefois synchronisée entre toutes les répliques afin de garantir une capacité suffisante pour répliquer les écritures dans toutes les régions.

Pour définir la capacité de lecture d'une réplique de table dans une région spécifique, vous pouvez configurer les paramètres suivants dans le cadre de la table `replicaSpecifications` :

- La région
- Les unités de capacité de lecture provisionnées (facultatif)
- Paramètres de mise à l'échelle automatique pour la capacité de lecture (facultatif)

Lorsque vous créez des tables multirégionales provisionnées avec des paramètres de dimensionnement automatique complexes et différentes configurations pour les répliques de tables, il est utile de charger les paramètres de mise à l'échelle automatique et les configurations de réplique de la table à partir de fichiers JSON.

Pour utiliser l'exemple de code suivant, vous pouvez télécharger les exemples de fichiers JSON depuis auto-scaling.zip, puis extraire `auto-scaling.json` et`replication.json`. Prenez note du chemin d'accès aux fichiers.

Dans cet exemple, les fichiers JSON se trouvent dans le répertoire actuel. Pour connaître les différentes options de chemin de fichier, consultez Comment charger des paramètres à partir d'un fichier.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
           \ --schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=int}],partitionKeys=[{name=pk},{name=ck}]'
           \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
           \ --auto-scaling-specification file://auto-scaling.json
           \ --replica-specifications file://replication.json
```

Mise à jour de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CLI)

Pour mettre à jour le mode provisionné et la configuration de dimensionnement automatique d'une table existante, vous pouvez utiliser la AWS CLI `update-table` commande.

Notez que vous devez utiliser les commandes de la CLI Amazon Keyspaces pour créer ou modifier les paramètres de mise à l'échelle automatique multirégionale. Cela est dû au fait qu'Application Auto Scaling, le service qu'Amazon Keyspaces utilise pour effectuer le dimensionnement automatique de la capacité des tables en votre nom, ne prend pas en charge les tables multiples. Régions AWS

Lorsque vous mettez à jour le mode provisionné ou les paramètres de dimensionnement automatique d'une table multirégionale, vous pouvez mettre à jour les paramètres de capacité de lecture et la configuration de dimensionnement automatique de lecture pour chaque réplique de la table.

La capacité d'écriture reste toutefois synchronisée entre toutes les répliques afin de garantir une capacité suffisante pour répliquer les écritures dans toutes les régions. Pour mettre à jour la capacité de lecture d'une réplique de table dans une région spécifique, vous pouvez modifier l'un des paramètres facultatifs suivants de la table `replicaSpecifications` :

- Les unités de capacité de lecture provisionnées (facultatif)
- Paramètres de mise à l'échelle automatique pour la capacité de lecture (facultatif)

Lorsque vous mettez à jour des tables multirégionales avec des paramètres de mise à l'échelle automatique complexes et différentes configurations pour les répliques de tables, il est utile de charger les paramètres de mise à l'échelle automatique et les configurations de réplique de la table à partir de fichiers JSON.

Pour utiliser l'exemple de code suivant, vous pouvez télécharger les exemples de fichiers JSON depuis auto-scaling.zip, puis extraire `auto-scaling.json` et`replication.json`. Prenez note du chemin d'accès aux fichiers.

Dans cet exemple, les fichiers JSON se trouvent dans le répertoire actuel. Pour connaître les différentes options de chemin de fichier, consultez Comment charger des paramètres à partir d'un fichier.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
          \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
          \ --auto-scaling-specification file://auto-scaling.json
          \ --replica-specifications file://replication.json
```

Affichage de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CLI)

Pour afficher la configuration de mise à l'échelle automatique d'une table multirégionale, vous pouvez utiliser l'`get-table-auto-scaling-settings`opération. La commande CLI suivante en est un exemple.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

Le résultat suivant doit s'afficher.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:777788889999:/keyspace/mykeyspace/
table/mytable",
    "autoScalingSpecification": {
        "writeCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 0,
                    "scaleOutCooldown": 0,
```

```
                    "targetValue": 50.0
                }
            }
        },
        "readCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 20,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 60,
                    "scaleOutCooldown": 60,
                    "targetValue": 70.0
                }
            }
        }
    },
    "replicaSpecifications": [
        {
            "region": "us-east-1",
            "autoScalingSpecification": {
                "writeCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 0,
                            "scaleOutCooldown": 0,
                            "targetValue": 50.0
                        }
                    }
                },
                "readCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 20,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 60,
                            "scaleOutCooldown": 60,
```

```
                            "targetValue": 70.0
                        }
                    }
                }
            }
        },
        {
            "region": "eu-north-1",
            "autoScalingSpecification": {
                "writeCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 0,
                            "scaleOutCooldown": 0,
                            "targetValue": 50.0
                        }
                    }
                },
                "readCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 60,
                            "scaleOutCooldown": 60,
                            "targetValue": 50.0
                        }
                    }
                }
            }
        }
    ]
}
```

Désactiver le dimensionnement automatique pour une table multirégionale (CLI)

Vous pouvez utiliser la AWS CLI `update-table` commande pour désactiver le dimensionnement automatique pour une table existante. Notez que vous ne pouvez pas désactiver le redimensionnement automatique pour une réplique de table individuelle.

Dans l'exemple suivant, la mise à l'échelle automatique est désactivée pour la capacité de lecture de la table.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
          \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Pour supprimer le rôle lié à un service utilisé par Application Auto Scaling, vous devez désactiver le dimensionnement automatique sur toutes les tables du compte. Régions AWS

Configuration manuelle de la capacité provisionnée d'une table multirégionale (CLI)

Si vous devez désactiver le dimensionnement automatique pour une table multirégionale, vous pouvez l'utiliser pour `update-table` provisionner manuellement la capacité de lecture de la table pour une table de réplique.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
          \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
          \ --replica-specifications region="us-east-1",readCapacityUnits=5
```

> **Note**
>
> Nous recommandons d'utiliser le dimensionnement automatique pour les tables multirégionales qui utilisent de la capacité allouée. Pour de plus amples informations, veuillez consulter the section called “Tableaux multirégionaux”.

oint-in-time Restauration de fichiers dans le passé pour Amazon Keyspaces (pour Apache Cassandra)

Point-in-time recovery (PITR) permet de protéger vos tables Amazon Keyspaces contre les opérations d'écriture ou de suppression accidentelles en vous fournissant des sauvegardes continues des données de vos tables.

Supposons, par exemple, qu'un script de test soit écrit accidentellement dans une table Amazon Keyspaces de production. point-in-time La restauration vous permet de restaurer les données de cette table à n'importe quel instant dans le passé au cours des 35 derniers jours. Si vous supprimez une table alors que la point-in-time restauration est activée, vous pouvez rechercher les données de la table supprimée pendant 35 jours (sans frais supplémentaires) et la restaurer dans l'état dans lequel elle se trouvait juste avant le point de suppression.

Vous pouvez restaurer une table Amazon Keyspaces à un moment donné en utilisant la console, leAWS SDK et leAWS Command Line Interface (AWS CLI), ou le langage CQL (Cassandra Query Language). Pour plus d'informations, veuillez consulter Restauration d'une table Amazon Keyspaces à un instant dans le passé.

Lesoint-in-time opérations P n'ont aucun impact sur les performances ou la disponibilité de la table de base, et la restauration d'une table ne consomme pas de débit supplémentaire.

Pour obtenir des informations sur les quotas PIR, veuillez consulterQuotas.

Pour obtenir des informations sur la tarification, veuillez consulter la tarification d'Amazon Keyspaces (pour Apache Cassandra).

Rubriques

- Comment fonctionne point-in-time la restauration dans Amazon Keyspaces
- Restauration d'une table Amazon Keyspaces à un instant dans le passé

Comment fonctionne point-in-time la restauration dans Amazon Keyspaces

Cette section fournit un aperçu du fonctionnement de Amazon Keyspaces point-in-time Recovery (PITR). Pour plus d'informations sur les tarifs, consultez les tarifs d'Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Activation point-in-time de la restauration (PITR)

Vous pouvez activer le PITR à l'aide de la console, ou vous pouvez l'activer par programmation.

Activation du PITR avec la console

Les paramètres PITR pour les nouvelles tables peuvent être gérés sous l'option Paramètres personnalisés. Par défaut, PITR est activé sur les nouvelles tables créées via la console.

Pour activer le PITR pour une table existante, procédez comme suit.

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le volet de navigation, choisissez Tables et sélectionnez la table que vous souhaitez modifier.
3. Dans l'onglet Sauvegardes, choisissez Modifier.
4. Dans la section Modifier les paramètres de point-in-time restauration, sélectionnez Activer oint-in-time la restauration P.

Vous pouvez désactiver le PITR sur une table à tout moment en procédant comme suit.

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le volet de navigation, choisissez Tables et sélectionnez la table que vous souhaitez modifier.

3. Dans l'onglet Sauvegardes, choisissez Modifier.
4. Dans la section Modifier les paramètres de point-in-time restauration, décochez la case Activer la oint-in-time restauration P.

> **Important**
>
> La désactivation du PITR supprime immédiatement l'historique de vos sauvegardes, même si vous réactivez le PITR sur la table dans les 35 jours.

Pour savoir comment restaurer une table à l'aide de la console, consultezthe section called "Restauration d'une table à un instant dans le passé (console)".

Activation du PITR à l'aide du AWS CLI

Vous pouvez gérer les paramètres PITR des tables à l'aide de l'`UpdateTable`API.

Lorsque vous créez une nouvelle table à l'aide deAWS CLI, vous devez activer explicitement le PITR lors de la création de la nouvelle table.

Pour activer le PITR lorsque vous créez une nouvelle table, vous pouvez utiliser la AWS CLI commande suivante à titre d'exemple. La commande a été divisée en lignes distinctes pour améliorer la lisibilité.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --point-in-time-recovery 'status=ENABLED'
```

> **Note**
>
> Si aucune valeur point-in-time de restauration n'est spécifiée, point-in-time la restauration est désactivée par défaut.

Pour confirmer le paramètre de point-in-time restauration d'une table, vous pouvez utiliser la AWS CLI commande suivante.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Pour activer le PITR pour une table existante à l'aide deAWS CLI, exécutez la commande suivante.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=ENABLED'
```

Pour désactiver le PITR sur une table existante, exécutez la AWS CLI commande suivante.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=DISABLED'
```

> **Important**
>
> La désactivation du PITR supprime immédiatement l'historique de vos sauvegardes, même si vous réactivez le PITR sur la table dans les 35 jours.

Activation de PITR à l'aide de CQL

Vous pouvez gérer les paramètres PITR des tables à l'aide de la propriété `point_in_time_recovery` personnalisée.

Lorsque vous créez une nouvelle table à l'aide de CQL, vous devez activer explicitement PITR lors de la création de la nouvelle table.

Pour activer le PITR lorsque vous créez une nouvelle table, vous pouvez utiliser la commande CQL suivante à titre d'exemple.

```
CREATE TABLE "my_keyspace1"."my_table1"(
 "id" int,
 "name" ascii,
 "date" timestamp,
 PRIMARY KEY("id"))
WITH CUSTOM_PROPERTIES = {
 'capacity_mode':{'throughput_mode':'PAY_PER_REQUEST'},
 'point_in_time_recovery':{'status':'enabled'}
}
```

> **Note**
>
> Si aucune propriété personnalisée de point-in-time restauration n'est spécifiée, point-in-time la restauration est désactivée par défaut.

Pour activer le PITR pour une table existante à l'aide de CQL, exécutez la commande CQL suivante.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'enabled'}}
```

Pour désactiver le PITR sur une table existante, exécutez la commande CQL suivante.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

> **Important**
>
> La désactivation du PITR supprime immédiatement l'historique de vos sauvegardes, même si vous réactivez le PITR sur la table dans les 35 jours.

Pour plus d'informations dans le manuel de référence du langage CQL, reportez-vous aux sections the section called “CREATE TABLE” etthe section called “ALTER TABLE”. Pour savoir comment restaurer une table à l'aide de CQL, consultezthe section called “Restauration d'une table à un instant dans le passé avec CQL”.

Autorisations requises pour restaurer une table

Pour restaurer correctement une table, l'utilisateur ou le rôle IAM doit disposer des autorisations minimales suivantes :

- `cassandra:Restore`— L'action de restauration est requise pour que la table cible soit restaurée.
- `cassandra:Select`— L'action de sélection est requise pour lire dans la table source.
- `cassandra:TagResource`— L'action de balise est facultative et n'est requise que si l'opération de restauration ajoute des balises.

Voici un exemple de politique qui accorde les autorisations minimales requises à un utilisateur pour restaurer des tables dans Keyspace`mykeyspace`.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Restore",
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/*",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Des autorisations supplémentaires pour restaurer une table peuvent être requises en fonction d'autres fonctionnalités sélectionnées. Par exemple, si la table source est chiffrée au repos à l'aide d'une clé gérée par le client, Amazon Keyspaces doit être autorisé à accéder à la clé gérée par le client de la table source pour pouvoir restaurer correctement la table. Pour plus d'informations, consultez the section called “PITR et tables cryptées”.

Si vous utilisez des politiques IAM avec des clés de condition pour restreindre le trafic entrant vers des sources spécifiques, vous devez vous assurer qu'Amazon Keyspaces est autorisé à effectuer une opération de restauration pour le compte de votre principal. Vous devez ajouter une clé de `aws:ViaAWSService` condition à votre politique IAM si celle-ci limite le trafic entrant à l'une des conditions suivantes :

- Points de terminaison VPC avec `aws:SourceVpce`
- plages d'adresses IP avec `aws:SourceIp`
- VPC avec `aws:SourceVpc`

La clé de `aws:ViaAWSService` condition permet l'accès lorsqu'un AWS service fait une demande en utilisant les informations d'identification du principal. Pour plus d'informations, voir Éléments de politique IAM JSON : clé de condition dans le guide de l'utilisateur IAM.

Voici un exemple de politique qui limite le trafic source à une adresse IP spécifique et permet à Amazon Keyspaces de restaurer une table pour le compte du principal.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"CassandraAccessForCustomIp",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"false"
            },
            "ForAnyValue:IpAddress":{
               "aws:SourceIp":[
                  "123.45.167.89"
               ]
            }
         }
      },
      {
         "Sid":"CassandraAccessForAwsService",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"true"
            }
         }
      }
   ]
}
```

Pour un exemple de politique utilisant la clé de condition `aws:ViaAWSService` globale, voirthe section called “Politiques relatives aux terminaux VPC et restauration d'Amazon Keyspaces point-in-time (PITR)”.

Fenêtre temporelle pour les sauvegardes continues PITR

Amazon Keyspaces PITR utilise deux horodatages pour maintenir la période pendant laquelle les sauvegardes restaurables sont disponibles pour une table.

- Heure de restauration la plus ancienne : indique l'heure de la première sauvegarde restaurable. La première sauvegarde restaurable remonte à 35 jours ou à la date d'activation du PITR, selon la date la plus récente. La période de sauvegarde maximale de 35 jours ne peut pas être modifiée.
- Heure actuelle — L'horodatage de la dernière sauvegarde restaurable est l'heure actuelle. Si aucun horodatage n'est fourni lors d'une restauration, l'heure actuelle est utilisée.

Lorsque le PITR est activé, vous pouvez effectuer une restauration à n'importe quel moment entre `EarliestRestorableDateTime` et`CurrentTime`. Vous ne pouvez restaurer les données d'une table qu'à une date où le PITR était activé.

Si vous désactivez le PITR puis le réactivez ultérieurement, vous redéfinissez l'heure de début de la première sauvegarde disponible à la date de réactivation du PITR. Cela signifie que la désactivation du PITR efface l'historique de vos sauvegardes.

> **Note**
>
> Les opérations du langage de définition des données (DDL) sur les tables, telles que les modifications de schéma, sont effectuées de manière asynchrone. Vous ne pouvez voir que les opérations terminées dans les données de votre table restaurée, mais vous pouvez voir des actions supplémentaires sur votre table source si elles étaient en cours au moment de la restauration. Pour obtenir la liste des instructions DDL, consultezthe section called “Instructions DDL”.

Il n'est pas nécessaire qu'une table soit active pour être restaurée. Vous pouvez également restaurer les tables supprimées si le PITR a été activé sur la table supprimée et si la suppression a eu lieu dans la fenêtre de sauvegarde (ou au cours des 35 derniers jours).

> **Note**
>
> Si une nouvelle table est créée avec le même nom qualifié (par exemple, mykeyspace.mytable) qu'une table précédemment supprimée, la table supprimée ne pourra plus être restaurée. Si vous essayez de le faire depuis la console, un avertissement s'affiche.

Paramètres de restauration PITR

Lorsque vous restaurez une table à l'aide de PITR, Amazon Keyspaces restaure le schéma et les données de votre table source dans l'état basé sur l'horodatage `day:hour:minute:second` () sélectionné pour une nouvelle table. PITR ne remplace pas les tables existantes.

Outre le schéma et les données de la table, PITR les restaure `custom_properties` à partir de la table source. Contrairement aux données de la table, qui sont restaurées en fonction de l'horodatage sélectionné entre l'heure de restauration la plus ancienne et l'heure actuelle, les propriétés personnalisées sont toujours restaurées en fonction des paramètres de la table à l'heure actuelle.

Les paramètres de la table restaurée correspondent aux paramètres de la table source avec l'horodatage du lancement de la restauration. Si vous souhaitez remplacer ces paramètres lors de la restauration, vous pouvez le faire à l'aide `WITH custom_properties` de. Les propriétés personnalisées incluent les paramètres suivants.

- Mode de capacité en lecture/écriture
- Paramètres de capacité de débit provisionnée
- Réglages PITR

Si la table est en mode capacité allouée avec le dimensionnement automatique activé, l'opération de restauration rétablit également les paramètres de dimensionnement automatique de la table. Vous pouvez les remplacer à l'aide du `autoscaling_settings` paramètre dans CQL ou à l'`autoScalingSpecification`aide de la CLI. Pour plus d'informations sur les paramètres de mise à l'échelle automatique, consultezthe section called “Gérez la capacité de débit grâce à la mise à l'échelle automatique”.

Lorsque vous effectuez une restauration complète de la table, tous les paramètres de la table restaurée proviennent des paramètres actuels de la table source au moment de la restauration.

Supposons par exemple que le débit alloué d'une table vienne d'être abaissé à 50 unités de capacité de lecture et 50 unités de capacité d'écriture. Vous rétablissez ensuite l'état de la table à il y a trois semaines. À cette époque, son débit provisionné était fixé à 100 unités de capacité de lecture et à 100 unités de capacité d'écriture. Dans ce cas, Amazon Keyspaces restaure les données de votre table à ce moment-là, mais utilise les paramètres de débit actuels (50 unités de capacité de lecture et 50 unités de capacité d'écriture).

Les paramètres suivants ne sont pas restaurés et vous devez les configurer manuellement pour la nouvelle table.

- AWS Identity and Access Management politiques (IAM)
- CloudWatch Mesures et alarmes Amazon
- Tags (peuvent être ajoutés à l'`RESTORE`instruction CQL en utilisant`WITH TAGS`)

Restauration PITR de tables chiffrées

Lorsque vous restaurez une table à l'aide du PITR, Amazon Keyspaces restaure les paramètres de chiffrement de votre table source. Si la table a été chiffrée avec un Clé détenue par AWS (par défaut), elle est automatiquement restaurée avec le même paramètre. Si la table que vous souhaitez restaurer a été chiffrée à l'aide d'une clé gérée par le client, cette même clé gérée par le client doit être accessible à Amazon Keyspaces pour restaurer les données de la table.

Vous pouvez modifier les paramètres de chiffrement de la table au moment de la restauration. Pour passer d'une clé gérée par le client Clé détenue par AWS à une clé gérée par le client, vous devez fournir une clé gérée par le client valide et accessible au moment de la restauration.

Si vous souhaitez passer d'une clé gérée par le client à uneClé détenue par AWS, vérifiez qu'Amazon Keyspaces a accès à la clé gérée par le client de la table source pour restaurer la table avec un. Clé détenue par AWS Pour plus d'informations sur les paramètres de chiffrement au repos des tables, consultezthe section called “Comment ça marche”.

> **Note**
>
> Si le tableau a été supprimé parce qu'Amazon Keyspaces a perdu l'accès à votre clé gérée par le client, vous devez vous assurer que la clé gérée par le client est accessible à Amazon Keyspaces avant d'essayer de restaurer le tableau. Une table chiffrée à l'aide d'une clé gérée par le client ne peut pas être restaurée si Amazon Keyspaces n'a pas accès à cette clé. Pour

plus d'informations, consultez la section Résolution des problèmes d'accès aux clés dans le Guide du AWS Key Management Service développeur.

Restauration PITR de tables multirégionales

Vous pouvez restaurer une table multirégionale à l'aide de PITR. Pour que l'opération de restauration soit réussie, la table source et la table de destination doivent être répliquées sur la même Régions AWS table.

Amazon Keyspaces rétablit les paramètres de la table source dans chacune des régions répliquées faisant partie de l'espace de touches. Vous pouvez également modifier les paramètres lors de l'opération de restauration. Pour plus d'informations sur les paramètres qui peuvent être modifiés au cours de la restauration, consultezthe section called “Restaurer les paramètres”.

Pour plus d'informations sur la réplication multirégionale, consultezthe section called “Comment ça marche”.

Durée de restauration des tables avec PITR

Le temps nécessaire pour restaurer une table dépend de plusieurs facteurs et n'est pas toujours directement corrélé à la taille de la table.

Voici quelques considérations relatives aux délais de restauration.

- Vous restaurez vos sauvegardes vers une nouvelle table. Même si la table est vide, la réalisation de toutes les actions nécessaires à la création de la nouvelle table et au lancement du processus de restauration peut prendre jusqu'à 20 minutes.
- Les temps de restauration des grandes tables dotées de modèles de données bien distribués peuvent être de plusieurs heures, voire plus.
- Si votre table source contient des données très asymétriques, le délai de restauration peut augmenter. Par exemple, si la clé primaire de votre table utilise le mois de l'année comme clé de partition et que toutes vos données datent du mois de décembre, vous avez des données asymétriques.

Une bonne pratique de planification de la reprise après sinistre consiste à documenter régulièrement les temps moyens de restauration et à déterminer comment ces délais affectent votre objectif global de temps de récupération.

Amazon Keyspaces PITR et intégration aux services AWS

Les opérations PITR suivantes sont enregistrées AWS CloudTrail pour permettre une surveillance et un audit continus.

- Créez une nouvelle table avec PITR activé ou désactivé.
- Activez ou désactivez le PITR sur une table existante.
- Restaurez une table active ou supprimée.

Pour plus d'informations, consultez Journalisation des appels d'API Amazon Keyspaces avec AWS CloudTrail.

Vous pouvez effectuer les actions PITR suivantes à l'aide AWS CloudFormation de.

- Créez une nouvelle table avec PITR activé ou désactivé.
- Activez ou désactivez le PITR sur une table existante.

Pour de plus amples informations, veuillez consulter le document Référence du type de ressource Cassandra dans le Guide de l'utilisateur AWS CloudFormation.

Restauration d'une table Amazon Keyspaces à un instant dans le passé

Amazon Keyspaces (pour Apache Cassandra) point-in-time recovery (PITR) vous permet de restaurer les données des tables Amazon Keyspaces à tout moment au cours des 35 derniers jours. La première partie de ce didacticiel explique comment restaurer une table à un moment donné à l'aide de la console Amazon Keyspaces, duAWS Command Line Interface (AWS CLI) et du langage de requête Cassandra (CQL). La deuxième partie explique comment restaurer une table supprimée à l'aide duAWS CLI et CQL.

Rubriques

- Restaurer une table supprimée à l'aide deAWS CLI
- Restaurer une table supprimée avec CQL

Avant de commencer

Si vous ne l'avez pas déjà fait, vous devez configurer les autorisations appropriées pour que l'utilisateur restaure les tables Amazon Keyspaces. DansAWS Identity and Access Management (IAM), la politiqueAWS géréeAmazonKeyspacesFullAccess inclut les autorisations permettant de restaurer les tables Amazon Keyspaces. Pour connaître les étapes détaillées de mise en œuvre d'une politique avec les autorisations minimales requises, consultezthe section called “Restaurer les autorisations”.

Restauration d'une table à un instant dans le passé (console)

L'exemple suivant montre comment utiliser la console Amazon Keyspaces pour restaurer une table existante nomméemytable à un instant dans le passé.

> **Note**
>
> Cette procédure part du principe que vous avez activé point-in-time la restauration. Pour activer le PITR pour lemytable tableau, suivez les étapes décrites dansthe section called “Utilisation de la console”.

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le volet de navigation sur le côté gauche de la console, choisissez Tables.
3. Dans la liste des tables, choisissez la table mytable.
4. Dans l'onglet Sauvegardes dumytable tableau, dans la section oint-in-time Restauration de PC, choisissez Restaurer.
5. Pour le nom de la nouvelle table, tapez **mytable_restored**.
6. Pour définir le moment de l'opération de restauration, vous pouvez choisir entre deux options :
   - Sélectionnez l'heure la plus ancienne préconfigurée.
   - Sélectionnez Spécifier la date et l'heure de restauration souhaitée de la nouvelle table.

> **Note**
>
> Vous pouvez procéder à une restauration à n'importe quel instant dans le passé entre l'heure la plus ancienne et l'heure actuelle. Amazon Keyspaces restaure vos données de table dans un état dépendant de la date et de l'heure sélectionnées (day:hour:minute:second).

7. Choisissez Restaurer pour démarrer le processus de restauration.

   La table en cours de restauration s'affiche avec le statut Restauration en cours. Une fois le processus de restauration terminé, l'état de la table `mytable_restored` devient Active.

> **Important**
>
> Pendant la restauration, ne modifiez ou ne supprimez pas les politiques AWS Identity and Access Management (IAM) qui accordent à l'entité (par exemple, utilisateur, groupe ou rôle) l'autorisation d'effectuer une restauration. Sinon, il peut en résulter un comportement inattendu. Supposons, par exemple, que vous ayez supprimé les autorisations d'écriture pour une table alors que cette table était en cours de restauration. Dans ce cas, l'opération `RestoreTableToPointInTime` sous-jacente ne peut pas écrire les données restaurées dans la table.
> Vous pouvez modifier ou supprimer des autorisations uniquement lorsque l'opération de restauration est terminée.

Restauration d'une table à un instant dans le passé à l'aide deAWS CLI

La procédure suivante montre comment utiliser l'AWS CLI pour restaurer une table existante nommée `myTable` à un instant dans le passé.

1. Dans un premier temps, vous créez une table simple nommée sur`myTable` laquelle PITR est activé. La commande a été divisée en lignes distinctes pour des raisons de lisibilité.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --point-in-time-recovery 'status=ENABLED'
```

2. Confirmez les propriétés de la nouvelle table et passez en revue le`earliestRestorableTimestamp` pour PITR.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

La sortie de cette commande renvoie le résultat suivant.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/myKeyspace/
table/myTable",
    "creationTimestamp": "2022-06-20T14:34:57.049000-07:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2022-06-20T14:34:57.049000-07:00"
    },
    "encryptionSpecification": {
```

```
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "ENABLED",
        "earliestRestorableTimestamp": "2022-06-20T14:35:13.693000-07:00"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Vous pouvez restaurer une table active à n'importe quelle heure point-in-time comprise entre l'heure actuelle`earliestRestorableTimestamp` et l'heure actuelle à des intervalles d'une seconde. L'heure actuelle est l'heure par défaut.

3. Pour restaurer une table à un instant dans le passé, spécifiez un`restore_timestamp` au format ISO 8601. Vous pouvez choisir n'importe quel instant dans le passé au cours des 35 derniers jours à un instant dans le passé au cours des 35 derniers jours. Par exemple, la commande suivante restaure la table à la `EarliestRestorableDateTime`.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored' --restore-timestamp "2022-06-20 21:35:14.693"
```

La sortie de cette commande renvoie l'ARN de la table restaurée.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored"
}
```

Pour rétablir l'heure actuelle du tableau, vous pouvez omettre le`restore-timestamp`.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored1'"
```

> **Important**
>
> Pendant la restauration, ne modifiez ou ne supprimez pas les politiques AWS Identity and Access Management (IAM) qui accordent à l'entité (par exemple, utilisateur, groupe ou rôle) l'autorisation d'effectuer une restauration. Sinon, il peut en résulter un comportement inattendu. Supposons, par exemple, que vous ayez supprimé les autorisations d'écriture pour une table alors que cette table était en cours de restauration. Dans ce cas, l'opération `RestoreTableToPointInTime` sous-jacente ne peut pas écrire les données restaurées dans la table.
> Vous pouvez modifier ou supprimer des autorisations uniquement lorsque l'opération de restauration est terminée.

Restauration d'une table à un instant dans le passé avec CQL

La procédure suivante montre comment utiliser le code CQL pour restaurer une table existante nomméemytable à un instant dans le passé.

> **Note**
>
> Cette procédure part du principe que vous avez activé point-in-time la restauration. Pour activer le PITR sur le tableau, suivez les étapes décrites dansthe section called “CQL”.

1. Vous pouvez restaurer une table active à une heure point-in-time comprise entre l'heure actuelleearliest_restorable_timestamp et l'heure actuelle. L'heure actuelle est l'heure par défaut.

   Pour confirmer que la point-in-time restauration est activée pour lamytable table, interrogez la tablesystem_schema_mcs.tables comme suit.

   ```
   SELECT custom_properties
   FROM system_schema_mcs.tables
   WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
   ```

   Laoint-in-time restauration P est activée comme dans l'exemple de sortie suivant.

```
custom_properties
-----------------
{
  ...,
    "point_in_time_recovery": {
    "earliest_restorable_timestamp":"2020-06-30T19:19:21.175Z"
    "status":"enabled"
  }
}
```

2. Restaurez le tableau à un point dans le temps, spécifié par `arestore_timestamp` au format ISO 8601. Dans ce cas, le point de`mytable` table est restauré à l'heure actuelle. Vous pouvez omettre la`WITH restore_timestamp = ...` clause. Sans la clause, l'horodatage actuel est utilisé.

   ```
   RESTORE TABLE mykeyspace.mytable_restored
   FROM TABLE mykeyspace.mytable;
   ```

   Vous pouvez également restaurer à un instant spécifique dans le passé. Vous pouvez spécifier n'importe quel instant dans le passé au cours des 35 derniers jours. Par exemple, la commande suivante restaure la table à la `EarliestRestorableDateTime`.

   ```
   RESTORE TABLE mykeyspace.mytable_restored
   FROM TABLE mykeyspace.mytable
   WITH restore_timestamp = '2020-06-30T19:19:21.175Z';
   ```

   Pour une description complète de la syntaxe, reportez-vousthe section called “RESTAURER LA TABLE” à la référence du langage.

Pour vérifier que la restauration de la table a été réussie, interrogez le`system_schema_mcs.tables` pour confirmer l'état de la table.

```
SELECT status
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable_restored'
```

La requête génère le résultat suivant.

```
status
------
RESTORING
```

La table en cours de restauration s'affiche avec le statut Restauration en cours. Une fois le processus de restauration terminé, l'état de la table `mytable_restored` devient Active.

> **Important**
>
> Pendant la restauration, ne modifiez ou ne supprimez pas les politiques AWS Identity and Access Management (IAM) qui accordent à l'entité (par exemple, utilisateur, groupe ou rôle) l'autorisation d'effectuer une restauration. Sinon, il peut en résulter un comportement inattendu. Supposons, par exemple, que vous ayez supprimé les autorisations d'écriture pour une table alors que cette table était en cours de restauration. Dans ce cas, l'opération `RestoreTableToPointInTime` sous-jacente ne peut pas écrire les données restaurées dans la table.
> Vous pouvez modifier ou supprimer des autorisations uniquement lorsque l'opération de restauration est terminée.

Restaurer une table supprimée à l'aide deAWS CLI

La procédure suivante montre comment utiliser le pourAWS CLI restaurer une table supprimée dont le nom correspond`myTable` à l'heure de la suppression.

> **Note**
>
> Cette procédure suppose que le PITR a été activé sur la table supprimée.

1. Supprimez le point que vous avez créé dans le didacticiel précédent.

   ```
   aws keyspaces delete-table --keyspace-name 'myKeyspace' --table-name 'myTable'
   ```

2. Restaurez la table supprimée au moment de la suppression à l'aide de la commande suivante.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored2'
```

La sortie de cette commande renvoie l'ARN de la table restaurée.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored2"
}
```

Restaurer une table supprimée avec CQL

La procédure suivante montre comment utiliser CQL pour restaurer une table supprimée dont le nom indique`mytable` l'heure de la suppression.

> **Note**
>
> Cette procédure suppose que le PITR a été activé sur la table supprimée.

1. Pour confirmer que point-in-time la restauration est activée pour une table supprimée, interrogez la table système. Seules les tables dont point-in-time la restauration est activée sont affichées.

```
SELECT custom_properties
FROM system_schema_mcs.tables_history
WHERE keyspace_name = 'mykeyspace' AND table_name = 'my_table';
```

La requête génère le résultat suivant.

```
custom_properties
------------------
{
    ...,
   "point_in_time_recovery":{
      "restorable_until_time":"2020-08-04T00:48:58.381Z",
      "status":"enabled"
```

```
   }
}
```

2. Restaurez le tableau au moment de sa suppression à l'aide de l'exemple d'instruction suivant.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

Expiration de données en utilisant Amazon Keyspaces (TTL)

Amazon Keyspaces (pour Apache Cassandra) vous aide à simplifier la logique de votre application et à optimiser le prix de stockage en faisant expirer automatiquement les données des tables. Les données dont vous n'avez plus besoin sont automatiquement supprimées de votre tableau en fonction de la valeur Time to Live que vous avez définie. Cela facilite le respect des politiques de conservation des données basées sur les exigences commerciales, industrielles ou réglementaires qui définissent la durée de conservation des données ou précisent quand les données doivent être supprimées.

Par exemple, vous pouvez utiliser le TTL dans une AdTech application pour planifier le moment où les données relatives à des publicités spécifiques expirent et ne sont plus visibles pour les clients. Vous pouvez également utiliser le TTL pour supprimer automatiquement les anciennes données et économiser sur vos coûts de stockage. Vous pouvez définir une valeur TTL par défaut pour l'ensemble du tableau et remplacer cette valeur pour les lignes et les colonnes individuelles. Les opérations TTL n'ont aucun impact sur les performances de votre application. De plus, le nombre de lignes et de colonnes marquées comme expirant par le protocole TTL n'affecte pas la disponibilité de votre tableau.

Amazon Keyspaces filtre automatiquement les données expirées afin que les données expirées ne soient pas renvoyées dans les résultats des requêtes ou disponibles pour une utilisation dans les instructions en langage de manipulation de données (DML). Amazon Keyspaces supprime généralement les données expirées du stockage dans les 10 jours suivant la date d'expiration. Dans de rares cas, Amazon Keyspaces peut ne pas être en mesure de supprimer des données dans les 10 jours en cas d'activité soutenue sur la partition de stockage sous-jacente afin de protéger la disponibilité. Dans ces cas, Amazon Keyspaces continue de tenter de supprimer les données expirées une fois que le trafic sur la partition diminue. Une fois les données définitivement supprimées du stockage, vous n'avez plus à payer de frais de stockage. Pour plus d'informations, veuillez consulter the section called “Comment ça marche”.

Vous pouvez définir, modifier ou désactiver les paramètres TTL par défaut pour les tables nouvelles et existantes à l'aide de la console ou du langage CQL (Cassandra Query Language). Sur les tables dont le TTL est configuré par défaut, vous pouvez utiliser le langage CQL (Cassandra Query Language) pour remplacer les paramètres TTL par défaut et appliquer des valeurs TTL personnalisées aux lignes et aux colonnes. Pour plus d'informations, veuillez consulter the section called “Comment utiliser le Time-à-live ?”.

La tarification TTL est basée sur la taille des lignes supprimées ou mises à jour à l'aide de Time to Live. Les opérations TTL sont mesurées en unités de`TTL deletes`. Une suppression TTL est consommée par Ko de données par ligne supprimée ou mise à jour. Par exemple, pour mettre à jour une ligne qui contient 2,5 Ko de données et pour supprimer une ou plusieurs colonnes de la ligne en même temps, vous devez effectuer trois suppressions TTL. Ou bien, pour supprimer une ligne entière contenant 3,5 Ko de données, quatre suppressions TTL sont nécessaires. Une suppression TTL est consommée par Ko de données supprimées par ligne. Pour de plus amples informations sur la tarification, veuillez consulter la tarification Amazon Keyspaces (pour Apache Cassandra).

Rubriques

- Fonctionnement : Time Keyspaces (TTL).
- Fonctionnement d'un Time-à-live (Tlive) en live (Tlive) dans le live

Fonctionnement : Time Keyspaces (TTL).

Amazon Keyspaces Time to Live (TTL) est entièrement géré. Vous n'avez pas à gérer les paramètres système de bas niveau tels que les stratégies de compactage. Les données expirent au moment que vous spécifiez, et Amazon Keyspaces supprime automatiquement les données expirées (généralement dans les 10 jours) sans affecter les performances ou la disponibilité de votre application.

Les données Expiration sont marquées pour suppression et ne sont pas disponibles pour les instructions en langage de manipulation de données (DML). Au fur et à mesure que vous effectuez des lectures et des écritures sur des lignes contenant des données expirées, ces dernières continuent de prendre en compte les unités de capacité de lecture (RCU) et les unités de capacité d'écriture (WCU) jusqu'à ce qu'elles soient supprimées de la mémoire.

Rubriques

- Définition de la valeur TTL par défaut pour une table
- Définition de valeurs TTL personnalisées pour les lignes et les colonnes
- Activation du TTL sur les tables
- Amazon Keyspaces Time to Live et intégration auxAWS services

Définition de la valeur TTL par défaut pour une table

Dans Amazon Keyspaces, vous pouvez définir une valeur TTL par défaut pour toutes les lignes d'un tableau lors de sa création. Vous pouvez également modifier un tableau existant pour définir ou modifier la valeur TTL par défaut pour les nouvelles lignes insérées dans le tableau. La modification de la valeur TTL par défaut d'une table ne modifie pas la valeur TTL des données existantes dans la table. La valeur TTL d'une table est zéro, ce qui signifie que les données n'expirent pas automatiquement. Si la valeur TTL par défaut d'une table est supérieure à zéro, un horodatage d'expiration est ajouté à chaque ligne.

Amazon Keyspaces calcule un nouvel horodatage TTL chaque fois que les données sont mises à jour. Les valeurs TTL sont définies en secondes et la valeur de paramétrage de 630 720 000 secondes, soit l'équivalent de 20 ans. Pour plus d'informations sur la définition, la modification et la désactivation de la valeur TTL par défaut pour les tables à l'aide duAWS Management Console ou CQL, consultezthe section called “Comment utiliser le Time-à-live ?”.

Définition de valeurs TTL personnalisées pour les lignes et les colonnes

> **Note**
>
> Avant de définir des valeurs TTL personnalisées pour les lignes et les colonnes, le TTL doit d'abord être activé sur la table. Pour plus d'informations, veuillez consulter the section called “Comment activer le Time-à-live (Tlive) sur des tables existantes à l'aide de propriétés personnalisées”.

Pour remplacer la valeur TTL par défaut d'une table ou pour définir des dates d'expiration pour des lignes individuelles, vous pouvez utiliser les instructions du langage de manipulation de données (DML) CQL suivantes :

- `INSERT`— Permet d'insérer une nouvelle ligne de données avec une valeur TTL définie.
- `UPDATE`— Permet de modifier une ligne de données existante avec une nouvelle valeur TTL.

La définition des valeurs TTL pour les lignes est prioritaire par rapport au paramètre TTL par défaut pour la table.

Pour consulter la syntaxe CQL et des exemples, reportez-vous à la sectionthe section called “À utiliser `INSERT` pour modifier des paramètres de durée de vie (TTL) à l'aide de CQL”.

Pour remplacer ou définir des valeurs TTL pour des colonnes individuelles, vous pouvez mettre à jour le paramètre TTL pour un sous-ensemble de colonnes au sein de lignes existantes à l'aide de l'instruction CQL DML suivante :

- `UPDATE`— Permet de mettre à jour une colonne de données.

La définition des valeurs TTL pour les colonnes est prioritaire par rapport au paramètre TTL par défaut du tableau et à tout paramètre TTL personnalisé pour la ligne. Pour consulter la syntaxe CQL et des exemples, reportez-vous à la sectionthe section called “À utiliser `UPDATE` pour modifier des paramètres de durée de vie (TTL) à l'aide de CQL”.

Activation du TTL sur les tables

Le TTL est automatiquement activé pour les tables lorsque vous spécifiez une`default_time_to_live` valeur supérieure à 0 dans l'une`CREATE TABLE` ou l'autre`ALTER TABLE` des instructions. Si vous ne spécifiez pas de valeur`default_time_to_live` pour la table, mais que vous souhaitez spécifier des valeurs TTL personnalisées pour les lignes ou les colonnes à l'aide d'`UPDATE`opérations`INSERT` or, vous devez d'abord activer le TTL pour la table. Vous pouvez activer le TTL pour une table à l'aide de la propriété`ttl` personnalisée.

Lorsque vous activez le TTL sur une table, Amazon Keyspaces commence à stocker des métadonnées supplémentaires liées au TTL pour chaque ligne. En outre, le TTL utilise des horodatages d'expiration pour suivre l'expiration des lignes ou des colonnes. Les horodatages sont stockés sous forme de métadonnées de ligne et contribuent au coût de stockage de la ligne.

Une fois que la fonction TTL est activée, vous ne pouvez pas la désactiver pour une table. La définition de`default_time_to_live` la table sur 0 désactive les délais d'expiration par défaut pour les nouvelles données, mais cela ne désactive pas la fonctionnalité TTL et ne rétablit pas les métadonnées de stockage ou le comportement d'écriture d'origine d'Amazon Keyspaces.

Amazon Keyspaces Time to Live et intégration auxAWS services

La métrique TTL suivante est disponible sur Amazon CloudWatch pour permettre une surveillance continue.

- `TTLDeletes`— Les unités utilisées pour supprimer ou mettre à jour des données d'affilée à l'aide de Time to Live (TTL).

Pour plus d'informations sur la surveillance CloudWatch des métriques, consultezthe section called “Surveillance avec CloudWatch”.

Lorsque vous l'utilisezAWS CloudFormation, vous pouvez activer le TTL lors de la création d'une table Amazon Keyspaces. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur AWS CloudFormation.

Fonctionnement d'un Time-à-live (Tlive) en live (Tlive) dans le live

Vous pouvez utiliser la console Amazon Keyspaces (pour Apache Cassandra) ou CQL pour activer, mettre à jour et désactiver les paramètres de Time-à-live.

Rubriques

Pour créer une nouvelle table avec les paramètres de durée de vie (TTL) activés (TTL) par défaut (console)

Suivez ces étapes pour créer un nouveau tableau avec les paramètres de Time-à-live en utilisant la console Amazon Keyspaces.

1. Connectez-vous à la AWS Management Console et ouvrez la console Amazon Keyspaces à la page https://console.aws.amazon.com/keyspaces/home.

2. Dans le panneau de navigation, choisissez Tables, puis Create table (Créer une table).
3. Sur la page Créer un tableau, dans la section Détails du tableau, sélectionnez un espace clavier et donnez un nom au nouveau tableau.
4. Dans la section Schéma, créez le schéma de votre table.
5. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
6. Continuez vers le Time-à-live (Tlive).

   Au cours de cette étape, vous sélectionnez les paramètres TTL par défaut pour le tableau.

   Pour la période TTL par défaut, entrez le délai d'expiration et choisissez l'unité de temps que vous avez saisie, par exemple les secondes, les jours ou les années. Amazon Keyspaces stockera la valeur en quelques secondes.
7. Choisissez Créer un tableau. Votre table est créée avec la valeur TTL par défaut spécifiée.

> **Note**
>
> Vous pouvez remplacer le paramètre TTL par défaut du tableau pour des lignes ou des colonnes spécifiques en utilisant le langage de manipulation de données (DML) dans l'éditeur CQL.

Pour mettre à jour les paramètres de durée de vie (TTL) sur des tables existantes (console) sur des tables existantes (console)

Suivez ces étapes pour mettre à jour les paramètres Time to Live des tables existantes à l'aide de la console Amazon Keyspaces.

1. Connectez-vous à la AWS Management Console et ouvrez la console Amazon Keyspaces à la page https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la version que vous souhaitez mettre à jour, puis l'onglet Paramètres supplémentaires.
3. Passez à Time to Live (TTL) et choisissez Modifier.
4. Pour la période TTL par défaut, entrez le délai d'expiration et choisissez l'unité de temps que vous avez saisie, par exemple les secondes, les jours ou les années. Amazon Keyspaces

stockera la valeur en quelques secondes. Cela ne modifie pas la valeur TTL des lignes existantes.

5. Lorsque les paramètres TTL sont définis, choisissez Enregistrer les modifications.

Pour désactiver les paramètres de durée de vie (TTL) sur des tables existantes (console) sur des tables existantes (console)

Suivez ces étapes pour désactiver les paramètres Time to Live pour les tables existantes à l'aide des Amazon KeyspacesAWS Management Console.

1. Connectez-vous à la AWS Management Console et ouvrez la console Amazon Keyspaces à la page https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la version que vous souhaitez mettre à jour, puis l'onglet Paramètres supplémentaires.
3. Passez à Time to Live (TTL) et choisissez Modifier.
4. Sélectionnez Période TTL par défaut et définissez la valeur sur zéro. Cela désactive le TTL pour la table par défaut pour les données future. Cela ne modifie pas la valeur TTL des lignes existantes.
5. Lorsque les paramètres TTL sont définis, choisissez Enregistrer les modifications.

Pour créer une nouvelle table avec les paramètres de durée de vie (TTL) activés à l'aide de CQL

Activez la TTL lorsque vous créez une nouvelle table avec la valeur TTL par défaut définie sur 3 024 000 secondes, ce qui représente 35 jours.

```
CREATE TABLE my_table (
                userid uuid,
                time timeuuid,
                subject text,
                body text,
                user inet,
                PRIMARY KEY (userid, time)
                ) WITH default_time_to_live = 3024000;
```

Pour confirmer les paramètres TTL de la nouvelle table, utilisez l'`cqlsh` `describe`instruction comme indiqué dans l'exemple suivant. La sortie indique le paramètre TTL par défaut pour le tableau sous `default_time_to_live` la forme.

```
describe my_table;
```

À utiliser **`ALTER TABLE`** pour modifier les paramètres de durée de vie (TTL) à l'aide de CQL

Mettez à jour les paramètres TTL de la table existante à 2 592 000 secondes, ce qui représente 30 jours.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Pour confirmer les paramètres TTL de la table mise à jour, utilisez l'`cqlsh` `describe`instruction comme indiqué dans l'exemple suivant. La sortie indique le paramètre TTL par défaut pour le tableau sous `default_time_to_live` la forme.

```
describe my_table;
```

Comment activer le Time-à-live (Tlive) sur de nouvelles tables à l'aide de propriétés personnalisées

Pour activer les paramètres personnalisés Time to Live qui peuvent être appliqués aux lignes et aux colonnes sans activer les paramètres TTL par défaut pour l'ensemble du tableau, vous pouvez utiliser l'instruction CQL suivante.

```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH CUSTOM_PROPERTIES={'ttl':
{'status': 'enabled'}};
```

Une fois `ttl` activé, vous ne pouvez pas le désactiver pour la table.

Comment activer le Time-à-live (Tlive) sur des tables existantes à l'aide de propriétés personnalisées

Pour activer les paramètres personnalisés Time to Live qui peuvent être appliqués aux lignes et aux colonnes sans activer les paramètres TTL par défaut pour l'ensemble du tableau, vous pouvez utiliser l'instruction CQL suivante.

```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

Une fois `ttl` activé, vous ne pouvez pas le désactiver pour la table.

À utiliser **INSERT** pour modifier des paramètres de durée de vie (TTL) à l'aide de CQL

L'instruction CQL suivante insère une ligne de données dans le tableau et modifie le paramètre TTL par défaut à 259 200 secondes (ce qui équivaut à 3 jours).

```
INSERT INTO my_table (userid, time, subject, body, user)
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello','205.212.123.123')
        USING TTL 259200;
```

Pour confirmer les paramètres TTL de la ligne insérée, utilisez l'instruction suivante.

```
SELECT TTL (subject) from my_table;
```

À utiliser **UPDATE** pour modifier des paramètres de durée de vie (TTL) à l'aide de CQL

Pour modifier les paramètres TTL de la colonne « objet » insérée précédemment de 259 200 secondes (3 jours) à 86 400 secondes (un jour), utilisez l'instruction suivante.

```
UPDATE my_table USING TTL 86400 set subject = 'Updated Message' WHERE userid =
 B79CB3BA-745E-5D9A-8903-4A02327A7E09 and time = 96a29100-5e25-11ec-90d7-b5d91eceda0a;
```

Vous pouvez exécuter une simple requête de sélection pour voir l'enregistrement mis à jour avant la date d'expiration.

```
SELECT * from my_table;
```

La requête affiche la sortie suivante.

```
userid                               | time                                 | body  |
 subject         | user
```

```
--------------------------------------+--------------------------------------+-------
+-----------------+----------------
b79cb3ba-745e-5d9a-8903-4a02327a7e09  | 96a29100-5e25-11ec-90d7-b5d91eceda0a | Hello |
 Updated Message | 205.212.123.123
50554d6e-29bb-11e5-b345-feff819cdc9f  | cf03fb21-59b5-11ec-b371-dff626ab9620 | Hello |
        Message | 205.212.123.123
```

Pour confirmer que l'expiration a bien été effectuée, exécutez à nouveau la même requête après le délai d'expiration configuré.

```
SELECT * from my_table;
```

La requête affiche le résultat suivant une fois que la colonne « objet » a expiré.

```
userid                               | time                                 | body  |
 subject | user
--------------------------------------+--------------------------------------+-------
+---------+-----------------
b79cb3ba-745e-5d9a-8903-4a02327a7e09  | 96a29100-5e25-11ec-90d7-b5d91eceda0a | Hello |
   null | 205.212.123.123
50554d6e-29bb-11e5-b345-feff819cdc9f  | cf03fb21-59b5-11ec-b371-dff626ab9620 | Hello |
 Message | 205.212.123.123
```

Utilisation des horodatages côté client dans Amazon Keyspaces

Dans Amazon Keyspaces, les horodatages côté client sont des horodatages compatibles avec Cassandra qui sont conservés pour chaque cellule de votre tableau. Vous pouvez utiliser des horodatages côté client pour la résolution des conflits en laissant vos applications clientes déterminer l'ordre des écritures. Par exemple, lorsque les clients d'une application distribuée dans le monde entier mettent à jour les mêmes données, les horodatages côté client conservent l'ordre dans lequel les mises à jour ont été effectuées sur les clients. Amazon Keyspaces utilise ces horodatages pour traiter les écritures. Pour plus d'informations, veuillez consulter the section called “Comment ça marche”.

Une fois les horodatages côté client activés pour une table, vous pouvez spécifier un horodatage à l'aide de la`USING TIMESTAMP` clause dans votre requête CQL en langage de manipulation de données (DML). Si vous ne spécifiez pas d'horodatage dans votre requête CQL, Amazon Keyspaces utilise l'horodatage transmis par votre pilote client. Si le pilote client ne fournit pas d'horodatage, Amazon Keyspaces attribue automatiquement un horodatage au niveau de la cellule. Pour rechercher des horodatages, vous pouvez utiliser la`WRITETIME` fonction dans votre instruction DML. Pour plus d'informations, veuillez consulter the section called “Comment utiliser les horodatages côté client”.

Amazon Keyspaces ne facture aucun supplément pour activer les horodatages côté client. Toutefois, les horodatages côté client vous permettent de stocker et d'écrire des données supplémentaires pour chaque valeur de votre ligne. Cela peut entraîner une utilisation supplémentaire du stockage et, dans certains cas, une augmentation du débit. Pour en savoir plus sur l'estimation de l'impact sur la taille des lignes, consultezthe section called “Comment ça marche”. Pour en savoir plus sur la tarification d'Amazon Keyspaces, consultez la tarification d'Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Comment fonctionnent les horodatages côté client dans Amazon Keyspaces

Les horodatages côté client Amazon Keyspaces sont entièrement gérés. Vous n'avez pas à gérer les paramètres de base du système tels que les stratégies de nettoyage et de compactage.

Lorsque vous supprimez des données, les lignes sont marquées pour plus amples informations sur les lignes qui sont plus amples plus amples plus amples plus amples plus amples plus amples plus informations Amazon Keyspaces supprime automatiquement les données tombales (généralement dans les 10 jours) sans affecter les performances ou la disponibilité de vos applications. Les données plus informations ne sont pas disponibles pour les instructions DML (Data Manipulation Language). Au fur et à mesure que vous continuez à effectuer des lectures et des écritures sur des lignes contenant des données gravées, celles-ci continuent à être prises en compte dans le stockage, les unités de capacité de lecture (RCU) et les unités de capacité d'écriture (WCU) jusqu'à ce qu'elles soient supprimées du stockage.

Rubriques

Comment fonctionnent les horodatages côté client dans Amazon Keyspaces

Lorsque les horodatages côté client sont activés dans Amazon Keyspaces, chaque colonne de chaque ligne contient un horodatage. Ces horodatages occupent environ 20 à 40 octets (selon vos données) et contribuent au coût de stockage et de débit de la ligne. Ces octets de métadonnées sont également pris en compte dans votre quota de taille de ligne de 1 Mo. Pour déterminer l'augmentation globale de l'espace de stockage (afin de garantir que la taille des lignes reste inférieure à 1 Mo), prenez en compte le nombre de colonnes de votre tableau et le nombre d'éléments de collection dans chaque ligne. Par exemple, si une table comporte 20 colonnes, chaque colonne stockant 40 octets de données, la taille de la ligne passe de 800 octets à 1 200 octets. Pour plus plus plus plus plus plus informations sur plus plus plus plus plus plus plus plus plus plus plus plus plus plus plus plus plus plus plus plusthe section called “Calcul de la taille des lignes” informations sur Outre les 400 octets supplémentaires pour le stockage, dans cet exemple, le nombre d'unités de capacité d'écriture (WCU) consommées par écriture passe de 1 WCU à 2 WCU. Pour plus

d'informations sur le calcul de la capacité de lecture et d'écriture, consultezthe section called “Modes de capacité de lecture/écriture”.

Une fois que les horodatages côté client ont été activés pour une table, vous ne pouvez pas la désactiver. De plus, les horodatages ne peuvent pas l'être. AinsiNULL, si aucun horodatage côté client n'est fourni par l'instruction CQL ou le pilote client, un horodatage généré par Amazon Keyspaces est automatiquement ajouté.

Horodatage côté client Amazon Keyspaces et intégration auxAWS services

La métrique d'horodatage côté client suivante est disponible sur Amazon CloudWatch pour permettre une surveillance continue.

- `SystemReconciliationDeletes`— Nombre d'opérations de suppression requises pour supprimer les données tombstone.

Pour plus plus plus plus plus informations sur plus plus plus plus plus plus plus plus CloudWatch plus plus plus plus plusthe section called “Surveillance avec CloudWatch” informations sur

Utilisation des horodatages côté client dans Amazon Keyspaces

Vous pouvez utiliser la console Amazon Keyspaces (pour Apache Cassandra), le langage CQL (Cassandra Query Language), leAWS SDK et leAWS Command Line Interface (AWS CLI) pour activer les horodatages côté client. Cette section fournit des exemples montrant comment activer les horodatages côté client sur des tables nouvelles et existantes et comment utiliser les horodatages côté client dans les requêtes. Pour de plus amples informations sur l'API, veuillez consulter la référence de l'API Amazon Keyspaces.

> **Important**
>
> Les horodatages côté client ne peuvent pas être désactivés. L'activation des horodatages côté client est une modification ponctuelle. Amazon Keyspaces ne propose pas d'option permettant de le désactiver sans supprimer le tableau.

Rubriques

- Création d'un nouveau tableau avec les horodatages côté client activés (console)

- Activation de l'horodatage côté client sur les tables existantes (console)
- Création d'une nouvelle table avec les horodatages côté client activés (CQL)
- Activation de l'horodatage côté client pour les tables existantes à l'aide deALTER TABLE (CQL)
- Création d'une nouvelle table avec les horodatages côté client activés (CLI)
- Activer les horodatages côté client sur une table existante (CLI)
- Utilisation d'horodatages côté client dans les instructions DML (Data Manipulation Language)

Création d'un nouveau tableau avec les horodatages côté client activés (console)

Suivez ces étapes pour créer une table avec des horodatages côté client pour activer la console Amazon Keyspaces.

Pour créer une nouvelle table avec des horodatages côté client (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Tables, puis Create table (Créer une table).
3. Sur la page Créer une table, dans la section Détails de la table, sélectionnez un espace clé et donnez un nom à la nouvelle table.
4. Dans la section Schéma, créez le schéma de votre table.
5. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
6. Passez à l'horodatage côté client.

   Choisissez Activer les horodatages côté client pour activer les horodatages côté client pour le tableau.
7. Choisissez Créer un tableau. Votre tableau est créé avec les horodatages côté client activés.

Activation de l'horodatage côté client sur les tables existantes (console)

Suivez ces étapes pour activer les horodatages côté client pour les tables existantes à l'aide d'Amazon KeyspacesAWS Management Console.

Pour activer l'horodatage côté client pour une table existante (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez la table que vous souhaitez mettre à jour, puis l'onglet Paramètres supplémentaires.
3. Dans l'onglet Paramètres supplémentaires, accédez à Modifier les horodatages côté client et sélectionnez Activer les horodatages côté client
4. Choisissez Enregistrer les modifications pour modifier les paramètres du tableau.

Création d'une nouvelle table avec les horodatages côté client activés (CQL)

Pour activer les horodatages côté client lorsque vous créez une nouvelle table, vous pouvez utiliser l'instruction CQL suivante.

```
CREATE TABLE my_table (
   userid uuid,
   time timeuuid,
   subject text,
   body text,
   user inet,
   PRIMARY KEY (userid, time)
) WITH CUSTOM_PROPERTIES = {'client_side_timestamps': {'status': 'enabled'}};
```

Pour confirmer les paramètres d'horodatage côté client pour la nouvelle table, utilisez uneSELECT instruction pour les vérifier,custom_properties comme indiqué dans l'exemple suivant.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

La sortie de cette instruction indique l'état des horodatages côté client.

```
'client_side_timestamps': {'status': 'enabled'}
```

Activation de l'horodatage côté client pour les tables existantes à l'aide de`ALTER TABLE` (CQL)

Pour activer les horodatages côté client pour une table existante, vous pouvez utiliser l'instruction CQL suivante.

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps': {'status':
 'enabled'}};;
```

Pour confirmer les paramètres d'horodatage côté client pour la nouvelle table, utilisez une`SELECT` instruction pour les vérifier,`custom_properties` comme indiqué dans l'exemple suivant.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

La sortie de cette instruction indique l'état des horodatages côté client.

```
'client_side_timestamps': {'status': 'enabled'}
```

Création d'une nouvelle table avec les horodatages côté client activés (CLI)

Pour activer les horodatages côté client lorsque vous créez une table, vous pouvez utiliser l'instruction CLI suivante.

```
./aws keyspaces create-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED' \
--schema-definition 'allColumns=[{name=id,type=int},{name=date,type=timestamp},
{name=name,type=text}],partitionKeys=[{name=id}]'
```

Pour vérifier que les horodatages côté client sont activés pour la nouvelle table, exécutez le code suivant.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

La sortie doit ressembler à cet exemple.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-2:555555555555:/keyspace/my_keyspace/
table/my_table",
    "creationTimestamp": 1662681206.032,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1662681206.032
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
```

```
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Activer les horodatages côté client sur une table existante (CLI)

Pour activer les horodatages côté client pour une table existante à l'aide de l'interface de ligne de commande, vous pouvez utiliser le code suivant.

```
./aws keyspaces update-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED'
```

Pour vérifier que les horodatages côté client sont activés pour la table, exécutez le code suivant.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

La sortie doit ressembler à cet exemple.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-2:555555555555:/keyspace/my_keyspace/
table/my_table",
    "creationTimestamp": 1662681312.906,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
```

```
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1662681312.906
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Utilisation d'horodatages côté client dans les instructions DML (Data Manipulation Language)

Une fois que vous avez activé les horodatages côté client, vous pouvez transmettre l'horodatage dans vos`DELETE` instructions`INSERTUPDATE`, et avec la`USING TIMESTAMP` clause. La valeur d'horodatage`bigint` représente le nombre de microsecondes écoulées depuis l'heure de base standard connue sous le nom deepoch : 1er janvier 1970 à 00:00:00 GMT. L'horodatage fourni par le client doit se situer entre 2 jours dans le passé et 5 minutes dans le futur par rapport à l'heure actuelle de l'horloge murale. Amazon Keyspaces conserve les métadonnées d'horodatage pendant toute la durée de vie des données. Vous pouvez utiliser cette`WRITETIME` fonction pour rechercher des horodatages survenus au cours des années passées. Pour de plus amples informations sur la syntaxe CQL, consultezthe section called “Instructions DML”.

L'instruction CQL suivante est un exemple d'utilisation d'un horodatage en tant que`update_parameter`.

```
INSERT INTO catalog.book_awards (year, award, rank, category, book_title, author,
 publisher)
   VALUES (2022, 'Wolf', 4, 'Non-Fiction', 'Science Update', 'Ana Carolina Silva',
 'SomePublisher')
   USING TIMESTAMP 1669069624;
```

Si vous ne spécifiez pas d'horodatage dans votre requête CQL, Amazon Keyspaces utilise l'horodatage transmis par votre pilote client. Si aucun horodatage n'est fourni par le pilote client, Amazon Keyspaces attribue un horodatage côté serveur pour votre opération d'écriture.

Pour voir la valeur d'horodatage stockée pour une colonne spécifique, vous pouvez utiliser la`WRITETIME` fonction dans une`SELECT` instruction, comme illustré dans l'exemple suivant.

```
SELECT year, award, rank, category, book_title, author, publisher, WRITETIME(year),
 WRITETIME(award), WRITETIME(rank),
  WRITETIME(category), WRITETIME(book_title), WRITETIME(author), WRITETIME(publisher)
 from catalog.book_awards;
```

Création de ressources Amazon Keyspaces avecAWS CloudFormation

Amazon Keyspaces (pour Apache Cassandra) est intégré àAWS CloudFormation, un service qui vous aide à modéliser et à configurer vosAWS ressources, afin d'accélérer la création et la gestion de vos ressources et de votre infrastructure. Vous créez un modèle qui décrit toutes lesAWS ressources que vous voulez (telles que des espaces et des tables), etAWS CloudFormation s'occupe de leur mise en service et de leur configuration.

Lorsque vous utilisezAWS CloudFormation, vous pouvez réutiliser votre modèle pour configurer vos ressources Amazon Keyspaces de manière cohérente et répétée. Il suffit de décrire vos ressources une fois, puis mettez-le en service autant de fois que vous le souhaitez dans plusieursComptes AWS comptes et régions.

Amazon Keyspaces etAWS CloudFormation modèles

Pour mettre en service et configurer des ressources pour Amazon Keyspaces, vous devez maîtriser les AWS CloudFormationmodèles. Les modèles sont des fichiers texte formatés en JSON ou YAML. Ces modèles décrivent les ressources que vous souhaitez allouer dans vos piles AWS CloudFormation. Si JSON ou YAML ne vous est pas familier, vous pouvez utiliser AWS CloudFormation Designer pour vous aider à démarrer avec des modèles AWS CloudFormation. Pour plus d'ABABACAWS CloudFormation pour. dans le guide deAWS CloudFormation l'utilisateur.

Amazon Keyspaces prend en charge la création d'espaces clés et de tableaux dansAWS CloudFormation. Pour les tables que vous créez à l'aide deAWS CloudFormation modèles, vous pouvez spécifier le schéma, le mode lecture/écriture et les paramètres de débit provisionnés. Pour plus d'ABL, y compris des exemples de modèles JSON et YAML pour les espaces et les tables, consultez la Référence de type de ressource Cassandra dans le Guide deAWS CloudFormation l'utilisateur.

En savoir plus sur AWS CloudFormation

Pour en savoir plus sur AWS CloudFormation, consultez les ressources suivantes :

- AWS CloudFormation
- Guide de l'utilisateur AWS CloudFormation

- AWS CloudFormationGuide de l'utilisateur de l'interface de ligne de commande

Surveillance d'Amazon Keyspaces (pour Apache Cassandra)

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon Keyspaces et de vos autres AWS solutions. AWSfournit les outils de surveillance suivants pour surveiller Amazon Keyspaces, signaler un problème et prendre des mesures automatiques le cas échéant :

- Amazon Keyspaces propose un tableau de bord préconfiguré AWS Management Console indiquant la latence et les erreurs agrégées dans toutes les tables du compte.
- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques à l'aide de tableaux de bord personnalisés. Par exemple, vous pouvez créer une base de référence pour les performances normales d'Amazon Keyspaces dans votre environnement en mesurant les performances à différents moments et dans différentes conditions de charge. Lorsque vous surveillez Amazon Keyspaces, stockez les données de surveillance historiques afin de pouvoir les comparer aux données de performance actuelles, d'identifier les modèles de performances normaux et les anomalies de performance, et de concevoir des méthodes pour résoudre les problèmes. Pour établir une base de référence, vous devez au minimum surveiller les erreurs système. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- CloudWatch Les alarmes Amazon surveillent une seule métrique sur une période que vous spécifiez et exécutent une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. Par exemple, si vous utilisez Amazon Keyspaces en mode provisionné avec mise à l'échelle automatique des applications, l'action est une notification envoyée par Amazon Simple Notification Service (Amazon SNS) pour évaluer une politique d'Application Auto Scaling.

  CloudWatch les alarmes n'appellent pas d'actions simplement parce qu'elles sont dans un état particulier. L'état doit avoir changé et avoir été maintenu pendant un nombre de périodes spécifié. Pour plus d'informations, veuillez consulter Surveillance d'Amazon Keyspaces avec Amazon CloudWatch.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir de tables Amazon Keyspaces et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux

dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur Amazon CloudWatch Logs.

- AWS CloudTrail capture les appels d'API et les événements associés créés par votre Compte AWS ou au nom de celui-ci et livre les fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes qui ont appelé AWS, l'adresse IP source à partir de laquelle les appels ont été émis, ainsi que le moment où les appels ont eu lieu. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur AWS CloudTrail.

Amazon EventBridge est un service de bus d'événements sans serveur qui permet de connecter facilement vos applications à des données provenant de diverses sources. EventBridge fournit un flux de données en temps réel à partir de vos propres applications, applications oftware-as-a S-Service (SaaS) et AWS services et achemine ces données vers des cibles telles que Lambda. Cela vous permet de surveiller les événements qui se produisent dans les services et de créer des architectures basées sur les événements. Pour plus d'informations, consultez le guide de EventBridge l'utilisateur Amazon.

Rubriques

- Surveillance d'Amazon Keyspaces avec Amazon CloudWatch
- Journalisation des appels d'API Amazon Keyspaces avec AWS CloudTrail

Surveillance d'Amazon Keyspaces avec Amazon CloudWatch

Vous pouvez surveiller Amazon Keyspaces à l'aide d'Amazon CloudWatch, qui collecte des données brutes et les transforme en indicateurs lisibles en temps quasi réel. Ces statistiques sont enregistrées pour une durée de 15 mois ; par conséquent, vous pouvez accéder aux informations historiques et acquérir un meilleur point de vue de la façon dont votre service ou application web s'exécute.

Vous pouvez également définir des alarmes qui surveillent certains seuils et envoient des notifications ou prennent des mesures lorsque ces seuils sont atteints. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

> **Note**
>
> Pour démarrer rapidement avec un tableau de CloudWatch bord préconfiguré présentant les indicateurs courants pour Amazon Keyspaces, vous pouvez utiliser AWS CloudFormation

un modèle disponible sur. https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates

Rubriques

Comment utiliser les statistiques d'Amazon Keyspaces ?

Les statistiques rapportées par Amazon Keyspaces fournissent des informations que vous pouvez analyser de différentes manières. La liste suivante présente certaines utilisations courantes des métriques. Voici quelques suggestions pour vous aider à démarrer, qui ne forment pas une liste exhaustive. Pour plus d'informations sur les mesures et la rétention, consultez la section Mesures.

Comment puis-je ?	Métriques pertinentes
Comment déterminer si des erreurs système se sont produites	Vous pouvez superviser `SystemErrors` pour déterminer si des demandes ont entraîné un code erreur du serveur. En règle générale, cette métrique doit être égale à zéro. Si tel n'est pas le cas, vous devez enquêter.
Comment puis-je comparer la capacité moyenne de lecture allouée à la capacité de lecture consommée ?	Pour surveiller la capacité moyenne de lecture allouée et la capacité de lecture consommée 1. Définissez la période `ConsumedReadCapacityUnits` et `ProvisionedReadCapacityUnits` l'intervalle que vous souhaitez surveiller. 2. Modifiez la statistique pour `ConsumedReadCapacityUnits` de `Average` en `Sum`. 3. Créez une nouvelle Expression Mathvide. 4. Dans la section Détails de la nouvelle expression mathématique, entrez l'identifiant de la métrique `ConsumedReadCapacityUnits` et divisez-le par la fonction

Comment puis-je ?	Métriques pertinentes
	CloudWatch PERIOD de la métrique (metric_id/ (PERIOD (metric_id)). 5. Désélectionnez `ConsumedReadCapacityUnits` . Vous pouvez maintenant comparer votre capacité de lecture consommée moyenne à votre capacité allouée. Pour plus d'informations sur les fonctions arithmétiques de base et sur la création d'une série chronologique, voir Utilisation des mathématiques métriques.
Comment puis-je comparer l'écriture allouée moyenne à la capacité d'écriture consommée ?	Pour surveiller la capacité moyenne d'écriture allouée et la capacité d'écriture consommée 1. Définissez la période `ConsumedWriteCapacityUnits` et `ProvisionedWriteCapacityUnits` l'intervalle que vous souhaitez surveiller. 2. Modifiez la statistique pour `ConsumedWriteCapacityUnits` de `Average` en `Sum`. 3. Créez une nouvelle Expression Mathvide. 4. Dans la section Détails de la nouvelle expression mathématique, entrez l'identifiant de la métrique `ConsumedWriteCapacityUnits` et divisez-le par la fonction CloudWatch PERIOD de la métrique (metric_id/ (PERIOD (metric_id)). 5. Désélectionnez `ConsumedWriteCapacityUnits` . Vous pouvez maintenant comparer votre capacité d'écriture consommée moyenne à votre capacité allouée. Pour plus d'informations sur les fonctions arithmétiques de base et sur la création d'une série chronologique, voir Utilisation des mathématiques métriques.

Statistiques et dimensions d'Amazon Keyspaces

Lorsque vous interagissez avec Amazon Keyspaces, celui-ci envoie les statistiques et dimensions suivantes à Amazon. CloudWatch Toutes les métriques sont agrégées et signalées chaque minute. Vous pouvez utiliser les procédures suivantes pour consulter les statistiques d'Amazon Keyspaces.

Pour afficher les métriques à l'aide de la CloudWatch console

Les métriques sont d'abord regroupées par espace de noms de service, puis par les différentes combinaisons de dimension au sein de chaque espace de noms.

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Si nécessaire, changez la région. Dans la barre de navigation, choisissez la région dans laquelle se trouvent vos AWS ressources. Pour plus d'informations, consultez Points de terminaison du service AWS.
3. Dans le panneau de navigation, sélectionnez Métriques.
4. Sous l'onglet Toutes les métriques, choisissez `AWS/Cassandra`.

Pour afficher les métriques à l'aide de la AWS CLI

- À partir d'une invite de commande, utilisez la commande suivante :

```
aws cloudwatch list-metrics --namespace "AWS/Cassandra"
```

Statistiques et dimensions d'Amazon Keyspaces

Les statistiques et les dimensions qu'Amazon Keyspaces envoie à Amazon CloudWatch sont répertoriées ici.

Métriques Amazon Keyspaces

Amazon CloudWatch agrège les statistiques Amazon Keyspaces à intervalles d'une minute.

Certaines statistiques, telles que `Average` ou `Sum`, s'appliquent à chaque métrique. Cependant, toutes ces valeurs sont disponibles via la console Amazon Keyspaces, ou en utilisant la CloudWatch console ou des AWS SDK pour toutes les métriques. AWS CLI Dans le tableau suivant, chaque métrique dispose d'une liste de statistiques valides qui s'appliquent à cette métrique.

Métrique	Description
`AccountMaxTableLevelReads`	Nombre maximal d'unités de capacité en lecture pouvant être utilisées par la table d'un compte. Pour les tables à la demande, cette limite définit le nombre maximal d'unités de demande de lecture qu'une table peut utiliser. Unités `Count`: Statistiques valides : • `Maximum`— Le nombre maximum d'unités de capacité de lecture pouvant être utilisées par une table du compte.
`AccountMaxTableLevelWrites`	Nombre maximal d'unités de capacité en écriture pouvant être utilisées par la table d'un compte. Pour les tables à la demande, cette limite définit le nombre maximal d'unités de demande d'écriture qu'une table peut utiliser. Unités `Count`: Statistiques valides : • `Maximum`— Le nombre maximum d'unités de capacité d'écriture pouvant être utilisées par une table du compte.
`AccountProvisionedReadCapacityUtilization`	Pourcentage d'unités de capacité de lecture approvisionnée qu'un compte utilise. Unités : `Percent` Statistiques valides : • `Maximum` – Pourcentage maximum d'unités de capacité de lecture approvisionnée que le compte utilise. • `Minimum` – Pourcentage minimum d'unités de capacité de lecture approvisionnée que le compte utilise.

Métrique	Description
	• `Average` – Pourcentage moyen d'unités de capacité de lecture approvisionnée que le compte utilise. La métrique est publiée à intervalles de cinq minutes. Par conséquent, si vous ajustez rapidement les unités de capacité de lecture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.
`AccountProvisionedWriteCapacityUtilization`	Pourcentage d'unités de capacité d'écriture approvisionnée qu'un compte utilise. Unités : `Percent` Statistiques valides : • `Maximum` – Pourcentage maximum d'unités de capacité d'écriture approvisionnée que le compte utilise. • `Minimum` – Pourcentage minimum d'unités de capacité d'écriture approvisionnée que le compte utilise. • `Average` – Pourcentage moyen d'unités de capacité d'écriture approvisionnée que le compte utilise. La métrique est publiée à intervalles de cinq minutes. Par conséquent, si vous ajustez rapidement les unités de capacité d'écriture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.

Métrique	Description
`BillableTableSizeInBytes`	Taille facturable de la table en octets. Il s'agit de la somme de la taille codée de toutes les lignes du tableau. Cette métrique vous permet de suivre les coûts de stockage de vos tables au fil du temps. Unités : `Bytes` Dimensions : `Keyspace, TableName` Statistiques valides : • `Maximum`— La taille de stockage maximale de la table. • `Minimum`— La taille de stockage minimale de la table. • `Average`— La taille de stockage moyenne de la table. Cette métrique est calculée sur des intervalles de 4 à 6 heures.

Métrique	Description
`ConditionalCheckFailedRequests`	Nombre de demandes d'écriture de transactions légères ayant échoué. Les opérations `INSERT`, `UPDATE` et `DELETE` vous permettent de fournir une condition logique dont l'évaluation doit être true avant que l'opération puisse continuer. Si cette condition est fausse, elle `ConditionalCheckFailedRequests` est incrémentée d'une unité. Les contrôles de condition qui évaluent de manière erronée consomment des unités de capacité d'écriture en fonction de la taille de la ligne. Pour plus d'informations, consultez the section called "Transactions légères". Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Minimum` • `Maximum` • `Average` • `SampleCount` • `Sum`

Métrique	Description
`ConsumedReadCapacityUnits`	Nombre d'unités de capacité de lecture consommées au cours de la période spécifiée. Pour plus d'informations, consultez la section Mode de capacité de lecture/écriture. **Note** Pour comprendre votre utilisation moyenne du débit par seconde, utilisez la statistique `Sum` pour calculer le débit consommé pour la période d'une minute. Puis divisez la somme par le nombre de secondes en une minute (60) pour calculer la `ConsumedReadCapacityUnits` moyenne par seconde (en sachant que cette moyenne ne mettra pas en évidence les grands pics d'activité de lecture). Pour plus d'informations sur la comparaison de la capacité de lecture moyenne consommée à la capacité de lecture allouée, voir the section called “Utilisation de métriques ” Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Minimum`— Le nombre minimal d'unités de capacité de lecture consommées par chaque demande individuelle envoyée à la table. • `Maximum`— Le nombre maximal d'unités de capacité de lecture consommées par une requête individuelle envoyée à la table. • `Average` – Capacité de lecture moyenne par demande consommée.

Métrique	Description
	Note La valeur `Average` est influencée par des périodes d'inactivité où la valeur de l'échantillon est nulle. • `Sum` – Nombre total d'unités de capacité de lecture consommées. Il s'agit de la statistique la plus utile pour la métrique `ConsumedReadCapacityUnits` . • `SampleCount` — Le nombre de demandes adressées à Amazon Keyspaces, même si aucune capacité de lecture n'a été consommée. **Note** La valeur `SampleCount` est influencée par des périodes d'inactivité où la valeur de l'échantillon est nulle.

Métrique	Description
`ConsumedWriteCapacityUnits`	Nombre d'unités de capacité d'écriture consommées au cours de la période spécifiée. Vous pouvez récupérer la capacité d'écriture totale consommée pour une table. Pour plus d'informations, consultez la section Mode de capacité de lecture/écriture. **Note** Pour comprendre votre utilisation moyenne du débit par seconde, utilisez la statistique `Sum` pour calculer le débit consommé pour la période d'une minute. Puis divisez la somme par le nombre de secondes en une minute (60) pour calculer la `ConsumedWriteCapacityUnits` moyenne par seconde (en sachant que cette moyenne ne mettra pas en évidence les grands pics d'activité d'écriture). Pour plus d'informations sur la comparaison de la capacité d'écriture moyenne consommée à la capacité d'écriture allouée, voir the section called “Utilisation de métriques ” Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Minimum`— Le nombre minimal d'unités de capacité d'écriture consommées par chaque demande individuelle envoyée à la table. • `Maximum`— Le nombre maximal d'unités de capacité d'écriture consommées par une requête individuelle envoyée à la table.

Métrique	Description
	• `Average` – Capacité d'écriture moyenne par demande consommée. **Note** La valeur `Average` est influencée par des périodes d'inactivité où la valeur de l'échantillon est nulle. • `Sum` – Nombre total d'unités de capacité d'écriture consommées. Il s'agit de la statistique la plus utile pour la métrique `ConsumedWriteCapacityUnits` . • `SampleCount` — Le nombre de demandes adressées à Amazon Keyspaces, même si aucune capacité d'écriture n'a été consommée. **Note** La valeur `SampleCount` est influencée par des périodes d'inactivité où la valeur de l'échantillon est nulle.

Métrique	Description
`MaxProvisionedTabl eReadCapacityUtilization`	Pourcentage moyen d'unités de capacité en lecture allouées utilisées par la table en lecture allouée la plus élevée du compte. Unités `Percent`: Statistiques valides : • `Maximum` – Pourcentage maximum d'unités de capacité de lecture approvisionnée qu'utilise la table de lecture la plus approvisionnée du compte. • `Minimum` – Pourcentage minimum d'unités de capacité de lecture approvisionnée qu'utilise la table de lecture la plus approvisionnée du compte. • `Average` – Pourcentage moyen d'unités de capacité de lecture approvisionnée qu'utilise la table de lecture la plus approvisionnée du compte. La métrique est publiée à intervalles de cinq minutes. Par conséquen t, si vous ajustez rapidement les unités de capacité de lecture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.

Métrique	Description
`MaxProvisionedTableWriteCapacityUtilization`	Pourcentage de capacité en écriture allouée utilisée par la table en écriture allouée la plus élevée d'un compte. Unités `Percent`: Statistiques valides : • `Maximum`— Le pourcentage maximal d'unités de capacité d'écriture provisionnées utilisées par la table d'écriture provisionnée la plus élevée du compte. • `Minimum`— Le pourcentage minimum d'unités de capacité d'écriture provisionnées utilisées par la table d'écriture provisionnée la plus élevée du compte. • `Average`— Le pourcentage moyen d'unités de capacité d'écriture provisionnées utilisées par la table d'écriture provisionnée la plus élevée du compte. La métrique est publiée à intervalles de cinq minutes. Par conséquent, si vous ajustez rapidement les unités de capacité d'écriture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.

Métrique	Description
`PerConnectionRequestRateExceeded`	Demandes adressées à Amazon Keyspaces qui dépassent le quota de demandes par connexion. Chaque connexion client à Amazon Keyspaces peut prendre en charge jusqu'à 3 000 demandes CQL par seconde. Les clients peuvent créer plusieurs connexions pour augmenter le débit. Lorsque vous utilisez la réplication multirégionale, chaque écriture répliquée contribue également à ce quota. Il est recommandé d'augmenter le nombre de connexions à vos tables pour éviter les `PerConnectionRequestRateExceeded` erreurs. Il n'y a aucune limite au nombre de connexions que vous pouvez avoir sur Amazon Keyspaces. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `SampleCount` • `Sum`

Métrique	Description
`ProvisionedReadCapacityUnits`	Nombre d'unités de capacité de lecture allouées pour une table. La dimension `TableName` renvoie la valeur `ProvisionedReadCapacityUnits` de la table. Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Minimum` – Paramètre le plus bas pour la capacité de lecture approvisionnée. Si vous utilisez `ALTER TABLE` pour augmenter la capacité de lecture, cette métrique indique la valeur `ReadCapacityUnits` approvisionnée la plus faible pendant cette période. • `Maximum` – Paramètre le plus élevé pour la capacité de lecture approvisionnée. Si vous utilisez `ALTER TABLE` pour réduire la capacité de lecture, cette métrique indique la valeur `ReadCapacityUnits` approvisionnée la plus élevée pendant cette période. • `Average` – Capacité de lecture approvisionnée moyenne. La métrique `ProvisionedReadCapacityUnits` est publiée à intervalles de cinq minutes. Par conséquent, si vous ajustez rapidement les unités de capacité de lecture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.

Métrique	Description
`ProvisionedWriteCapacityUnits`	Nombre d'unités de capacité d'écriture allouées pour une table. La dimension `TableName` renvoie la valeur `ProvisionedWriteCapacityUnits` de la table. Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Minimum` – Paramètre le plus bas pour la capacité d'écriture approvisionnée. Si vous utilisez `ALTER TABLE` pour augmenter la capacité d'écriture, cette métrique indique la valeur `WriteCapacityUnits` approvisionnée la plus faible pendant cette période. • `Maximum` – Paramètre le plus élevé pour la capacité d'écriture approvisionnée. Si vous utilisez `ALTER TABLE` pour réduire la capacité d'écriture, cette métrique indique la valeur `WriteCapacityUnits` approvisionnée la plus élevée pendant cette période. • `Average` – Capacité d'écriture approvisionnée moyenne. La métrique `ProvisionedWriteCapacityUnits` est publiée à intervalles de cinq minutes. Par conséquent, si vous ajustez rapidement les unités de capacité d'écriture approvisionnée, il se peut que cette statistique ne reflète pas la moyenne réelle.

Métrique	Description
`ReadThrottleEvents`	Demandes adressées à Amazon Keyspaces qui dépassent la capacité de lecture allouée pour une table, ou les quotas au niveau du compte, les quotas de demande par connexion ou les quotas au niveau de la partition. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `SampleCount` • `Sum`
`ReplicationLatency`	Cette métrique s'applique uniquement aux espaces clés multirégionaux et mesure le temps nécessaire à la réplication `updatesinserts`, ou le temps nécessaire pour passer d'une table `deletes` de réplique à une autre table de réplication dans un espace de saisie multirégi onal. Unités : `Millisecond` Dimensions : `TableName, ReceivingRegion` Statistiques valides : • `Average` • `Maximum` • `Minimum`

Métrique	Description
`ReturnedItemCountBySelect`	Nombre de lignes renvoyées par des `SELECT` requêtes à plusieurs lignes au cours de la période spécifiée. Les `SELECT` requêtes à plusieurs lignes sont des requêtes qui ne contiennent pas la clé primaire complète, telles que les scans complets de tables et les requêtes par plage. Notez que le nombre d'éléments retournés n'est pas nécessairement le même que le nombre d'éléments ayant été évalués. Par exemple, supposons que vous ayez demandé une sélection `SELECT *` avec une clause `ALLOW FILTERING` sur une table qui comportait 100 lignes, mais que vous ayez spécifié une clause `WHERE` afin de réduire les résultats à 15 lignes. Dans ce cas, la réponse de `SELECT` contiendra un `ScanCount` de 100 et un `Count` de 15 lignes retournées. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `Minimum` • `Maximum` • `Average` • `SampleCount` • `Sum`

Métrique	Description
`StoragePartitionThroughputCapacityExceeded`	Demandes adressées à une partition de stockage Amazon Keyspaces qui dépassent la capacité de débit de la partition. Les partitions de stockage Amazon Keyspaces peuvent prendre en charge jusqu'à 1 000 WCU/WRU par seconde et 3 000 RCU/RRU par seconde. Nous vous recommandons de revoir votre modèle de données afin de répartir le trafic de lecture/écriture sur un plus grand nombre de partitions afin de limiter ces exceptions. **Note** Les partitions logiques d'Amazon Keyspaces peuvent s'étendre sur plusieurs partitions de stockage et leur taille est pratiquement illimitée. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `SampleCount` • `Sum`
`SuccessfulRequestCount`	Le nombre de demandes traitées avec succès au cours de la période spécifiée. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `SampleCount`

Métrique	Description
SuccessfulRequestLatency	Les demandes envoyées à Amazon Keyspaces avec succès au cours de la période spécifiée. SuccessfulRequestLatency peut fournir deux types d'informations différents : • Temps écoulé pour les demandes réussies (Minimum, Maximum, Sum ou Average). • Nombre de requêtes réussies (SampleCount). SuccessfulRequestLatency reflète uniquement l'activité au sein d'Amazon Keyspaces et ne prend pas en compte la latence du réseau ni l'activité côté client. Unités : Milliseconds Dimensions : Keyspace, TableName, Operation Statistiques valides : • Minimum • Maximum • Average • SampleCount

Métrique	Description
SystemErrors	Les demandes adressées à Amazon Keyspaces qui génèrent un ServerError pendant la période spécifiée . Une erreur ServerError indique généralement une erreur de service interne. Unités : Count Dimensions : Keyspace, TableName, Operation Statistiques valides : • Sum • SampleCount
SystemReconciliationDeletes	Les unités consommées pour supprimer les données tombstone lorsque les horodatages côté client sont activés. Chacune SystemReconciliationDelete fournit une capacité suffisante pour supprimer ou mettre à jour jusqu'à 1 Ko de données par ligne. Par exemple, pour mettre à jour une ligne qui stocke 2,5 Ko de données et pour supprimer une ou plusieurs colonnes de la ligne en même temps, il en faut 3SystemReconciliationDeletes . Ou bien, pour supprimer une ligne entière contenant 3,5 Ko de données, 4 sont nécessaires SystemReconciliationDeletes . Unités : Count Dimensions : Keyspace, TableName Statistiques valides : • Sum— Le nombre total de produits SystemReconciliationDeletes consommés au cours d'une période donnée.

Métrique	Description
`TTLDeletes`	Unités consommées pour supprimer ou mettre à jour des données d'affilée à l'aide de Time to Live (TTL). Chacune `TTLDelete` fournit une capacité suffisante pour supprimer ou mettre à jour jusqu'à 1 Ko de données par ligne. Par exemple, pour mettre à jour une ligne qui stocke 2,5 Ko de données et pour supprimer une ou plusieurs colonnes de la ligne en même temps, 3 suppressions TTL sont nécessaires. Ou bien, pour supprimer une ligne entière contenant 3,5 Ko de données, 4 suppressions TTL sont nécessaires. Unités : `Count` Dimensions : `Keyspace, TableName` Statistiques valides : • `Sum`— Le nombre total de produits `TTLDeletes` consommés au cours d'une période donnée.

Métrique	Description
`UserErrors`	Demandes adressées à Amazon Keyspaces qui génèrent une `InvalidRequest` erreur au cours de la période spécifiée. Une erreur `InvalidRequest` indique généralement une erreur côté client, comme une combinaison de paramètres non valide, la tentative de mise à jour d'une table inexistante ou une signature de demande incorrecte. `UserErrors` représente l'ensemble des demandes non valides pour le courant Région AWS et le courant Compte AWS. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `Sum` • `SampleCount`
`WriteThrottleEvents`	Demandes adressées à Amazon Keyspaces qui dépassent la capacité d'écriture allouée pour une table, ou les quotas au niveau du compte, les quotas de demande par connexion ou les quotas au niveau de la partition. Unités : `Count` Dimensions : `Keyspace, TableName, Operation` Statistiques valides : • `SampleCount` • `Sum`

Dimensions des statistiques Amazon Keyspaces

Les statistiques d'Amazon Keyspaces sont qualifiées par les valeurs du compte, du nom de la table ou de l'opération. Vous pouvez utiliser la CloudWatch console pour récupérer les données Amazon Keyspaces selon l'une des dimensions indiquées dans le tableau suivant.

Dimension	Description
`Keyspace`	Cette dimension limite les données à un keyspace spécifique. Cette valeur peut être n'importe quel espace-clé de la région actuelle et de la région actuelle Compte AWS.
`Operation`	Cette dimension limite les données à l'une des opérations CQL d'Amazon Keyspaces, telle que `INSERT` les opérations. `SELECT`
`TableName`	Cette dimension limite les données à une table spécifique. Cette valeur peut être n'importe quel nom de table de la région actuelle et de la région en cours Compte AWS. Si le nom de la table n'est pas unique dans le compte, vous devez également spécifier `Keyspace`.

Création d' CloudWatch alarmes pour surveiller Amazon Keyspaces

Vous pouvez créer une CloudWatch alarme Amazon pour Amazon Keyspaces qui envoie un message Amazon Simple Notification Service (Amazon SNS) lorsque l'alarme change d'état. Une alarme surveille une seule métrique pendant la période que vous spécifiez. Elle réalise une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. L'action est une notification envoyée à une rubrique Amazon SNS ou à une politique d'Application Auto Scaling.

Lorsque vous utilisez Amazon Keyspaces en mode provisionné avec Application Auto Scaling, le service crée deux paires d' CloudWatch alarmes en votre nom. Chaque paire représente vos limites supérieure et inférieure pour les paramètres de débit alloué. Ces CloudWatch alarmes sont déclenchées lorsque l'utilisation réelle de la table s'écarte de votre utilisation cible pendant une période prolongée. Pour en savoir plus sur les CloudWatch alarmes créées par Application Auto Scaling, consultezthe section called “Comment fonctionne le dimensionnement automatique d'Amazon Keyspaces”.

Les alarmes déclenchent des actions uniquement pour les changements d'état prolongés. CloudWatch les alarmes n'appellent pas d'actions simplement parce qu'elles sont dans un état particulier. L'état doit avoir changé et avoir été maintenu pendant un nombre de périodes spécifié.

Pour plus d'informations sur la création d' CloudWatch alarmes, consultez la section Utilisation des CloudWatch alarmes Amazon dans le guide de CloudWatch l'utilisateur Amazon.

Journalisation des appels d'API Amazon Keyspaces avec AWS CloudTrail

Amazon Keyspaces est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions effectuées par un utilisateur, un rôle ou un AWS service dans Amazon Keyspaces. CloudTrail capture les appels d'API DDL (Data Definition Language) et les appels d'API DML (Data Manipulation Language) pour Amazon Keyspaces sous forme d'événements. Les appels capturés incluent les appels provenant de la console Amazon Keyspaces et les appels programmatiques vers les opérations de l'API Amazon Keyspaces.

Si vous créez un suivi, vous pouvez activer la diffusion continue des CloudTrail événements vers un bucket Amazon Simple Storage Service (Amazon S3), y compris les événements pour Amazon Keyspaces.

Si vous ne configurez pas de suivi, vous pouvez toujours consulter les derniers événements pris en charge sur la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été faite à Amazon Keyspaces, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite, ainsi que des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Rubriques

- Configuration des entrées du fichier journal Amazon Keyspaces dans CloudTrail
- Informations sur le langage de définition des données (DDL) Amazon Keyspaces dans CloudTrail
- Informations sur le langage de manipulation des données (DML) d'Amazon Keyspaces dans CloudTrail
- Comprendre les entrées des fichiers journaux Amazon Keyspaces

Configuration des entrées du fichier journal Amazon Keyspaces dans CloudTrail

Chaque action d'API Amazon Keyspaces connectée CloudTrail inclut des paramètres de demande exprimés dans le langage de requête CQL. Pour plus d'informations, consultez le Référence du langage CQL.

Vous pouvez afficher, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements.

Pour un enregistrement continu des événements survenus dans votre environnement Compte AWS, y compris des événements pour Amazon Keyspaces, créez un historique. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un parcours dans la console, celui-ci s'applique à toutes les AWS régions. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence.

Pour plus d'informations, consultez les rubriques suivantes dans le AWS CloudTrail Guide de l'utilisateur :

- Présentation de la création d'un journal d'activité
- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux provenant de plusieurs régions
- Réception de fichiers CloudTrail journaux provenant de plusieurs comptes

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été faite avec les informations d'identification de l'utilisateur root ou AWS Identity and Access Management (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez l'élément CloudTrail UserIdentity.

Informations sur le langage de définition des données (DDL) Amazon Keyspaces dans CloudTrail

CloudTrail est activé sur votre compte Compte AWS lorsque vous créez le compte. Lorsqu'une activité DDL se produit dans Amazon Keyspaces, cette activité est automatiquement enregistrée en CloudTrail tant qu'événement avec les AWS autres événements de service dans l'historique des événements. Le tableau suivant présente les instructions DDL enregistrées pour Amazon Keyspaces.

CloudTrail **`eventName`**	Instruction	Action CQL	AWS Action du SDK
CreateKeyspace	DDL	`CREATE KEYSPACE`	`CreateKeyspace`
DropKeyspace	DDL	`DROP KEYSPACE`	`DeleteKeyspace`
CreateTable	DDL	`CREATE TABLE`	`CreateTable`
DropTable	DDL	`DROP TABLE`	`DeleteTable`
AlterTable	DDL	`ALTER TABLE`	`UpdateTable` , `TagResource` , `UntagResource`

Informations sur le langage de manipulation des données (DML) d'Amazon Keyspaces dans CloudTrail

Pour activer la journalisation des instructions DML d'Amazon Keyspaces avec CloudTrail, vous devez d'abord activer la journalisation de l'activité de l'API du plan de données dans. CloudTrail Vous pouvez commencer à enregistrer les événements DML Amazon Keyspaces dans des pistes nouvelles ou existantes en choisissant de consigner l'activité pour le type d'événement de données (table Cassandra) à l'aide de la CloudTrail console, ou en définissant la `resources.type` valeur à l'aide de `AWS::Cassandra::Table` la AWS CLI ou des opérations d'API. CloudTrail Pour plus d'informations, veuillez consulter Journalisation des événements de données.

Le tableau suivant indique les événements de données enregistrés CloudTrail par`Cassandra table`.

CloudTrail **eventName**	Instruction	Action CQL	AWS Action du SDK
Select	DML	`SELECT`	`GetKeyspace` , `GetTable`, `ListKeyspaces` , `ListTables` `ListTagsForResource`
Insert	DML	`INSERT`	aucune action du AWS SDK disponible
Mettre à jour	DML	`UPDATE`	aucune action du AWS SDK disponible
Suppression	DML	`DELETE`	aucune action du AWS SDK disponible

Comprendre les entrées des fichiers journaux Amazon Keyspaces

CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre les `DropTable` actions `CreateKeyspace` `DropKeyspaceCreateTable`,, et :

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
```

```
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T18:53:04Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "CreateKeyspace",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "\n\tCREATE KEYSPACE \"mykeyspace\"\n\tWITH\n\t\tREPLICATION =
{'class': 'SingleRegionStrategy'}\n\t\t",
        "keyspaceName": "mykeyspace"
      },
      "responseElements": null,
      "requestID": "bfa3e75d-bf4d-4fc0-be5e-89d15850eb41",
      "eventID": "d25beae8-f611-4229-877a-921557a07bb9",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Keyspace",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
```

```
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    },
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T19:28:39Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "DropKeyspace",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "DROP KEYSPACE \"mykeyspace\"",
        "keyspaceName": "mykeyspace"
      },
      "responseElements": null,
      "requestID": "66f3d86a-56ae-4c29-b46f-abcd489ed86b",
      "eventID": "e5aebeac-e1dd-41e3-a515-84fe6aaabd7b",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Keyspace",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
```

```
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
          "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    },
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T18:55:24Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "CreateTable",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "\n\tCREATE TABLE \"mykeyspace\".\"mytable\"(\n\t\t\"ID\" int,
\n\t\t\"username\" text,\n\t\t\"email\" text,\n\t\t\"post_type\" text,\n\t\tPRIMARY
 KEY((\"ID\", \"username\", \"email\")))",
```

```
        "keyspaceName": "mykeyspace",
        "tableName": "mytable"
      },
      "responseElements": null,
      "requestID": "5f845963-70ea-4988-8a7a-2e66d061aacb",
      "eventID": "fe0dbd2b-7b34-4675-a30c-740f9d8d73f9",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable"
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
          "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    },
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
```

```
          }
        }
      },
      "eventTime": "2020-01-15T19:27:59Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "DropTable",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "DROP TABLE \"mykeyspace\".\"mytable\"",
        "keyspaceName": "mykeyspace",
        "tableName": "mytable"
      },
      "responseElements": null,
      "requestID": "025501b0-3582-437e-9d18-8939e9ef262f",
      "eventID": "1a5cbedc-4e38-4889-8475-3eab98de0ffd",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable"
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
          "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
  ]
}
```


Le fichier journal suivant montre un exemple d'SELECTinstruction.


```
{
    "eventVersion": "1.09",
```

```
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
        "accountId": "111122223333",
        "userName": "alice"
    },
    "eventTime": "2023-11-17T10:38:04Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Select",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
          "tableName": "my_table",
          "conditions": [
              "pk = **(Redacted)",
              "ck < 3**(Redacted)0",
              "region = 't**(Redacted)t'"
          ],
          "select": [
              "pk",
              "ck",
              "region"
          ],
          "allowFiltering": true
    },
    "responseElements": null,
    "requestID": "6d83bbf0-a3d0-4d49-b1d9-e31779a28628",
    "eventID": "e00552d3-34e9-4092-931a-912c4e08ba17",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::Cassandra::Table",
            "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table"
        }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
```

```
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Le fichier journal suivant montre un exemple d'`INSERT`instruction.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
        "accountId": "111122223333",
        "userName": "alice"
    },
    "eventTime": "2023-12-01T22:11:43Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Insert",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
          "tableName": "my_table",
          "primaryKeys": {
              "pk": "**(Redacted)",
              "ck": "1**(Redacted)8"
          },
          "columnNames": [
              "pk",
              "ck",
              "region"
          ],
          "updateParameters": {
              "TTL": "2**(Redacted)0"
          }
      }
    },
    "responseElements": null,
```

```
    "requestID": "edf8af47-2f87-4432-864d-a960ac35e471",
    "eventID": "81b56a1c-9bdd-4c92-bb8e-92776b5a3bf1",
    "readOnly": false,
    "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
        }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Le fichier journal suivant montre un exemple d'UPDATEinstruction.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
          "type": "IAMUser",
          "principalId": "AKIAIOSFODNN7EXAMPLE",
          "arn": "arn:aws:iam::111122223333:user/alice",
          "accountId": "111122223333",
          "userName": "alice"
    },
    "eventTime": "2023-12-01T22:11:43Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Update",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
          "tableName": "my_table",
```

```
            "primaryKeys": {
                "pk": "'t**(Redacted)t'",
                "ck": "'s**(Redacted)g'"
            },
            "assignmentColumnNames": [
               "nonkey"
            ],
            "conditions": [
                "nonkey < 1**(Redacted)7"
            ]
        },
        "responseElements": null,
        "requestID": "edf8af47-2f87-4432-864d-a960ac35e471",
        "eventID": "81b56a1c-9bdd-4c92-bb8e-92776b5a3bf1",
        "readOnly": false,
        "resources": [
            {
              "accountId": "111122223333",
              "type": "AWS::Cassandra::Table",
              "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
            }
        ],
        "eventType": "AwsApiCall",
        "apiVersion": "3.4.4",
        "managementEvent": false,
        "recipientAccountId": "111122223333",
        "eventCategory": "Data",
        "tlsDetails": {
              "tlsVersion": "TLSv1.3",
              "cipherSuite": "TLS_AES_128_GCM_SHA256",
              "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
        }
}
```

Le fichier journal suivant montre un exemple d'`DELETE`instruction.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
```

```
        "accountId": "111122223333",
        "userName": "alice",
    },
    "eventTime": "2023-10-23T13:59:05Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Delete",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
        "keyspaceName": "my_keyspace",
        "tableName": "my_table",
        "primaryKeys": {
            "pk": "**(Redacted)",
            "ck": "**(Redacted)"
        },
        "conditions": [],
        "deleteColumnNames": [
            "m",
            "s"
        ],
        "updateParameters": {}
    },
    "responseElements": null,
    "requestID": "3d45e63b-c0c8-48e2-bc64-31afc5b4f49d",
    "eventID": "499da055-c642-4762-8775-d91757f06512",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::Cassandra::Table",
        "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
      }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
```

```
    }
}
```

Sécurité dans Amazon Keyspaces (pour Apache Cassandra)

Chez AWS, la sécurité dans le cloud est notre priorité numéro 1. En tant que client AWS, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité.

La sécurité est une responsabilité partagée entre AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est responsable de la protection de l'infrastructure qui exécute des services AWS dans le cloud AWS Cloud. AWS vous fournit également les services que vous pouvez utiliser en toute sécurité. L'efficacité de notre sécurité est régulièrement testée et vérifiée par des auditeurs tiers dans le cadre des programmes de conformité AWS. Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon Keyspaces, consultez la section AWSServices concernés par programme de conformité.
- Sécurité dans le cloud : votre responsabilité est déterminée par le service AWS que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris la sensibilité de vos données, les exigences de votre organisation, et la législation et la réglementation applicables.

Cette documentation vous aidera à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amazon Keyspaces. Les rubriques suivantes expliquent comment configurer Amazon Keyspaces pour répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui peuvent vous aider à surveiller et à sécuriser vos ressources Amazon Keyspaces.

Rubriques

Protection des données dans Amazon Keyspaces

Le modèle de responsabilité AWS partagée s'applique à la protection des données dans Amazon Keyspaces (pour Apache Cassandra). Comme décrit dans ce modèle, AWS est responsable de la protection de l'infrastructure globale sur laquelle l'ensemble d'AWS Cloud s'exécute. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité pour les Services AWS que vous utilisez. Pour en savoir plus sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWSet RGPD (Règlement général sur la protection des données) sur le AWSBlog de sécurité.

À des fins de protection des données, nous vous recommandons de protéger les informations d'identification Compte AWS et de configurer les comptes utilisateur individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez les certificats SSL/TLS pour communiquer avec les ressources AWS. Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez une API (Interface de programmation) et le journal de l'activité des utilisateurs avec AWS CloudTrail.
- Utilisez des solutions de chiffrement AWS, ainsi que tous les contrôles de sécurité par défaut au sein des Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés FIPS (Federal Information Processing Standard) 140-2 lorsque vous accédez à AWS via une CLI (Interface de ligne de commande) ou une API (Interface de programmation), utilisez un point de terminaison FIPS (Federal Information Processing Standard). Pour en savoir plus sur les points de terminaison FIPS (Federal Information Processing Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels

que le champ Name (Nom). Cela inclut lorsque vous travaillez avec Amazon Keyspaces ou d'autres utilisateurs à Services AWS l'aide de la console, de l'API ou AWS des AWS CLI SDK. Toutes les données que vous saisissez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Rubriques

Chiffau repos dans Amazon Keyspaces

Le chiffrement Amazon Keyspaces (pour Apache Cassandra) offre une sécurité renforcée en chiffffau repos à l'aide de clés de chiffrement dans AWS Key Management Service(AWS KMS). Cette fonctionnalité réduit la lourdeur opérationnelle et la complexité induites par la protection des données sensibles. Avec le chiffrement au repos, vous pouvez créer des applications sensibles à la sécurité qui répondent à des exigences strictes en matière de conformité et de réglementation en matière de protection des données.

Le chiffrement Amazon Keyspaces au repos chiffre vos données à l'aide du chiffrement Advanced Encryption Standard 256 bits (AES-256). Cela permet de sécuriser vos données contre tout accès non autorisé au stockage sous-jacent.

Amazon Keyspaces chiffre et déchiffre les données de table de manière transparente. Amazon Keyspaces utilise le chiffrement des enveloppes et une hiérarchie de clés pour protéger les clés de chiffrement des données. Il s'intègre àAWS KMS la fonction de stockage et de gestion de la clé de chiffrement racine. Pour plus d'informations sur la hiérarchie des clés de chiffrement, consultezthe section called “Comment ça marche”. Pour plus d'informations surAWS KMS des concepts tels que le chiffrement des enveloppes, consultez les conceptsAWS KMS des services de gestion dans le Guide duAWS Key Management Service développeur.

Lorsque vous créez une table, vous pouvez choisir l'une des AWS KMSclés suivantes (clés KMS) :

- Clé détenue par AWS— Il s'agit du type de cryptage par défaut. La clé est détenue par Amazon Keyspaces (sans frais supplémentaires).

- Clé gérée par le client — Cette clé est créée, détenue et gérée et détenue et gérée par vous, et créée, détenue et gérée par vous, et détenue et gérée par vous, Vous disposez d'un contrôle total sur la clé gérée par le client (AWS KMSdes frais s'appliquent).

Vous pouvez basculer à tout moment entre la cléClé détenue par AWS et la clé gérée par le client. Vous pouvez spécifier une clé gérée par le client lorsque vous créez une table ou changez la clé KMS d'une table existante en utilisant la console ou par programme à l'aide d'instructions CQL. Pour savoir comment procéder, veuillez consulter la section Chiffrement au repos : comment utiliser des clés gérées par le client pour chiffrer des tables dans Amazon Keyspaces.

Le chiffrement au repos à l'aide de l'option par défaut deClés détenues par AWS est fourni sans frais supplémentaires. Toutefois, desAWS KMS frais s'appliquent pour les clés gérées par le client. Pour de plus amples informations sur la tarification, veuillez consulter Tarification AWS KMS.

Le chiffrement Keyspaces Keyrepos est disponible dans toutes les régionsRégions AWS, y compris laAWS Chine (Beijing) etAWS la Chine (Ningxia). Pour plus d'informations, veuillez consulter Fonctionnement du chiffrement au repos dans Amazon Keyspaces.

Rubriques

- Fonctionnement du chiffrement au repos dans Amazon Keyspaces
- Chiffrement au repos : comment utiliser des clés gérées par le client pour chiffrer des tables dans Amazon Keyspaces

Fonctionnement du chiffrement au repos dans Amazon Keyspaces

Le chiffrement Amazon Keyspaces (pour Apache Cassandra) au repos chiffre vos données à l'aide de la norme Advanced Encryption Standard 256 bits (AES-256). Cela permet de sécuriser vos données contre tout accès non autorisé au stockage sous-jacent. Toutes les données clients figurant dans les tables Amazon Keyspaces sont cryptées au repos par défaut, et le chiffrement côté serveur est transparent, ce qui signifie qu'aucune modification des applications n'est requise.

Le chiffrement au repos s'intègre avec AWS Key Management Service (AWS KMS) pour la gestion de la clé de chiffrement utilisée pour chiffrer vos tables. Lorsque vous créez une table ou mettez à jour une table existante, vous pouvez choisir l'une des options AWS KMSclés suivantes :

- Clé détenue par AWS— Il s'agit du type de cryptage par défaut. La clé est détenue par Amazon Keyspaces (sans frais supplémentaires).

- Clé gérée par le client — Cette clé est créée, détenue et gérée par vous, et stockée dans votre compte. Vous disposez d'un contrôle total sur la clé gérée par le client (AWS KMSdes frais s'appliquent).

AWS KMSclé (clé KMS)

Le chiffrement au repos protège toutes vos données Amazon Keyspaces à l'aide d'uneAWS KMS clé. Par défaut, Amazon Keyspaces utilise une Clé détenue par AWS, c'est-à-dire une clé de chiffrement à locataires multiples créée et gérée dans un compte de service Amazon Keyspaces.

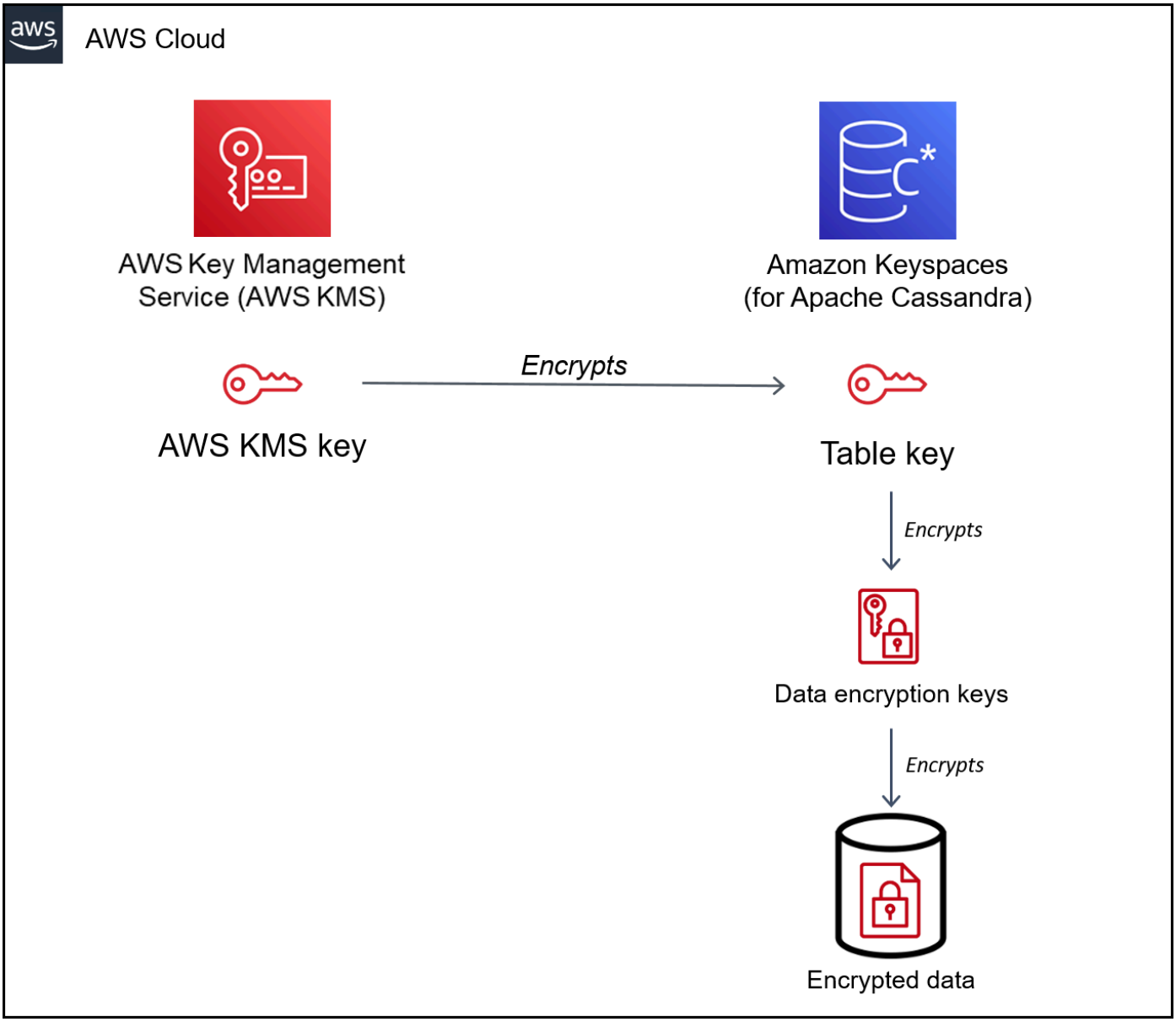
Vous pouvez toutefois chiffrer vos tables Amazon Keyspaces à l'aide d'une clé gérée par le client dans votreCompte AWS. Vous pouvez sélectionner une autre clé KMS pour chaque table dans un espace de clés. La clé KMS que vous sélectionnez pour une table est également utilisée pour chiffrer toutes ses métadonnées et les sauvegardes restaurables.

Vous sélectionnez la clé KMS pour une table lorsque vous créez ou mettez à jour la table. Vous pouvez modifier la clé KMS d'une table à tout moment, soit dans la console Amazon Keyspaces, soit en exécutant l'instruction ALTER TABLE. Le processus de changement de clé KMS est fluide et ne nécessite pas de temps d'arrêt ni de dégradation du service.

Hiérarchie de clés

Amazon Keyspaces utilise une hiérarchie de clés pour chiffrer les données. Dans cette hiérarchie de clés, la clé KMS est la clé racine. Il est utilisé pour chiffrer et déchiffrer la clé de chiffrement de table Amazon Keyspaces. La clé de chiffrement de table est utilisée pour crypter les clés de chiffrement utilisées en interne par Amazon Keyspaces pour chiffrer et déchiffrer les données lors des opérations de lecture et d'écriture.

Grâce à la hiérarchie des clés de chiffrement, vous pouvez modifier la clé KMS sans avoir à rechiffrer les données ni affecter les applications et les opérations de données en cours.



Clé de table

La clé de table Amazon Keyspaces est utilisée comme clé de chiffrement de clé. Amazon Keyspaces utilise la clé de table pour protéger les clés de chiffrement des données internes utilisées pour chiffrer les données stockées dans les tables, les fichiers journaux et les sauvegardes restaurables. Amazon Keyspaces génère une clé de chiffrement des données unique pour chaque structure sous-jacente dans une table. Toutefois, plusieurs lignes de table peuvent être protégées par la même clé de chiffrement des données.

Lorsque vous définissez la clé KMS pour la première fois sur une clé gérée par le client, une clé de données estAWS KMS générée. La clé deAWS KMS données fait référence à la clé de table dans Amazon Keyspaces.

Lorsque vous accédez à une table chiffrée, Amazon Keyspaces envoie une demande àAWS KMS afin de pouvoir utiliser la clé KMS pour déchiffrer la clé de table. Il utilise ensuite la clé de table en texte brut pour déchiffrer les clés de chiffrement des données Amazon Keyspaces, et il utilise les clés de chiffrement de données en texte brut pour déchiffrer les données de la table.

Amazon Keyspaces utilise et stocke la clé de table et les clés de chiffrement des données en dehors deAWS KMS. Il protège toutes les clés avec les clés de chiffrement Advanced Encryption Standard (AES) et 256 bits. Ensuite, il stocke les clés cryptées avec les données cryptées afin qu'elles soient disponibles pour déchiffrer les données de la table à la demande.

Mise en cache des clés de table

Pour éviter d'appelerAWS KMS pour chaque opération Amazon Keyspaces, Amazon Keyspaces met en cache les clés de table en texte brut pour chaque connexion en mémoire. Si Amazon Keyspaces obtient une demande pour la clé de table mise en cache après cinq minutes d'inactivité, il envoie une nouvelle demandeAWS KMS à pour déchiffrer la clé de table. Cet appel capture les modifications apportées aux politiques d'accès de la clé KMS dansAWS KMS ouAWS Identity and Access Management (IAM) depuis la dernière demande de déchiffrement de la clé de table.

Chiffrement d'enveloppe

Si vous modifiez la clé gérée par le client pour votre table, Amazon Keyspaces génère une nouvelle clé de table. Il utilise ensuite la nouvelle clé de table pour rechiffrer les clés de chiffrement des données. Il utilise également la nouvelle clé de table pour crypter les clés de table précédentes utilisées pour protéger les sauvegardes restaurables. Ce processus est appelé chiffrement d'enveloppe. Cela garantit que vous pouvez accéder à des sauvegardes restaurables même si vous effectuez une rotation de la clé gérée par le client. Pour plus d'informations sur le chiffrement d'enveloppe, consultez Chiffrement d'enveloppe dans le Guide duAWS Key Management Service développeur.

Rubriques

- AWSclés possédées
- Clés gérées par le client
- Notes d'utilisation du chiffrement au repos

AWSclés possédées

Clés détenues par AWSne sont pas stockés dans votreCompte AWS. Elles font partie d'une collection de clés KMS qu'AWSpossède et gère pour une utilisation dans plusieurs comptesComptes AWS. AWSles services peuvent être utilisésClés détenues par AWS pour protéger vos données.

Vous ne pouvez pas afficher, gérer ou utiliser lesClés détenues par AWS clés, ou vérifier leur utilisation. Toutefois, vous n'avez pas besoin de travailler ou de modifier les programmes pour protéger les clés qui chiffrent vos données.

Les clés n'occasionnent aucuns frais mensuels ou d'utilisationClés détenues par AWS, et ne sont pas prises en compte dans le calculAWS KMS des quotas pour votre compte.

Clés gérées par le client

Les clés gérées par le client sont des clés de votreCompte AWS que vous créez, possédez et gérez. Vous disposez d'un contrôle total sur ces clés KMS.

Utilisez une clé gérée par le client pour obtenir les fonctions suivantes.

- Vous créez et gérez la clé gérée par le client, y compris la définition et la mise à jour des politiques clés, des politiques IAM et des autorisations pour contrôler l'accès à la clé gérée par le client. Vous pouvez activer et désactiver la clé gérée par le client, activer et désactiver la rotation automatique des clés, et planifier la suppression de la clé gérée par le client lorsqu'elle n'est plus utilisée. Vous pouvez créer des balises et des alias pour les clés gérées par le client que vous gérez.
- Vous pouvez utiliser une clé gérée par le client avec un élément de clé importé ou dans un magasin de clés personnalisé que vous possédez et gérez.
- Vous pouvez utiliserAWS CloudTrail et Amazon CloudWatch Logs pour effectuer le suivi des demandes que Amazon Keyspaces envoie pourAWS KMS vous à. Pour plus d'informations, veuillez consulter the section called “Étape 6 : Configurer la surveillance avecAWS CloudTrail”.

Les clés gérées par le client sont facturées pour chaque appel d'API, etAWS KMS des quotas s'appliquent à ces clés KMS. Pour plus d'informations, consultez la section Quotas AWS KMSdes ressources ou des demandes.

Lorsque vous spécifiez une clé gérée par le client comme clé de chiffrement de base pour une table, les sauvegardes restaurables sont chiffrées avec la même clé que celle spécifiée pour la table au

moment de leur création. Si la clé KMS de la table est pivotée, la mise en enveloppe des clés garantit que la dernière clé KMS a accès à toutes les sauvegardes restaurables.

Amazon Keyspaces doit avoir accès à votre clé gérée par le client pour vous permettre d'accéder aux données de votre table. Si l'état de la clé de chiffrement est désactivé ou si sa suppression est planifiée, Amazon Keyspaces ne sera pas en mesure de chiffrer ou de déchiffrer les données. Par conséquent, vous n'êtes pas en mesure d'effectuer des opérations de lecture et d'écriture sur la table. Dès que le service détecte que votre clé de chiffrement est inaccessible, Amazon Keyspaces envoie une notification par e-mail pour vous alerter.

Vous devez rétablir l'accès à votre clé de chiffrement dans les sept jours, faute de quoi Amazon Keyspaces supprimera automatiquement votre table. Par mesure de précaution, Amazon Keyspaces crée une sauvegarde restaurable des données de votre table avant de supprimer la table. Amazon Keyspaces conserve la sauvegarde restaurable pendant 35 jours. Après 35 jours, vous ne pouvez plus restaurer les données de votre table. La sauvegarde restaurable ne vous est pas facturée, mais des frais de restauration standard s'appliquent.

Vous pouvez utiliser cette sauvegarde restaurable pour restaurer vos données vers une nouvelle table. Pour lancer la restauration, la dernière clé gérée par le client utilisée pour la table doit être activée et Amazon Keyspaces doit y avoir accès.

> **Note**
>
> Lorsque vous créez une table cryptée à l'aide d'une clé gérée par le client qui est inaccessible ou dont la suppression est planifiée avant la fin du processus de création, une erreur se produit. L'opération de création de table échoue et vous recevez une notification par e-mail.

Notes d'utilisation du chiffrement au repos

Lorsque vous utilisez un chiffrement au repos dans Amazon Keyspaces, tenez compte des considérations suivantes.

- Le chiffrement au repos côté serveur est activé sur toutes les tables Amazon Keyspaces et ne peut pas être désactivé. La table entière est cryptée au repos, vous ne pouvez pas sélectionner de colonnes ou de lignes spécifiques pour le chiffrement.
- Par défaut, Amazon Keyspaces utilise une clé par défaut à service unique (Clé détenue par AWS) pour crypter toutes vos tables. Si cette clé n'existe pas, elle est créée pour vous. Les clés par défaut du service ne peuvent pas être désactivées.

- Le chiffrement au repos ne chiffre les données que lorsqu'elles sont statiques (au repos) sur un support de stockage persistant. Si la sécurité des données est un élément important pour les données en transit ou en cours d'utilisation, vous devez prendre des mesures supplémentaires :
  - Données en transit : toutes vos données dans Amazon Keyspaces sont chiffrées en transit. Par défaut, les communications avec Amazon Keyspaces sont protégées à l'aide du chiffrement Secure Sockets Layer (SSL) /Transport Layer Security (TLS).
  - Données en cours d'utilisation : protégez vos données avant de les envoyer à Amazon Keyspaces à l'aide d'un chiffrement côté client.
  - Clés gérées par le client : les données stockées dans vos tables sont toujours cryptées à l'aide de vos clés gérées par le client. Toutefois, les opérations qui effectuent des mises à jour atomiques de plusieurs lignes cryptent les données temporairement utiliséesClés détenues par AWS pendant le traitement. Cela inclut les opérations de suppression de plages et les opérations qui accèdent simultanément à des données statiques et non statiques.
- Une seule clé gérée par le client peut avoir jusqu'à 50 000 autorisations. Chaque table Amazon Keyspaces associée à une clé gérée par le client nécessite 2 autorisations. Une subvention est accordée lorsque la table est supprimée. La seconde subvention est utilisée pour créer un instantané automatique du tableau afin de se protéger contre la perte de données au cas où Amazon Keyspaces perdrait involontairement l'accès à la clé gérée par le client. Cette autorisation est accordée 42 jours après la suppression de la table.

Chiffrement au repos : comment utiliser des clés gérées par le client pour chiffrer des tables dans Amazon Keyspaces

Vous pouvez utiliser la console ou les instructions CQLAWS KMS key pour spécifier les quatre nouvelles tables et mettre à jour les clés de chiffrement des tables existantes dans Amazon Keyspaces. La rubrique suivante explique comment implémenter des clés gérées par le client pour les tables nouvelles et existantes.

Rubriques

- Prérequis : créer une clé gérée par le client à l'aide d'Amazon KeyspacesAWS KMS et octroyer des autorisations à celui-ci
- Étape 3 : Spécifier une clé gérée par le client pour une nouvelle table
- Étape 4 : Mettre à jour la clé de chiffrement d'une table existante
- Étape 5 : Utiliser le contexte de chiffrement Amazon Keyspaces dans les journaux
- Étape 6 : Configurer la surveillance avecAWS CloudTrail

Prérequis : créer une clé gérée par le client à l'aide d'Amazon KeyspacesAWS KMS et octroyer des autorisations à celui-ci

Avant de protéger une table Amazon Keyspaces à l'aide d'une clé gérée par le client, vous devez d'abord créer la clé dansAWS Key Management Service (AWS KMS), puis autoriser Amazon Keyspaces à utiliser cette clé.

Étape 1 : Créer une clé gérée par le client à l'aide dAWS KMS

Pour créer une clé gérée par le client à utiliser pour protéger une table Amazon Keyspaces, vous pouvez suivre les étapes décrites dans Création de clés KMS de chiffrement symétrique à l'aide de la console ou de l'AWSAPI.

Étape 2 : Autoriser l'utilisation de votre clé gérée par le client

Avant de choisir une clé gérée par le client pour protéger une table Amazon Keyspaces, les politiques associées à cette clé gérée par le client doivent autoriser Amazon Keyspaces à l'utiliser en votre nom. Vous avez un contrôle total des politiques et des octrois de clé gérée par le client. Vous pouvez fournir ces autorisations dans une politique de clé, une politique IAM ou un octroi.

Amazon Keyspaces n'a pas besoin d'une autorisation supplémentaire pour utiliser l'par défaut Clé détenue par AWSet protéger les tables Amazon Keyspaces de votreAWS compte.

Les rubriques suivantes expliquent comment configurer les autorisations requises à l'aide des politiques et des autorisations IAM qui autorisent les tables Amazon Keyspaces à utiliser une clé gérée par le client.

Rubriques

- Politique de clé pour les clés gérées par le client
- Exemple de politique de clé
- Utilisation d'octrois pour autoriser Amazon Keyspaces

Politique de clé pour les clés gérées par le client

Lorsque vous sélectionnez une clé gérée par le client pour protéger une table Amazon Keyspaces, Amazon Keyspaces obtient l'autorisation d'utiliser la clé gérée par le client au nom du mandataire qui effectue la sélection. Ce principal, un utilisateur ou un rôle, doit disposer des autorisations requises par Amazon Keyspaces sur la clé gérée par le client.

Amazon Keyspaces requiert au minimum les autorisations suivantes sur une clé gérée par le client :

- kms:Encrypt
- kms:Decrypt
- kmReEncrypt :* (pour les km :ReEncryptFrom et les km :ReEncryptTo)
- kmGenerateDataKey :* (pour les km :GenerateDataKey et les km :GenerateDataKeyWithoutPlaintext)
- km :DescribeKey
- km :CreateGrant

Exemple de politique de clé

Par exemple, l'exemple de politique de clé suivant fournit uniquement les autorisations requises. La politique a les effets suivants :

- Elle permet à Amazon Keyspaces d'utiliser la clé gérée par le client dans les opérations cryptographiques et de créer des octrois, mais seulement lorsqu'elle agit au nom des principals du compte qui ont l'autorisation d'utiliser Amazon Keyspaces. Si les mandataires spécifiés dans l'énoncé de stratégie ne sont pas autorisés à utiliser Amazon Keyspaces, l'appel échoue, même lorsqu'il provient du service Amazon Keyspaces.
- La clé deViaService condition kms : permet l'accord d'autorisations uniquement lorsque la demande provient d'Amazon Keyspaces au nom des mandataires répertoriés dans l'énoncé de stratégie. Ces principals ne peuvent pas appeler ces opérations directement. Notez que la commande valeur `kms:ViaService`, `cassandra.*.amazonaws.com`, possède un astérisque (*) dans la position Région. Amazon Keyspaces nécessite une autorisation indépendante de tout élément particulierRégion AWS.
- Elle accorde aux administrateurs de clés gérées par le client (utilisateurs qui peuvent endosser le`db-team` rôle) un accès en lecture seule à la clé gérée par le client, ainsi que l'autorisation de révoquer les octrois, en particulier ceux requis par Amazon Keyspaces pour protéger la table.
- Elle accorde à Amazon Keyspaces un accès en lecture seule à la clé gérée par le client. Dans ce cas, Amazon Keyspaces peut appeler ces opérations directement. Il n'est pas nécessaire qu'il agisse au nom du principal du compte.

Avant d'utiliser un exemple de politique de clé, remplacez les exemples de principals par des principals réels de votre Compte AWS.

```
{
```

```
  "Id": "key-policy-cassandra",
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid" : "Allow access through Amazon Keyspaces for all principals in the account
 that are authorized to use Amazon Keyspaces",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:user/db-lead"},
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey",
        "kms:CreateGrant"
      ],
      "Resource": "*",
      "Condition": {
         "StringLike": {
           "kms:ViaService" : "cassandra.*.amazonaws.com"
         }
      }
    },
    {
      "Sid":  "Allow administrators to view the customer managed key and revoke
 grants",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/db-team"
       },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Utilisation d'octrois pour autoriser Amazon Keyspaces

Outre les politiques de clé, Amazon Keyspaces utilise des octrois pour définir des autorisations sur une clé gérée par le client. Pour visualiser les octrois sur une clé gérée par le client dans votre compte, utilisez l'opération ListGrants. Amazon Keyspaces n'a pas besoin d'octrois, ni d'autorisations supplémentaires, pour utiliser la Clé détenue par AWSet protéger votre table.

Amazon Keyspaces utilise les autorisations d'octroi lorsqu'il effectue des tâches de maintenance système en arrière-plan et de protection des données en continu. Il utilise également des octrois pour générer les clés de table.

Chaque octroi est spécifique à une table. Si le compte inclut plusieurs tables chiffrées avec la même clé gérée par le client, il existe un octroi de chaque type pour chaque table. L'autorisation est limitée par le contexte de chiffrement Amazon Keyspaces, qui inclut le nom de la table et l'Compte AWSID. La subvention inclut l'autorisation de retirer la subvention si elle n'est plus nécessaire.

Pour créer les octrois, Amazon Keyspaces doit être autorisé à appeler au`CreateGrant` nom de l'utilisateur qui a créé la table chiffrée.

La politique de clé peut également permettre au compte de révoquer l'octroi sur la clé gérée par le client. Toutefois, si vous révoquez l'octroi sur une table chiffrée active, Amazon Keyspaces ne pourra pas protéger ni maintenir la table.

Étape 3 : Spécifier une clé gérée par le client pour une nouvelle table

Suivez ces étapes pour spécifier la clé gérée par le client sur une nouvelle table à l'aide de la console Amazon Keyspaces ou de CQL.

Créer une table chiffrée à l'aide d'une clé gérée par le client (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le panneau de navigation, choisissez Tables, puis Create table (Créer une table).
3. Sur la page Créer une table, dans la section Détails de la table, sélectionnez un espace clé et donnez un nom à la nouvelle table.
4. Dans la section Schéma, créez le schéma de votre table.
5. Dans la section Paramètres du tableau, choisissez Personnaliser les paramètres.
6. Passez aux paramètres de chiffrement.

   Au cours de cette étape, vous sélectionnez les paramètres de chiffrement du tableau.

Dans la section Chiffrement au repos, sous Choisir unAWS KMS key, choisissez l'option Choisir une autre clé KMS (avancée), puis dans le champ de recherche, choisissez un nom de ressource Amazon (ARN)AWS KMS key ou entrez un nom de ressource Amazon (ARN).

> **Note**
>
> Si la clé que vous avez sélectionnée n'est pas accessible ou ne possède pas les autorisations requises, consultez la section Résolution des problèmes d'accès par clé dans le Guide duAWS Key Management Service développeur.

7. Choisissez Créer pour créer la table chiffrée.

Créer une nouvelle table à l'aide d'une clé gérée par le client pour le chiffrement au repos au repos (CQL)

Pour créer une nouvelle table qui utilise une clé gérée par le client pour le chiffrement au repos au repos, vous pouvez utiliser l'`CREATE TABLE`instruction comme illustré dans l'exemple suivant : Assurez-vous de remplacer l'ARN de la clé par un ARN pour obtenir une clé valide avec des autorisations accordées à Amazon Keyspaces.

```
CREATE TABLE my_keyspace.my_table(id bigint, name text, place text STATIC, PRIMARY
 KEY(id, name)) WITH CUSTOM_PROPERTIES = {
        'encryption_specification':{
                'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
            }
    };
```

Si vous recevez un`Invalid Request Exception`, vous devez confirmer que la clé gérée par le client est valide et qu'Amazon Keyspaces dispose des autorisations requises. Pour vérifier que la clé a été correctement configurée, consultez la section Résolution des problèmes d'accès par clé dans le Guide duAWS Key Management Service développeur.

Étape 4 : Mettre à jour la clé de chiffrement d'une table existante

Vous pouvez également utiliser la console Amazon Keyspaces ou CQL pour modifier à tout moment les clés de chiffrement d'une table existante entre une cléClé détenue par AWS et une clé KMS gérée par le client.

Mettre à jour une table existante avec la nouvelle clé gérée par le client (console)

1. Connectez-vous à et ouvrezAWS Management Console la console Amazon Keyspaces à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Dans le volet de navigation, choisissez Tables.
3. Choisissez la version que vous souhaitez mettre à jour, puis l'onglet Paramètres supplémentaires.
4. Dans la section Chiffrement au repos, choisissez Gérer le chiffrement pour modifier les paramètres de chiffrement du tableau.

   Sous Choisir unAWS KMS key, choisissez l'option Choisir une autre clé KMS (avancée), puis dans le champ de recherche, choisissez un nom de ressource Amazon (ARN)AWS KMS key ou entrez un nom de ressource Amazon (ARN).

   > **Note**
   >
   > Si la clé que vous avez sélectionnée n'est pas valide, consultez la section Résolution des problèmes d'accès par clé dans le Guide duAWS Key Management Service développeur.

   Vous pouvez également choisir unClé détenue par AWS pour une table chiffrée avec une clé gérée par le client.
5. Choisissez Enregistrer les modifications pour enregistrer les modifications apportées au tableau.

Mettre à jour la clé de chiffrement utilisée pour une table existante

Pour modifier la clé de chiffrement d'une table existante, vous utilisez l'`ALTER TABLE`instruction pour spécifier une clé gérée par le client pour le chiffrement au repos. Assurez-vous de remplacer l'ARN de la clé par un ARN pour obtenir une clé valide avec des autorisations accordées à Amazon Keyspaces.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
              'encryption_specification':{
                     'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                     'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                  }
         };
```

Si vous recevez un`Invalid Request Exception`, vous devez confirmer que la clé gérée par le client est valide et qu'Amazon Keyspaces dispose des autorisations requises. Pour vérifier que la clé a été correctement configurée, consultez la section Résolution des problèmes d'accès par clé dans le Guide duAWS Key Management Service développeur.

Pour rétablir la clé de chiffrement à l'option de chiffrement par défaut au repos avecClés détenues par AWS, vous pouvez utiliser l'`ALTER TABLE`instruction comme indiqué dans l'exemple suivant.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
                'encryption_specification':{
                      'encryption_type' : 'AWS_OWNED_KMS_KEY'
                    }
         };
```

Étape 5 : Utiliser le contexte de chiffrement Amazon Keyspaces dans les journaux

Un contexte de chiffrement est un ensemble de paires clé-valeur qui contiennent des données non secrètes arbitraires. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie de manière chiffrée le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez transmettre le même contexte de chiffrement.

Amazon Keyspaces utilise le même contexte de chiffrement dans toutes les opérationsAWS KMS cryptographiques. Si vous utilisez une clé gérée par le client pour protéger votre table Amazon Keyspaces, vous pouvez utiliser le contexte de chiffrement pour identifier l'utilisation de la clé gérée par le client dans les enregistrements d'audit et les journaux. Il apparaît également en texte brut dans les journaux, tels que les journaux pour AWS CloudTrailet Amazon CloudWatch Logs.

Dans ses demandes adressées àAWS KMS, Amazon Keyspaces utilise un contexte de chiffrement avec trois paires clé-valeur.

```
"encryptionContextSubset": {
    "aws:cassandra:keyspaceName": "my_keyspace",
    "aws:cassandra:tableName": "mytable"
    "aws:cassandra:subscriberId": "111122223333"
}
```

- Keyspace — La première paire clé-valeur identifie l'espace de clés qui inclut la table chiffrée par Amazon Keyspaces. La clé est `aws:cassandra:keyspaceName`. La valeur est le nom de l'espace de clés.

```
"aws:cassandra:keyspaceName": "<keyspace-name>"
```

Par exemple :

```
"aws:cassandra:keyspaceName": "my_keyspace"
```

- Table — La deuxième paire clé-valeur identifie la table chiffrée par Amazon Keyspaces. La clé est `aws:cassandra:tableName`. La valeur correspond au nom de la table.

  ```
  "aws:cassandra:tableName": "<table-name>"
  ```

  Par exemple :

  ```
  "aws:cassandra:tableName": "my_table"
  ```

- Compte — La troisième paire clé-valeur identifie leCompte AWS. La clé est `aws:cassandra:subscriberId`. La valeur correspond à l'ID de compte.

  ```
  "aws:cassandra:subscriberId": "<account-id>"
  ```

  Par exemple :

  ```
  "aws:cassandra:subscriberId": "111122223333"
  ```

Étape 6 : Configurer la surveillance avecAWS CloudTrail

Si vous utilisez une clé gérée par le client pour protéger vos tables Amazon Keyspaces, vous pouvez utiliserAWS CloudTrail les journaux pour suivre les demandes envoyées par Amazon Keyspaces àAWS KMS en votre nom.

Les`GenerateDataKey`,`DescribeKey`et`Decrypt`, et les`CreateGrant` demandes sont abordées dans cette section. En outre, Amazon Keyspaces utilise une RetireGrantopération pour supprimer une autorisation lorsque vous supprimez un tableau.

GenerateDataKey

Amazon Keyspaces crée une clé de table unique pour chiffrer les données au repos au repos. Il envoie une GenerateDataKeydemande àAWS KMS laquelle est spécifiée la clé KMS de la table.

L'événement qui enregistre l'opération `GenerateDataKey` est similaire à l'exemple d'événement suivant. L'utilisateur est le compte de service Amazon Keyspaces. Les paramètres incluent l'Amazon Resource Name (ARN) de la clé gérée par le client, un spécificateur de clé qui nécessite une clé de 256 bits et le contexte de chiffrement qui identifie l'espace de clés, la table et leCompte AWS.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T04:56:05Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keySpec": "AES_256",
        "encryptionContext": {
            "aws:cassandra:keyspaceName": "my_keyspace",
            "aws:cassandra:tableName": "my_table",
            "aws:cassandra:subscriberId": "123SAMPLE012"
        },
        "keyId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
    },
    "responseElements": null,
    "requestID": "5e8e9cb5-9194-4334-aacc-9dd7d50fe246",
    "eventID": "49fccab9-2448-4b97-a89d-7d5c39318d6f",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
```

```
    "recipientAccountId": "123SAMPLE012",
    "sharedEventID": "84fbaaf0-9641-4e32-9147-57d2cb08792e"
}
```

DescribeKey

Amazon Keyspaces utilise une DescribeKeyopération pour déterminer si la clé KMS que vous avez sélectionnée existe dans le compte et la région.

L'événement qui enregistre l'opération `DescribeKey` est similaire à l'exemple d'événement suivant. L'utilisateur est le compte de service Amazon Keyspaces. Les paramètres incluent l'ARN de la clé gérée par le client et un spécificateur de clé qui nécessite une clé de 256 bits.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAZ3FNIIVIZZ6H7CFQG",
        "arn": "arn:aws:iam::123SAMPLE012:user/admin",
        "accountId": "123SAMPLE012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "admin",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-16T04:55:42Z"
            }
        },
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T04:55:58Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
    },
    "responseElements": null,
```

```
    "requestID": "c25a8105-050b-4f52-8358-6e872fb03a6c",
    "eventID": "0d96420e-707e-41b9-9118-56585a669658",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012"
}
```

Decrypt

Lorsque vous accédez à une table Amazon Keyspaces, Amazon Keyspaces a besoin de déchiffrer la clé de table pour pouvoir déchiffrer les clés situées au-dessous dans la hiérarchie. Il déchiffre ensuite les données de la table. Pour déchiffrer la clé de table, Amazon Keyspaces envoie une demande Decrypt àAWS KMS qui spécifie la clé KMS de la table.

L'événement qui enregistre l'opération `Decrypt` est similaire à l'exemple d'événement suivant. L'utilisateur est le principal de votre Compte AWS qui accède à la table. Les paramètres incluent la clé de table chiffrée (comme un objet blob de texte chiffré) et le contexte de chiffrement qui identifie la table et leCompte AWS. AWS KMSdérive l'ID de la clé gérée par le client à partir du texte chiffré.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T05:29:44Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
```

```
    "requestParameters": {
        "encryptionContext": {
            "aws:cassandra:keyspaceName": "my_keyspace",
            "aws:cassandra:tableName": "my_table",
            "aws:cassandra:subscriberId": "123SAMPLE012"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "50e80373-83c9-4034-8226-5439e1c9b259",
    "eventID": "8db9788f-04a5-4ae2-90c9-15c79c411b6b",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012",
    "sharedEventID": "7ed99e2d-910a-4708-a4e3-0180d8dbb68e"
}
```


CreateGrant

Lorsque vous utilisez une clé gérée par le client pour protéger votre table Amazon Keyspaces, Amazon Keyspaces utilise des octrois pour autoriser le service à assurer une protection continue des données, ainsi que des tâches de maintenance et de durabilité. Ces subventions ne sont pas requises pour Clés détenues par AWS.

Les octrois qu'Amazon Keyspaces crée sont spécifiques à une table. Le principal figurant dans la CreateGrantdemande est l'utilisateur qui a créé la table.

L'événement qui enregistre l'opération `CreateGrant` est similaire à l'exemple d'événement suivant. Les paramètres incluent l'ARN de la clé gérée par le client pour la table, le principal bénéficiaire et le principal de retrait (le service Amazon Keyspaces), ainsi que les opérations couvertes par la subvention. Elle inclut également une contrainte qui exige que toutes les opérations de chiffrement utilisent le contexte de chiffrement spécifié.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAZ3FNIIVIZZ6H7CFQG",
        "arn": "arn:aws:iam::arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111:user/admin",
        "accountId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "admin",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-16T04:55:42Z"
            }
        },
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T05:11:10Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "a7d328af-215e-4661-9a69-88c858909f20",
        "operations": [
            "DescribeKey",
            "GenerateDataKey",
            "Decrypt",
            "Encrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "RetireGrant"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:cassandra:keyspaceName": "my_keyspace",
                "aws:cassandra:tableName": "my_table",
                "aws:cassandra:subscriberId": "123SAMPLE012"
```

```
            }
        },
        "retiringPrincipal": "cassandratest.us-east-1.amazonaws.com",
        "granteePrincipal": "cassandratest.us-east-1.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "18e4235f1b07f289762a31a1886cb5efd225f069280d4f76cd83b9b9b5501013"
    },
    "requestID": "b379a767-1f9b-48c3-b731-fb23e865e7f7",
    "eventID": "29ee1fd4-28f2-416f-a419-551910d20291",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012"
}
```


Chiffrement en transit dans Amazon Keyspaces

Amazon Keyspaces accepte uniquement les connexions sécurisées à l'aide du protocole TLS (Transport Layer Security). Le chiffrement en cours de transfert fournit un niveau supplémentaire de protection des données en chiffrant vos données lorsqu'elles transitent vers et depuis Amazon Keyspaces. Les politiques organisationnelles, les réglementations sectorielles ou gouvernementales et les exigences de conformité nécessitent souvent l'utilisation du chiffrement en transit afin de renforcer la sécurité des données de vos applications lorsqu'elles transmettent des données sur le réseau.

Pour savoir comment crypter`cqlsh` les connexions à Amazon Keyspaces à l'aide du protocole TLS, consultezthe section called “Comment configurer manuellement les `cqlsh` connexions pour le protocole TLS”. Pour savoir comment utiliser le Chiffrement TLS à l'aide des pilotes clients, consultezthe section called “Utilisation d'un pilote client Cassandra”.

Confidentialité du trafic interréseau dans Amazon Keyspaces

Cette rubrique décrit comment Amazon Keyspaces (pour Apache Cassandra) sécurise les connexions entre les applications locales et Amazon Keyspaces et entre Amazon Keyspaces et d'autres ressources au sein de celles-ci. AWS Région AWS

Trafic entre les clients de service et sur site et les applications

Vous disposez de deux options de connectivité entre votre réseau privé et AWS:

- Une connexion AWS Site-to-Site VPN. Pour plus d'informations, consultez Présentation de AWS Site-to-Site VPN dans le Guide de l'utilisateur AWS Site-to-Site VPN.
- Une connexion AWS Direct Connect. Pour plus d'informations, consultez Présentation de AWS Direct Connect dans le Guide de l'utilisateur AWS Direct Connect.

En tant que service géré, Amazon Keyspaces (pour Apache Cassandra) est protégé par un système de sécurité réseau AWS mondial. Pour plus d'informations sur les services de sécurité AWS et la manière dont AWS protège l'infrastructure, consultez la section Sécurité du cloud AWS. Pour concevoir votre environnement AWS en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le Security Pillar AWS Well-Architected Framework (Pilier de sécurité de l'infrastructure Well-Architected Framework).

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon Keyspaces via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et nous recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Amazon Keyspaces prend en charge deux méthodes d'authentification des demandes des clients. La première méthode utilise des informations d'identification spécifiques au service, qui sont des informations d'identification basées sur un mot de passe générées pour un utilisateur IAM spécifique.

Vous pouvez créer et gérer le mot de passe à l'aide de la console IAMAWS CLI, de ou de l'AWSAPI. Pour plus d'informations, consultez Utilisation d'IAM avec Amazon Keyspaces.

La seconde méthode utilise un plugin d'authentification pour le pilote DataStax Java open source pour Cassandra. Ce plug-in permet aux utilisateurs, aux rôles et aux identités fédérées IAM d'ajouter des informations d'authentification aux demandes d'API Amazon Keyspaces (pour Apache Cassandra) à l'aide du processus AWS Signature Version 4 (SigV4). Pour plus d'informations, veuillez consulter the section called “Informations d'identification IAM pour l' AWS authentification”.

Trafic entre des ressources AWS dans la même Région

Les points de terminaison d'interface VPC permettent une communication privée entre votre cloud privé virtuel (VPC) s'exécutant dans Amazon VPC et Amazon Keyspaces. Les points de terminaison d'un VPC d'interface sont alimentés par AWS PrivateLink, qui est un service AWS permettant une communication privée entre les VPC et les services AWS. AWS PrivateLink permet cela en utilisant une interface réseau Elastic avec des adresses IP privées dans votre VPC afin que le trafic réseau ne quitte pas le réseau Amazon. Les points de terminaison de VPC d'interface n'ont pas besoin d'une passerelle Internet, d'un périphérique NAT, d'une connexion VPN ou d'une connexion AWS Direct Connect. Pour plus d'informations, consultez Amazon Virtual Private Cloud and Interface VPC endpoints () AWS PrivateLink. Pour obtenir des exemples de politiques, consultez the section called “Utilisation des points de terminaison VPC de l'interface pour Amazon Keyspaces”.

AWS Identity and Access Management pour Amazon Keyspaces

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources Amazon Keyspaces. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

- Utilisation de rôles liés à un service pour Amazon Keyspaces

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans Amazon Keyspaces.

Utilisateur du service — Si vous utilisez le service Amazon Keyspaces pour faire votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Au fur et à mesure que vous utilisez de plus en plus les fonctionnalités d'Amazon Keyspaces dans le cadre de votre travail, il se peut que vous ayez besoin d'autorisations supplémentaires. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur. Si vous ne parvenez pas à accéder à une fonctionnalité d'Amazon Keyspaces, consultez. Résolution des problèmes d'identité et d'accès à Amazon Keyspaces

Administrateur du service — Si vous êtes responsable des ressources Amazon Keyspaces au sein de votre entreprise, vous avez probablement un accès complet à Amazon Keyspaces. C'est à vous de déterminer les fonctionnalités et les ressources d'Amazon Keyspaces auxquelles les utilisateurs de vos services doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour en savoir plus sur la manière dont votre entreprise peut utiliser IAM avec Amazon Keyspaces, consultez. Comment Amazon Keyspaces fonctionne avec IAM

Administrateur IAM — Si vous êtes administrateur IAM, vous souhaiterez peut-être en savoir plus sur la manière dont vous pouvez rédiger des politiques pour gérer l'accès à Amazon Keyspaces. Pour consulter des exemples de politiques basées sur l'identité d'Amazon Keyspaces que vous pouvez utiliser dans IAM, consultez. Exemples de politiques basées sur l'identité d'Amazon Keyspaces

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une

identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez la section Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer vous-même les demandes, consultez la section Signature des demandes AWS d'API dans le guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour en savoir plus, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Utilisation de l'authentification multifactorielle (MFA) dans l'interface AWS dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur racine, consultez Tâches nécessitant les informations d'identification de l'utilisateur racine dans le Guide de l'utilisateur IAM.

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous

recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons de faire pivoter les clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d'administrer des ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d'un rôle) dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Vous pouvez assumer temporairement un rôle IAM dans le en AWS Management Console changeant de rôle. Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Utilisation de rôles IAM dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré – Pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez la rubrique Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

- Autorisations d’utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d’autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d’un compte différent d’accéder aux ressources de votre compte. Les rôles constituent le principal moyen d’accorder l’accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour connaître la différence entre les rôles et les politiques basées sur les ressources pour l'accès entre comptes, consultez la section Accès aux ressources entre comptes dans IAM dans le guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d’appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d’autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d’accès.
  - Rôle de service : il s’agit d’un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d’IAM. Pour plus d’informations, consultez Création d’un rôle pour la délégation d’autorisations à un Service AWS dans le Guide de l’utilisateur IAM.
  - Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d’effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage

des clés d’accès au sein de l’instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d’instance contient le rôle et permet aux programmes qui s’exécutent sur l’instance EC2 d’obtenir des informations d’identification temporaires. Pour plus d’informations, consultez Utilisation d’un rôle IAM pour accorder des autorisations à des applications s’exécutant sur des instances Amazon EC2 dans le Guide de l’utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, consultez Quand créer un rôle IAM (au lieu d’un utilisateur) dans le Guide de l’utilisateur IAM.

Gestion des accès à l’aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d’informations sur la structure et le contenu des documents de politique JSON, consultez Vue d’ensemble des politiques JSON dans le Guide de l’utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C’est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d’aucune autorisation. Pour octroyer aux utilisateurs des autorisations d’effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L’administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d’une action, quelle que soit la méthode que vous utilisez pour exécuter l’opération. Par exemple, supposons que vous disposiez d’une politique qui autorise l’action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l’identité

Les politiques basées sur l’identité sont des documents de politique d’autorisations JSON que vous pouvez attacher à une identité telle qu’un utilisateur, un groupe d’utilisateurs ou un rôle IAM. Ces

politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principals (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, consultez Vue d'ensemble des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) — Les SCP sont des politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans. AWS Organizations AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer les politiques de contrôle des services (SCP) à l'un ou à l'ensemble de vos comptes. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Pour plus d'informations sur les organisations et les SCP, consultez Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations .
- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser

une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Comment Amazon Keyspaces fonctionne avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon Keyspaces, vous devez connaître les fonctionnalités IAM disponibles avec Amazon Keyspaces. Pour obtenir une vue d'ensemble de la façon dont Amazon Keyspaces et les autres AWS services fonctionnent avec IAM, consultez les AWS services compatibles avec IAM dans le guide de l'utilisateur d'IAM.

Rubriques

- Politiques basées sur l'identité d'Amazon Keyspaces
- Politiques basées sur les ressources d'Amazon Keyspaces
- Autorisation basée sur les tags Amazon Keyspaces
- Rôles IAM chez Amazon Keyspaces

Politiques basées sur l'identité d'Amazon Keyspaces

Avec les politiques IAM basées sur l’identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Amazon Keyspaces prend en charge des actions et des ressources spécifiques, ainsi que des clés de condition. Pour en savoir plus sur tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Pour voir les ressources et actions spécifiques au service Amazon Keyspaces, ainsi que les clés contextuelles de condition qui peuvent être utilisées pour les politiques d'autorisation IAM, consultez les actions, les ressources et les clés de condition pour Amazon Keyspaces (pour Apache Cassandra) dans le Service Authorization Reference.

Actions

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C’est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L’élément `Action` d’une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l’accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec

autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une stratégie afin d'accorder l'autorisation d'exécuter les opérations associées.

Les actions politiques dans Amazon Keyspaces utilisent le préfixe suivant avant l'action :. `cassandra:` Par exemple, pour autoriser quelqu'un à créer un espace de touches Amazon Keyspaces à l'aide de l'instruction Amazon Keyspaces `CREATE` CQL, vous devez inclure l'action dans sa politique. `cassandra:Create` Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`. Amazon Keyspaces définit son propre ensemble d'actions décrivant les tâches que vous pouvez effectuer avec ce service.

Pour spécifier plusieurs actions dans une seule déclaration, séparez-les par des virgules comme suit :

```
"Action": [
      "cassandra:CREATE",
      "cassandra:MODIFY"
          ]
```

Pour consulter la liste des actions Amazon Keyspaces, consultez la section Actions définies par Amazon Keyspaces (pour Apache Cassandra) dans le Service Authorization Reference.

Ressources

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Dans Amazon Keyspaces, les espaces clés et les tables peuvent être utilisés dans le cadre des autorisations `Resource` IAM.

La ressource keyspace Amazon Keyspaces possède l'ARN suivant :

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/
```

La ressource de table Amazon Keyspaces possède l'ARN suivant :

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/table/
${tableName}
```

Pour plus d'informations sur le format des ARN, consultez Amazon Resource Names (ARN) et espaces de noms de AWS services.

Par exemple, pour spécifier le keyspace `mykeyspace` dans votre déclaration, utilisez l'ARN suivant :

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/mykeyspace/"
```

Pour spécifier toues les instances qui appartiennent à un compte spécifique, utilisez le caractère générique (*) :

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/*"
```

Certaines actions Amazon Keyspaces, telles que celles relatives à la création de ressources, ne peuvent pas être effectuées sur une ressource spécifique. Dans ces cas-là, vous devez utiliser le caractère générique (*).

```
"Resource": "*"
```

Pour se connecter à Amazon Keyspaces par programmation à l'aide d'un pilote standard, le principal doit disposer d'un accès SELECT aux tables système, car la plupart des pilotes lisent les espaces-clés/tables du système lors de la connexion. Par exemple, pour accorder `SELECT` des autorisations à un utilisateur IAM pour `mytable` in`mykeyspace`, le principal doit être autorisé à lire à la fois, `mytable` ainsi que le`system keyspace`. Pour spécifier plusieurs ressources dans une seule instruction, séparez leurs ARN par des virgules.

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
```

Pour consulter la liste des types de ressources Amazon Keyspaces et de leurs ARN, consultez la section Ressources définies par Amazon Keyspaces (pour Apache Cassandra) dans le Service Authorization Reference. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par Amazon Keyspaces (pour Apache Cassandra).

Clés de condition

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Amazon Keyspaces définit son propre ensemble de clés de condition et prend également en charge l'utilisation de certaines clés de condition globales. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Toutes les actions Amazon Keyspaces prennent en charge les clés `aws:RequestTag/${TagKey}aws:ResourceTag/${TagKey}`, le et les clés de `aws:TagKeys` condition. Pour plus d'informations, consultez the section called " Accès aux ressources Amazon Keyspaces basé sur des balises".

Pour consulter la liste des clés de condition Amazon Keyspaces, consultez la section Clés de condition pour Amazon Keyspaces (pour Apache Cassandra) dans le Service Authorization Reference. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par Amazon Keyspaces (pour Apache Cassandra).

Exemples

Pour consulter des exemples de politiques basées sur l'identité d'Amazon Keyspaces, consultez. Exemples de politiques basées sur l'identité d'Amazon Keyspaces

Politiques basées sur les ressources d'Amazon Keyspaces

Amazon Keyspaces ne prend pas en charge les politiques basées sur les ressources. Pour afficher un exemple de page de stratégie basée sur les ressources détaillée, consultez https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

Autorisation basée sur les tags Amazon Keyspaces

Vous pouvez gérer l'accès à vos ressources Amazon Keyspaces à l'aide de balises. Pour gérer l'accès aux ressources basé sur des balises, vous devez fournir les informations de balise dans l'élément Condition d'une stratégie utilisant les clés de condition `cassandra:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Pour plus d'informations sur le balisage des ressources Amazon Keyspaces, consultez. the section called "Utilisation des tags"

Pour visualiser un exemple de stratégie basée sur l'identité permettant de limiter l'accès à une ressource en fonction des balises de cette ressource, veuillez consulter Accès aux ressources Amazon Keyspaces basé sur des balises.

Rôles IAM chez Amazon Keyspaces

Un rôle IAM est une entité au sein de vous Compte AWS qui possède des autorisations spécifiques.

Utilisation d'informations d'identification temporaires avec Amazon Keyspaces

Vous pouvez utiliser des informations d'identification temporaires pour vous connecter à l'aide de la fédération pour endosser un rôle IAM ou bien un rôle entre comptes. Vous obtenez des informations

d'identification de sécurité temporaires en appelant des opérations d' AWS STS API telles que AssumeRoleou GetFederationToken.

Amazon Keyspaces prend en charge l'utilisation d'informations d'identification temporaires avec le plugin d'authentification AWS Signature Version 4 (SigV4) disponible sur le référentiel Github pour les langues suivantes :

- Java :https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin.
- Node.js :https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.
- Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.
- Allez :https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Pour des exemples et des didacticiels qui implémentent le plugin d'authentification pour accéder à Amazon Keyspaces par programmation, consultez. the section called “Utilisation d'un pilote client Cassandra”

Rôles liés à un service

Les rôles liés aux AWS services permettent aux services d'accéder aux ressources d'autres services pour effectuer une action en votre nom. Les rôles liés à un service s’affichent dans votre compte IAM et sont la propriété du service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour en savoir plus sur la création ou la gestion des rôles liés aux services Amazon Keyspaces, consultez. the section called “Utilisation des rôles liés à un service”

Fonctions du service

Amazon Keyspaces ne prend pas en charge les rôles de service.

Exemples de politiques basées sur l'identité d'Amazon Keyspaces

Par défaut, les utilisateurs et les rôles IAM ne sont pas autorisés à créer ou à modifier les ressources Amazon Keyspaces. Ils ne peuvent pas non plus effectuer de tâches à l'aide de la console, du CQLSH ou AWS de AWS CLI l'API. Un administrateur IAM doit créer des politiques IAM autorisant les utilisateurs et les rôles à exécuter des opérations d'API spécifiques sur les ressources spécifiées dont ils ont besoin. Il doit ensuite attacher ces politiques aux utilisateurs ou aux groupes IAM ayant besoin de ces autorisations.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, veuillez consulter Création de politiques dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Rubriques

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer les ressources Amazon Keyspaces de votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accorder les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation de IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utiliser des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service

AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

- Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : IAM Access Analyzer valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politique IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue. Compte AWS Pour exiger le MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console Amazon Keyspaces

Amazon Keyspaces n'a pas besoin d'autorisations spécifiques pour accéder à la console Amazon Keyspaces. Vous devez disposer d'au moins des autorisations en lecture seule pour répertorier et consulter les informations relatives aux ressources Amazon Keyspaces dans votre. Compte AWS Si vous créez une politique basée sur l'identité qui est plus restrictive que les autorisations minimales requises, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs et rôles IAM) tributaires de cette politique.

Deux politiques AWS gérées sont à la disposition des entités pour l'accès à la console Amazon Keyspaces.

- AmazonKeyspacesReadOnlyAccess_v2 — Cette politique accorde un accès en lecture seule à Amazon Keyspaces.
- AmazonKeyspacesFullAccess— Cette politique accorde l'autorisation d'utiliser Amazon Keyspaces avec un accès complet à toutes les fonctionnalités.

Pour plus d'informations sur les politiques gérées par Amazon Keyspaces, consultez. the section called "Politiques gérées par AWS"

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Accès aux tables Amazon Keyspaces

Voici un exemple de politique qui accorde un accès en lecture seule (SELECT) aux tables du système Amazon Keyspaces. Pour tous les exemples, remplacez la région et l'ID de compte dans le nom de ressource Amazon (ARN) par les vôtres.

> **Note**
>
> Pour se connecter avec un pilote standard, un utilisateur doit avoir au moins un accès SELECT aux tables système, car la plupart des pilotes lisent les keyspaces/tables système lors de la connexion.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

L'exemple de politique suivant ajoute un accès en lecture seule à la table utilisateur `mytable` dans le keyspace. `mykeyspace`

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
```

```
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

L'exemple de stratégie suivant affecte l'accès en lecture/écriture à une table utilisateur et l'accès en lecture aux tables système.

> **Note**
>
> Les tables système sont toujours en lecture seule.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

L'exemple de stratégie suivant permet à un utilisateur de créer des tables dans le keyspace `mykeyspace`.

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

```
         "Effect":"Allow",
         "Action":[
            "cassandra:Create",
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/*",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Accès aux ressources Amazon Keyspaces basé sur des balises

Vous pouvez utiliser les conditions de votre politique basée sur l'identité pour contrôler l'accès aux ressources Amazon Keyspaces en fonction de balises. Ces politiques contrôlent la visibilité des espaces clés et des tables du compte. Notez que les autorisations basées sur des balises pour les tables système se comportent différemment lorsque les demandes sont effectuées à l'aide du AWS SDK par rapport aux appels d'API Cassandra Query Language (CQL) via les pilotes Cassandra et les outils de développement.

- Pour effectuer `Get` des demandes `List` et y affecter des ressources avec le AWS SDK lors de l'utilisation de l'accès basé sur des balises, l'appelant doit disposer d'un accès en lecture aux tables système. Par exemple, des autorisations `Select` d'action sont nécessaires pour lire les données des tables système via l'`GetTable`opération. Si l'appelant ne dispose que d'un accès basé sur des balises à une table spécifique, une opération nécessitant un accès supplémentaire à une table système échouera.
- Pour des raisons de compatibilité avec le comportement établi du pilote Cassandra, les politiques d'autorisation basées sur les balises ne sont pas appliquées lors de l'exécution d'opérations sur les tables système à l'aide d'appels d'API CQL (Cassandra Query Language) via des pilotes Cassandra et des outils de développement.

L'exemple suivant montre comment créer une stratégie qui accorde des autorisations à un utilisateur pour afficher une table si son `Owner` contient la valeur du nom d'utilisateur de cet utilisateur. Dans cet exemple, vous accordez également un accès en lecture aux tables système.

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

```
   "Statement":[
      {
         "Sid":"ReadOnlyAccessTaggedTables",
         "Effect":"Allow",
         "Action":"cassandra:Select",
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/*",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ],
         "Condition":{
            "StringEquals":{
               "aws:ResourceTag/Owner":"${aws:username}"
            }
         }
      }
   ]
}
```

Vous pouvez rattacher cette politique aux utilisateurs IAM de votre compte. Si un utilisateur nommé `richard-roe` tente de consulter une table Amazon Keyspaces, la table doit être `Owner=richard-roe` balisée ou. `owner=richard-roe` Dans le cas contraire, l'utilisateur se voit refuser l'accès. La clé de condition d'étiquette `Owner` correspond à la fois à `Owner` et à `owner`, car les noms de clé de condition ne sont pas sensibles à la casse. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

La stratégie suivante autorise un utilisateur à créer des tables avec des balises si l'`Owner` de la table contient la valeur du nom d'utilisateur de cet utilisateur.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
          "Sid": "CreateTagTableUser",
          "Effect": "Allow",
          "Action": [
              "cassandra:Create",
              "cassandra:TagResource"
          ],
          "Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/*",
          "Condition":{
             "StringEquals":{
```

```
                "aws:RequestTag/Owner":"${aws:username}"
            }
        }
    }
]
}
```

AWSpolitiques gérées pour Amazon Keyspaces

Une politique gérée par AWS est une politique autonome créée et administrée par AWS. Les politiques gérées par AWS sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

Gardez à l'esprit que les politiques gérées par AWS peuvent ne pas accorder les autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont disponibles pour tous les clients AWS. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les stratégies gérées par AWS. Si AWS met à jour les autorisations définies dans une politique gérée par AWS, la mise à jour affecte toutes les identités de principal (utilisateurs, groupes et rôles) auxquelles la politique est associée. AWS est plus susceptible de mettre à jour une politique gérée par AWS lorsqu'un nouveau Service AWS est lancé ou que de nouvelles opérations API deviennent accessibles pour les services existants.

Pour plus d'informations, consultez la rubrique Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

AWSpolitique gérée : AmazonKeyspacesReadOnlyAccess _v2

Vous pouvez attacher la politique `AmazonKeyspacesReadOnlyAccess_v2` à vos identités IAM.

Cette politique accorde un accès en lecture seule à Amazon Keyspaces et inclut les autorisations requises lors de la connexion via des points de terminaison VPC privés.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes.

- `Amazon Keyspaces`— Fournit un accès en lecture seule à Amazon Keyspaces.
- `Application Auto Scaling`— Permet aux principaux de visualiser les configurations depuis Application Auto Scaling. Cela est nécessaire pour que les utilisateurs puissent consulter les politiques de dimensionnement automatique associées à une table.
- `CloudWatch`— Permet aux principaux de consulter les données métriques et les alarmes configurées dans CloudWatch. Cela est nécessaire pour que les utilisateurs puissent voir la taille de la table facturable et les CloudWatch alarmes configurées pour une table.
- `AWS KMS`— Permet aux principaux d'afficher les clés configurées dansAWS KMS. Cela est nécessaire pour que les utilisateurs puissent consulter AWS KMS les clés qu'ils créent et gèrent dans leur compte afin de confirmer que la clé attribuée à Amazon Keyspaces est une clé de chiffrement symétrique activée.
- `Amazon EC2`— Permet aux principaux se connectant à Amazon Keyspaces via des points de terminaison VPC d'interroger le VPC de votre instance Amazon EC2 pour obtenir des informations sur le point de terminaison et l'interface réseau. Cet accès en lecture seule à l'instance Amazon EC2 est requis pour qu'Amazon Keyspaces puisse rechercher et stocker les points de terminaison VPC d'interface disponibles dans le tableau utilisé pour l'équilibrage de charge de connexion. `system.peers`

Pour consulter le `JSON` format de la politique, consultez AmazonKeyspacesReadOnlyAccess_v2.

AWSpolitique gérée : AmazonKeyspacesReadOnlyAccess

Vous pouvez attacher la politique `AmazonKeyspacesReadOnlyAccess` à vos identités IAM.

Cette politique accorde un accès en lecture seule à Amazon Keyspaces.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes.

- `Amazon Keyspaces`— Fournit un accès en lecture seule à Amazon Keyspaces.
- `Application Auto Scaling`— Permet aux principaux de visualiser les configurations depuis Application Auto Scaling. Cela est nécessaire pour que les utilisateurs puissent consulter les politiques de dimensionnement automatique associées à une table.
- `CloudWatch`— Permet aux principaux de consulter les données métriques et les alarmes configurées dans CloudWatch. Cela est nécessaire pour que les utilisateurs puissent voir la taille de la table facturable et les CloudWatch alarmes configurées pour une table.
- `AWS KMS`— Permet aux principaux d'afficher les clés configurées dansAWS KMS. Cela est nécessaire pour que les utilisateurs puissent consulter AWS KMS les clés qu'ils créent et gèrent dans leur compte afin de confirmer que la clé attribuée à Amazon Keyspaces est une clé de chiffrement symétrique activée.

Pour consulter le `JSON` format de la politique, voir AmazonKeyspacesReadOnlyAccess.

AWS Politique gérée par: AmazonKeyspacesFullAccess

Vous pouvez attacher la politique `AmazonKeyspacesFullAccess` à vos identités IAM.

Cette politique accorde des autorisations administratives qui permettent à vos administrateurs d'accéder sans restriction à Amazon Keyspaces.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes.

- `Amazon Keyspaces`— Permet aux directeurs d'accéder à n'importe quelle ressource Amazon Keyspaces et d'effectuer toutes les actions.
- `Application Auto Scaling`— Permet aux principaux de créer, de consulter et de supprimer des politiques de dimensionnement automatique pour les tables Amazon Keyspaces. Cela est nécessaire pour que les administrateurs puissent gérer les politiques de dimensionnement automatique pour les tables Amazon Keyspaces.
- `CloudWatch`— Permet aux donneurs d'ordre de voir la taille de la table facturable ainsi que de créer, de consulter et de supprimer des CloudWatch alarmes conformément aux politiques

de dimensionnement automatique d'Amazon Keyspaces. Cela est nécessaire pour que les administrateurs puissent consulter la taille de la table facturable et créer un CloudWatch tableau de bord.

- `IAM`— Permet à Amazon Keyspaces de créer automatiquement des rôles liés à un service avec IAM lorsque les fonctionnalités suivantes sont activées :
  - `Application Auto Scaling`— Lorsqu'un administrateur active Application Auto Scaling pour une table, Amazon Keyspaces crée un rôle lié à un service pour effectuer des actions de dimensionnement automatique en votre nom.
  - `Amazon Keyspaces Multi-Region Replication`— Lorsqu'un administrateur crée un espace clé multirégional, un rôle lié à un service est automatiquement créé pour effectuer la réplication des données vers l'espace sélectionné Régions AWS en votre nom.

  Pour plus d'informations sur les rôles liés à un service, consultez the section called “Utilisation des rôles liés à un service”.
- `AWS KMS`— Permet aux principaux d'afficher les clés configurées dansAWS KMS. Cela est nécessaire pour que les utilisateurs puissent consulter AWS KMS les clés qu'ils créent et gèrent dans leur compte afin de confirmer que la clé attribuée à Amazon Keyspaces est une clé de chiffrement symétrique activée.
- `Amazon EC2`— Permet aux principaux se connectant à Amazon Keyspaces via des points de terminaison VPC d'interroger le VPC de votre instance Amazon EC2 pour obtenir des informations sur le point de terminaison et l'interface réseau. Cet accès en lecture seule à l'instance Amazon EC2 est requis pour qu'Amazon Keyspaces puisse rechercher et stocker les points de terminaison VPC d'interface disponibles dans le tableau utilisé pour l'équilibrage de charge de connexion. `system.peers`

Pour consulter le `JSON` format de la politique, voir AmazonKeyspacesFullAccess.

Mises à jour des politiques gérées par Amazon Keyspaces AWS

Consultez les informations relatives aux mises à jour des politiques AWS gérées pour Amazon Keyspaces depuis que ce service a commencé à suivre ces modifications. Pour obtenir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS de la page Historique de la documentation.

Modification	Description	Date
AmazonKeyspacesFullAccess - mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations en lecture seule pour les clients se connectant à Amazon Keyspaces via des points de terminaison VPC d'interface afin d'accéder à l'instance Amazon EC2 afin de rechercher des informations réseau. Amazon Keyspaces stocke les points de terminaison VPC d'interface disponibles dans le `system.peers` tableau pour l'équilibrage de charge de connexion. Pour plus d'informations, veuillez consulter the section called “Utilisation des points de terminaison de VPC d'interface ”.	3 octobre 2023
AmazonKeyspacesReadOnlyAccess_v2 — Nouvelle politique	Amazon Keyspaces a créé une nouvelle politique visant à ajouter des autorisations en lecture seule pour les clients se connectant à Amazon Keyspaces via des points de terminaison VPC d'interface afin d'accéder à l'instance Amazon EC2 afin de rechercher des informations réseau.	12 septembre 2023

Modification	Description	Date
	Amazon Keyspaces stocke les points de terminaison VPC d'interface disponibles dans le `system.peers` tableau pour l'équilibrage de charge de connexion. Pour plus d'informations, veuillez consulter the section called “Utilisation des points de terminaison de VPC d'interface ”.	
AmazonKeyspacesFullAccess : mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations pour permettre à Amazon Keyspaces de créer un rôle lié à un service lorsqu'un administrateur crée un espace de touches multirégional. Amazon Keyspaces utilise le rôle lié à un service pour effectuer des tâches de réplication de données en votre nom. Pour plus d'informations, veuillez consulter the section called “Réplication multirégionale”.	5 juin 2023

Modification	Description	Date
AmazonKeyspacesReadOnlyAccess : mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations pour permettre aux utilisateurs de consulter la taille facturable d'une table en utilisant. CloudWatch Amazon Keyspaces s'intègre CloudWatch à Amazon pour vous permettre de contrôler la taille de la table facturable. Pour plus d'informations, veuillez consulter the section called “Statistiques et dimensions d'Amazon Keyspaces”.	7 juillet 2022
AmazonKeyspacesFullAccess : mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations pour permettre aux utilisateurs de consulter la taille facturable d'une table en utilisant. CloudWatch Amazon Keyspaces s'intègre CloudWatch à Amazon pour vous permettre de contrôler la taille de la table facturable. Pour plus d'informations, veuillez consulter the section called “Statistiques et dimensions d'Amazon Keyspaces”.	7 juillet 2022

Modification	Description	Date
AmazonKeyspacesReadOnlyAccess : mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations pour permettre aux utilisateurs de consulter AWS KMS les clés configurées pour le chiffrement Amazon Keyspaces au repos. Le chiffrement au repos d'Amazon Keyspaces s'intègre AWS KMS à la protection et à la gestion des clés de chiffrement utilisées pour chiffrer les données au repos. Pour afficher la AWS KMS clé configurée pour Amazon Keyspaces, des autorisations de lecture seule ont été ajoutées.	1er juin 2021

Modification	Description	Date
AmazonKeyspacesFullAccess : mise à jour d'une politique existante	Amazon Keyspaces a ajouté de nouvelles autorisations pour permettre aux utilisateurs de consulter AWS KMS les clés configurées pour le chiffrement Amazon Keyspaces au repos. Le chiffrement au repos d'Amazon Keyspaces s'intègre AWS KMS à la protection et à la gestion des clés de chiffrement utilisées pour chiffrer les données au repos. Pour afficher la AWS KMS clé configurée pour Amazon Keyspaces, des autorisations de lecture seule ont été ajoutées.	1er juin 2021
Amazon Keyspaces a commencé à suivre les modifications	Amazon Keyspaces a commencé à suivre les modifications apportées à ses politiques AWS gérées.	1er juin 2021

Résolution des problèmes d'identité et d'accès à Amazon Keyspaces

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon Keyspaces et IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Amazon Keyspaces
- J'ai modifié un utilisateur ou un rôle IAM et les modifications n'ont pas pris effet immédiatement

- Je ne parviens pas à restaurer une table à l'aide d'Amazon Keyspaces point-in-time Recovery (PITR)
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je suis administrateur et je souhaite autoriser d'autres personnes à accéder à Amazon Keyspaces
- Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources Amazon Keyspaces

Je ne suis pas autorisé à effectuer une action dans Amazon Keyspaces

S'il vous AWS Management Console indique que vous n'êtes pas autorisé à effectuer une action, vous devez contacter votre administrateur pour obtenir de l'aide. Votre administrateur est la personne qui vous a fourni votre nom d'utilisateur et votre mot de passe.

L'exemple d'erreur suivant se produit lorsque l'utilisateur `mateojackson` IAM essaie d'utiliser la console pour afficher les détails d'une *`table`* mais ne dispose pas des `cassandra:`*`Select`* autorisations nécessaires pour cette table.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cassandra:Select on resource: mytable
```

Dans ce cas, Mateo demande à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource *`mytable`* à l'aide de l'action `cassandra:`*`Select`*.

J'ai modifié un utilisateur ou un rôle IAM et les modifications n'ont pas pris effet immédiatement

Les modifications de la politique IAM peuvent prendre jusqu'à 10 minutes pour prendre effet pour les applications disposant de connexions existantes et établies à Amazon Keyspaces. Les modifications de politique IAM prennent effet immédiatement lorsque les applications établissent une nouvelle connexion. Si vous avez apporté des modifications à un utilisateur ou à un rôle IAM existant et que cela n'a pas pris effet immédiatement, attendez 10 minutes ou déconnectez-vous et reconnectez-vous à Amazon Keyspaces.

Je ne parviens pas à restaurer une table à l'aide d'Amazon Keyspaces point-in-time Recovery (PITR)

Si vous essayez de restaurer une table Amazon Keyspaces avec point-in-time restauration (PITR) et que le processus de restauration commence mais ne se termine pas correctement, vous n'avez peut-

être pas configuré toutes les autorisations requises pour le processus de restauration. Vous devez contacter votre administrateur pour obtenir de l'aide et lui demander de mettre à jour vos politiques afin de vous permettre de restaurer une table dans Amazon Keyspaces.

Outre les autorisations des utilisateurs, Amazon Keyspaces peut avoir besoin d'autorisations pour effectuer des actions au nom de votre principal pendant le processus de restauration. C'est le cas si la table est chiffrée à l'aide d'une clé gérée par le client ou si vous utilisez des politiques IAM qui limitent le trafic entrant. Par exemple, si vous utilisez des clés de condition dans votre politique IAM pour limiter le trafic source à des points de terminaison ou à des plages d'adresses IP spécifiques, l'opération de restauration échoue. Pour permettre à Amazon Keyspaces d'effectuer l'opération de restauration des tables pour le compte de votre principal, vous devez ajouter une clé de condition `aws:ViaAWSService` globale dans la politique IAM.

Pour plus d'informations sur les autorisations de restauration de tables, consultezthe section called “Restaurer les autorisations”.

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon Keyspaces.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans Amazon Keyspaces. Toutefois, l’action nécessite que le service ait des autorisations accordées par un rôle de service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d’exécuter l’action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d’identification de connexion.

Je suis administrateur et je souhaite autoriser d'autres personnes à accéder à Amazon Keyspaces

Pour autoriser d'autres personnes à accéder à Amazon Keyspaces, vous devez créer une entité IAM (utilisateur ou rôle) pour la personne ou l'application qui a besoin d'y accéder. Ils utiliseront les informations d'identification de cette entité pour accéder à AWS. Vous devez ensuite joindre une politique à l'entité qui lui accorde les autorisations appropriées dans Amazon Keyspaces.

Pour démarrer immédiatement, consultez Création de votre premier groupe et utilisateur délégué IAM dans le Guide de l'utilisateur IAM.

Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources Amazon Keyspaces

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, consultez les éléments suivants :

- Pour savoir si Amazon Keyspaces prend en charge ces fonctionnalités, consultez. Comment Amazon Keyspaces fonctionne avec IAM
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour connaître la différence entre l'utilisation de rôles et de politiques basées sur les ressources pour l'accès entre comptes, consultez la section Accès aux ressources entre comptes dans IAM dans le guide de l'utilisateur d'IAM.

Utilisation de rôles liés à un service pour Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) utilise des rôles liés à un service AWS Identity and Access Management (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon Keyspaces. Les rôles liés à un service sont prédéfinis par Amazon Keyspaces et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Rubriques

- Utilisation des rôles pour le dimensionnement automatique des applications Amazon Keyspaces
- Utilisation des rôles pour la réplication multirégionale d'Amazon Keyspaces

Utilisation des rôles pour le dimensionnement automatique des applications Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) utilise des rôles liés à un service AWS Identity and Access Management (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon Keyspaces. Les rôles liés à un service sont prédéfinis par Amazon Keyspaces et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration d'Amazon Keyspaces, car vous n'avez pas à ajouter manuellement les autorisations nécessaires. Amazon Keyspaces définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul Amazon Keyspaces peut assumer ses rôles. Les autorisations définies comprennent la politique d’approbation et la politique d’autorisation. De plus, cette politique d’autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Cela protège vos ressources Amazon Keyspaces, car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Pour plus d'informations sur les autres services prenant en charge les rôles liés à un service, consultez les AWS services opérationnels avec IAM et recherchez les services présentant la mention Yes (Oui) dans la colonne Service-linked roles (Rôles liés à un service). Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations de rôle liées à un service pour Amazon Keyspaces

Amazon Keyspaces utilise le rôle lié au service nommé AWSServiceRoleForApplicationAutoScaling_CassandraTablepour permettre à Application Auto Scaling d'appeler Amazon Keyspaces et Amazon en votre nom. CloudWatch

Le rôle AWSServiceRoleForApplicationAutoScaling_CassandraTable lié à un service fait confiance aux services suivants pour assumer le rôle :

- `cassandra.application-autoscaling.amazonaws.com`

La politique d'autorisation des rôles permet à Application Auto Scaling d'effectuer les actions suivantes sur les ressources Amazon Keyspaces spécifiées :

- Action : `cassandra:Select` sur `arn:*:cassandra:*:*:/keyspace/system/table/*`
- Action : `cassandra:Select` sur la ressource `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
- Action : `cassandra:Select` sur la ressource `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Action : `cassandra:Alter` sur la ressource `arn:*:cassandra:*:*:"*"`

Création d'un rôle lié à un service pour Amazon Keyspaces

Il n'est pas nécessaire de créer manuellement un rôle lié à un service pour le dimensionnement automatique d'Amazon Keyspaces. Lorsque vous activez le dimensionnement automatique d'Amazon Keyspaces sur une table avec le CQLAWS Management Console, le ou l'AWSAPIAWS CLI, Application Auto Scaling crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous activez le dimensionnement automatique d'Amazon Keyspaces pour une table, Application Auto Scaling crée à nouveau le rôle lié à un service pour vous.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Pour en savoir plus, consultez la section Un nouveau rôle est apparu dans mon compte Compte AWS.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous activez le

dimensionnement automatique des applications Amazon Keyspaces pour une table, Application Auto Scaling crée à nouveau le rôle lié à un service pour vous.

Modification d'un rôle lié à un service pour Amazon Keyspaces

Amazon Keyspaces ne vous permet pas de modifier le rôle lié au AWSServiceRoleForApplicationAutoScaling_CassandraTable service. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Supprimer un rôle lié à un service pour Amazon Keyspaces

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Toutefois, vous devez d'abord désactiver le dimensionnement automatique sur toutes les tables du compte Régions AWS avant de pouvoir supprimer manuellement le rôle lié à un service. Pour désactiver le dimensionnement automatique sur les tables Amazon Keyspaces, consultez Modifier ou désactiver les paramètres de dimensionnement automatique d'Amazon Keyspaces.

> **Note**
>
> Si le dimensionnement automatique d'Amazon Keyspaces utilise le rôle lorsque vous essayez de modifier les ressources, la désinscription risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM, leAWS CLI, ou l'AWSAPI pour supprimer le rôle lié au AWSServiceRoleForApplicationAutoScaling_CassandraTable service. Pour plus d'informations, veuillez consulter Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

> **Note**
>
> Pour supprimer le rôle lié au service utilisé par le dimensionnement automatique d'Amazon Keyspaces, vous devez d'abord désactiver le dimensionnement automatique sur toutes les tables du compte.

Régions prises en charge pour les rôles liés au service Amazon Keyspaces

Amazon Keyspaces prend en charge l'utilisation de rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez la section Points de terminaison de service pour Amazon Keyspaces.

Utilisation des rôles pour la réplication multirégionale d'Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) utilise des rôles liés à un service AWS Identity and Access Management (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon Keyspaces. Les rôles liés à un service sont prédéfinis par Amazon Keyspaces et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration d'Amazon Keyspaces, car vous n'avez pas à ajouter manuellement les autorisations nécessaires. Amazon Keyspaces définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul Amazon Keyspaces peut assumer ses rôles. Les autorisations définies comprennent la politique d’approbation et la politique d’autorisation. De plus, cette politique d’autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Cela protège vos ressources Amazon Keyspaces, car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Pour plus d'informations sur les autres services prenant en charge les rôles liés à un service, consultez les AWS services opérationnels avec IAM et recherchez les services présentant la mention Yes (Oui) dans la colonne Service-linked roles (Rôles liés à un service). Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations de rôle liées à un service pour Amazon Keyspaces

Amazon Keyspaces utilise le rôle lié au service nommé pour permettre à AWSServiceRoleForAmazonKeyspacesReplicationAmazon Keyspaces de répliquer les écritures sur toutes les répliques d'une table multirégionale en votre nom.

Le rôle AWSServiceRoleForAmazonKeyspacesReplication lié à un service fait confiance aux services suivants pour assumer le rôle :

- `replication.cassandra.amazonaws.com`

La politique d'autorisation des rôles nommée KeyspacesReplicationServiceRolePolicy permet à Amazon Keyspaces d'effectuer les actions suivantes :

- Action : `cassandra:Select`
- Action : `cassandra:SelectMultiRegionResource`
- Action : `cassandra:Modify`
- Action : `cassandra:ModifyMultiRegionResource`

Bien que le rôle lié au service Amazon Keyspaces AWSServiceRoleForAmazonKeyspacesReplication fournisse les autorisations suivantes : « Action : » pour le nom de ressource Amazon (ARN) « arn : * » spécifié dans la politique, Amazon Keyspaces fournit l'ARN de votre compte.

Vous devez configurer les autorisations de manière à permettre à vos utilisateurs, groupes ou rôles de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon Keyspaces

Vous ne pouvez pas créer manuellement un rôle lié à un service. Lorsque vous créez un espace de saisie multirégional dans l'APIAWS Management Console, le ou l'AWSAPIAWS CLI, Amazon Keyspaces crée le rôle lié à un service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un espace de saisie multirégional, Amazon Keyspaces crée à nouveau le rôle lié à un service pour vous.

Modification d'un rôle lié à un service pour Amazon Keyspaces

Amazon Keyspaces ne vous permet pas de modifier le rôle lié au AWSServiceRoleForAmazonKeyspacesReplication service. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Supprimer un rôle lié à un service pour Amazon Keyspaces

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée

qui n’est pas surveillée ou gérée activement. Toutefois, vous devez d'abord supprimer tous les espaces clés multirégionaux du compte Régions AWS avant de pouvoir supprimer manuellement le rôle lié à un service.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord supprimer les espaces clés et les tables multirégionaux utilisés par le rôle.

> **Note**
>
> Si le service Amazon Keyspaces utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon Keyspaces utilisées par AWSServiceRoleForAmazonKeyspacesReplication (console)

1. Connectez-vous à la AWS Management Console console Amazon Keyspaces et ouvrez-la à l'adresse https://console.aws.amazon.com/keyspaces/home.
2. Choisissez Keyspaces dans le panneau de gauche.
3. Sélectionnez tous les espaces clés multirégionaux dans la liste.
4. Choisissez Supprimer, confirmez la suppression, puis sélectionnez Supprimer les espaces clés.

Vous pouvez également supprimer des espaces clés multirégionaux par programmation à l'aide de l'une des méthodes suivantes.

- L'instruction Cassandra Query Language (CQL). ???
- L'opération delete-keyspace de la CLI. AWS
- DeleteKeyspaceFonctionnement de l'API Amazon Keyspaces.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, l'AWS CLI ou l'API AWS pour supprimer le rôle lié à un service AWSServiceRoleForAmazonKeyspacesReplication. Pour plus d’informations, consultez Suppression d’un rôle lié à un service dans le Guide de l’utilisateur IAM.

Régions prises en charge pour les rôles liés au service Amazon Keyspaces

Amazon Keyspaces ne prend pas en charge l'utilisation de rôles liés à un service dans toutes les régions où le service est disponible. Vous pouvez utiliser le AWSServiceRoleForAmazonKeyspacesReplication rôle dans les régions suivantes.

Nom de la région	Identité de la région	Support sur Amazon Keyspaces
US East (Virginie du Nord)	us-east-1	Oui
USA Est (Ohio)	us-east-2	Oui
USA Ouest (Californie du Nord)	us-west-1	Oui
USA Ouest (Oregon)	us-west-2	Oui
Asie-Pacifique (Mumbai)	ap-south-1	Oui
Asie-Pacifique (Osaka)	ap-northeast-3	Oui
Asie-Pacifique (Séoul)	ap-northeast-2	Oui
Asie-Pacifique (Singapour)	ap-southeast-1	Oui
Asie-Pacifique (Sydney)	ap-southeast-2	Oui
Asie-Pacifique (Tokyo)	ap-northeast-1	Oui
Canada (Centre)	ca-central-1	Oui
Europe (Francfort)	eu-central-1	Oui
Europe (Irlande)	eu-west-1	Oui
Europe (Londres)	eu-west-2	Oui
Europe (Paris)	eu-west-3	Oui
Amérique du Sud (São Paulo)	sa-east-1	Oui
AWS GovCloud (USA Est)	us-gov-east-1	Non

Nom de la région	Identité de la région	Support sur Amazon Keyspaces
AWS GovCloud (US-Ouest)	us-gov-west-1	Non

Validation de conformité pour Amazon Keyspaces (pour Apache Cassandra)

Des auditeurs tiers évaluent la sécurité et la conformité d'Amazon Keyspaces (pour Apache Cassandra) dans le cadre de plusieurs AWS programmes de conformité. Il s'agit des licences suivantes :

- ISO/IEC 27001:2013, 27017:2015, 27018:2019et ISO/IEC 9001:2015. Pour plus d'informations, consultez les certifications et services AWS ISO et CSA STAR.
- System and Organization Controls (SOC)
- Payment Card Industry (PCI)
- Programme fédéral de gestion des risques et des autorisations (FedRAMP) élevé
- Health Insurance Portability and Accountability Act (HIPAA)

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur AWS la sécurité et la conformité.

- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : ce livre blanc décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.

> **Note**
>
> Tous ne Services AWS sont pas éligibles à la loi HIPAA. Pour plus d'informations, consultez le HIPAA Eligible Services Reference.

- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Guides de conformité destinés aux clients — Comprenez le modèle de responsabilité partagée sous l'angle de la conformité. Les guides résument les meilleures pratiques en matière de sécurisation Services AWS et décrivent les directives relatives aux contrôles de sécurité dans de nombreux cadres (notamment le National Institute of Standards and Technology (NIST), le Payment Card Industry Security Standards Council (PCI) et l'Organisation internationale de normalisation (ISO)).
- Évaluation des ressources à l'aide des règles du guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.
- Amazon GuardDuty — Cela Service AWS détecte les menaces potentielles qui pèsent sur vos charges de travail Comptes AWS, vos conteneurs et vos données en surveillant votre environnement pour détecter toute activité suspecte et malveillante. GuardDuty peut vous aider à répondre à diverses exigences de conformité, telles que la norme PCI DSS, en répondant aux exigences de détection des intrusions imposées par certains cadres de conformité.
- AWS Audit Manager— Cela vous Service AWS permet d'auditer en permanence votre AWS utilisation afin de simplifier la gestion des risques et la conformité aux réglementations et aux normes du secteur.

Résilience et reprise après sinistre dans Amazon Keyspaces

L'infrastructure mondiale d'AWS est construite autour de zones de disponibilité et de Régions AWS. Les Régions AWSfournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à débit élevé et à forte redondance. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone de disponibilité à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Amazon Keyspaces réplique automatiquement les données à trois reprises dans plusieurs zones de AWS disponibilité au sein de la même zone Région AWS pour garantir la durabilité et la haute disponibilité.

Pour plus d'informations sur les Régions AWS et les zones de disponibilité, consultez Infrastructure mondiale d'AWS.

Outre l'infrastructure AWS mondiale d', Amazon Keyspaces offre plusieurs fonctions qui contribuent à la prise en charge des besoins en matière de résilience et de sauvegarde de données.

Réplication multirégion

> Amazon Keyspaces permet la réplication multirégion si vous avez besoin de répliquer vos données ou applications sur distances. Vous pouvez répliquer vos tableaux Amazon Keyspaces sur un maximum de six tableaux différents Régions AWS de votre choix. Pour plus d'informations, veuillez consulter Réplication multirégionale.

oint-in-timeRécupération du phosphore (PITR)

> PITR permet de protéger vos tables Amazon Keyspaces contre les opérations d'écriture ou de suppression en continu de données de tables. Pour plus d'informations, consultez P oint-in-time recovery pour Amazon Keyspaces.

Sécurité de l'infrastructure dans Amazon Keyspaces

En tant que service géré, Amazon Keyspaces (pour Apache Cassandra) est protégé par un système de sécurité réseau AWS mondial. Pour plus d'informations sur les services de sécurité AWS et la manière dont AWS protège l'infrastructure, consultez la section Sécurité du cloud AWS. Pour concevoir votre environnement AWS en utilisant les meilleures pratiques en matière de sécurité de

l'infrastructure, consultez la section Protection de l'infrastructure dans le Security Pillar AWS Well-Architected Framework (Pilier de sécurité de l'infrastructure Well-Architected Framework).

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon Keyspaces via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et nous recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Amazon Keyspaces prend en charge deux méthodes d'authentification des demandes des clients. La première méthode utilise des informations d'identification spécifiques au service, qui sont des informations d'identification basées sur un mot de passe générées pour un utilisateur IAM spécifique. Vous pouvez créer et gérer le mot de passe à l'aide de la console IAMAWS CLI, de ou de l'AWSAPI. Pour plus d'informations, consultez Utilisation d'IAM avec Amazon Keyspaces.

La seconde méthode utilise un plugin d'authentification pour le pilote DataStax Java open source pour Cassandra. Ce plug-in permet aux utilisateurs, aux rôles et aux identités fédérées IAM d'ajouter des informations d'authentification aux demandes d'API Amazon Keyspaces (pour Apache Cassandra) à l'aide du processus AWS Signature Version 4 (SigV4). Pour plus d'informations, veuillez consulter the section called “Informations d'identification IAM pour l' AWS authentification”.

Vous pouvez utiliser un point de terminaison d'interface VPC pour empêcher le trafic entre votre Amazon VPC et Amazon Keyspaces de quitter le réseau Amazon. Les points de terminaison d'interface VPC sont alimentés parAWS PrivateLink, une AWS technologie qui permet une communication privée entre les AWS services à l'aide d'une interface réseau élastique avec des adresses IP privées dans votre Amazon VPC. Pour plus d'informations, veuillez consulter the section called “Utilisation des points de terminaison de VPC d'interface ”.

Utilisation d'Amazon Keyspaces avec des points de terminaison VPC d'interface

Les points de terminaison VPC d'interface permettent une communication privée entre votre cloud privé virtuel (VPC) exécuté dans Amazon VPC et Amazon Keyspaces. Les points de terminaison VPC d'interface sont alimentés par AWS PrivateLink un AWS service qui permet une communication privée entre les VPC et les services. AWS

AWS PrivateLink permet cela en utilisant une interface réseau élastique avec des adresses IP privées dans votre VPC afin que le trafic réseau ne quitte pas le réseau Amazon. Les points de terminaison de VPC d'interface n'ont pas besoin d'une passerelle Internet, d'un périphérique NAT, d'une connexion VPN ou d'une connexion AWS Direct Connect . Pour plus d'informations, consultez Amazon Virtual Private Cloud and Interface VPC endpoints ().AWS PrivateLink

Rubriques

- Utilisation des points de terminaison VPC de l'interface pour Amazon Keyspaces
- Remplissage des entrées de system.peers table avec les informations de point de terminaison VPC de l'interface
- Contrôle de l'accès aux points de terminaison VPC de l'interface pour Amazon Keyspaces
- Disponibilité
- Politiques relatives aux terminaux VPC et restauration d'Amazon Keyspaces point-in-time (PITR)
- Erreurs et avertissements courants

Utilisation des points de terminaison VPC de l'interface pour Amazon Keyspaces

Vous pouvez créer un point de terminaison VPC d'interface afin que le trafic entre Amazon Keyspaces et vos ressources Amazon VPC commence à passer par le point de terminaison VPC d'interface. Pour commencer, suivez les étapes pour créer un point de terminaison d'interface. Ensuite, modifiez le groupe de sécurité associé au point de terminaison que vous avez créé à l'étape précédente et configurez une règle entrante pour le port 9142. Pour plus d'informations, consultez la section Ajout, suppression et mise à jour de règles.

Pour un step-by-step didacticiel sur la configuration d'une connexion à Amazon Keyspaces via un point de terminaison VPC, consultez. the section called “Connexion aux points de terminaison VPC” Pour savoir comment configurer l'accès entre comptes pour les ressources Amazon Keyspaces

séparément des applications présentes dans Comptes AWS un VPC, consultez. the section called “Accès inter-comptes”

Remplissage des entrées de **`system.peers`** table avec les informations de point de terminaison VPC de l'interface

Les pilotes Apache Cassandra utilisent la `system.peers` table pour demander des informations sur les nœuds du cluster. Les pilotes Cassandra utilisent les informations du nœud pour équilibrer la charge des connexions et relancer les opérations. Amazon Keyspaces remplit automatiquement neuf entrées du `system.peers` tableau pour les clients qui se connectent via le point de terminaison public.

Pour fournir aux clients qui se connectent via des points de terminaison VPC d'interface des fonctionnalités similaires, Amazon Keyspaces remplit le `system.peers` tableau de votre compte avec une entrée pour chaque zone de disponibilité dans laquelle un point de terminaison VPC est disponible. Pour rechercher et stocker les points de terminaison VPC d'interface disponibles dans le tableau`system.peers`, Amazon Keyspaces exige que vous accordiez à l'entité IAM utilisée pour se connecter à Amazon Keyspaces les autorisations d'accès lui permettant de demander à votre VPC les informations relatives au point de terminaison et à l'interface réseau.

> **Important**
>
> Le remplissage du `system.peers` tableau avec les points de terminaison VPC de votre interface disponibles améliore l'équilibrage de charge et augmente le débit de lecture/écriture. Il est recommandé à tous les clients accédant à Amazon Keyspaces via des points de terminaison VPC d'interface et est requis pour Apache Spark.

Pour accorder à l'entité IAM utilisée pour se connecter à Amazon Keyspaces l'autorisation de rechercher les informations nécessaires sur le point de terminaison VPC de l'interface, vous pouvez mettre à jour votre rôle IAM ou votre politique utilisateur existants, ou créer une nouvelle politique IAM comme indiqué dans l'exemple suivant.

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

```
            "Action":[
               "ec2:DescribeNetworkInterfaces",
               "ec2:DescribeVpcEndpoints"
            ],
            "Resource":"*"
         }
      ]
}
```

> **Note**
>
> Les politiques gérées `AmazonKeyspacesFullAccess` incluent `AmazonKeyspacesReadOnlyAccess_v2` les autorisations requises pour permettre à Amazon Keyspaces d'accéder à l'instance Amazon EC2 afin de lire les informations sur les points de terminaison VPC d'interface disponibles.

Pour vérifier que la politique a été correctement configurée, interrogez le `system.peers` tableau pour consulter les informations réseau. Si le `system.peers` tableau est vide, cela peut indiquer que la politique n'a pas été configurée correctement ou que vous avez dépassé le quota de demandes `DescribeNetworkInterfaces` et les actions `DescribeVPCEndpoints` d'API. `DescribeVPCEndpoints`entre dans `Describe*` cette catégorie et est considérée comme une action non mutante. `DescribeNetworkInterfaces`appartient au sous-ensemble des actions non mutantes non filtrées et non paginées, et différents quotas s'appliquent. Pour plus d'informations, consultez la section Taille des compartiments de jetons de demande et taux de recharge dans le manuel Amazon EC2 API Reference.

Si vous voyez un tableau vide, réessayez quelques minutes plus tard pour exclure tout problème de quota de taux de demandes. Pour vérifier que vous avez correctement configuré les points de terminaison VPC, consultez. the section called “Erreurs de connexion au point de terminaison VPC” Si votre requête renvoie les résultats du tableau, cela signifie que votre politique a été correctement configurée.

Contrôle de l'accès aux points de terminaison VPC de l'interface pour Amazon Keyspaces

Avec les politiques de point de terminaison VPC, vous pouvez contrôler l'accès aux ressources de deux manières :

- Politique IAM : vous pouvez contrôler les demandes, les utilisateurs ou les groupes autorisés à accéder à Amazon Keyspaces via un point de terminaison VPC spécifique. Pour ce faire, utilisez une clé de condition dans la stratégie attachée à un utilisateur, un groupe ou un rôle IAM.
- Politique VPC : vous pouvez contrôler quels points de terminaison VPC ont accès à vos ressources Amazon Keyspaces en leur associant des politiques. Pour restreindre l'accès à un keyspace ou à une table spécifique afin d'autoriser uniquement le trafic passant par un point de terminaison VPC spécifique, modifiez la stratégie IAM existante qui restreint l'accès aux ressources et ajoutez ce point de terminaison VPC.

Vous trouverez ci-dessous des exemples de politiques relatives aux points de terminaison permettant d'accéder aux ressources Amazon Keyspaces.

- Exemple de politique IAM : limitez tout accès à une table Amazon Keyspaces spécifique, sauf si le trafic provient du point de terminaison VPC spécifié. Cet exemple de politique peut être associé à un utilisateur, un rôle ou un groupe IAM. Il restreint l'accès à une table Amazon Keyspaces spécifiée, sauf si le trafic entrant provient d'un point de terminaison VPC spécifié.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "UserOrRolePolicyToDenyAccess",
         "Action": "cassandra:*",
         "Effect": "Deny",
         "Resource": [
                         "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable",
                         "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
           ],
         "Condition": { "StringNotEquals" : { "aws:sourceVpce": "vpce-abc123" } }
      }
   ]
}
```

> **Note**
>
> Pour restreindre l'accès à une table spécifique, vous devez également inclure l'accès aux tables système. Les tables système sont en lecture seule.

- Exemple de politique VPC : accès en lecture seule — Cet exemple de politique peut être attaché à un point de terminaison VPC. (Pour plus d'informations, consultez la section Contrôle de l'accès aux ressources Amazon VPC). Il limite les actions à l'accès en lecture seule aux ressources Amazon Keyspaces via le point de terminaison VPC auquel il est connecté.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadOnly",
      "Principal": "*",
      "Action": [
        "cassandra:Select"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- Exemple de politique VPC : restreindre l'accès à une table Amazon Keyspaces spécifique — Cet exemple de politique peut être attaché à un point de terminaison VPC. Il limite l'accès à une table spécifique via le point de terminaison VPC auquel il est attaché.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "RestrictAccessToTable",
            "Principal": "*",
            "Action": "cassandra:*",
            "Effect": "Allow",
            "Resource": [
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable",
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
           ]
        }
   ]
}
```

> **Note**
>
> Pour restreindre l'accès à une table spécifique, vous devez également inclure l'accès aux tables système. Les tables système sont en lecture seule.

Disponibilité

Amazon Keyspaces prend en charge l'utilisation des points de terminaison VPC d'interface partout Régions AWS où le service est disponible. Pour plus d'informations, consultez ???.

Politiques relatives aux terminaux VPC et restauration d'Amazon Keyspaces point-in-time (PITR)

Si vous utilisez des politiques IAM avec des clés de condition pour restreindre le trafic entrant, l'opération de restauration des tables risque d'échouer. Par exemple, si vous limitez le trafic source à des points de terminaison VPC spécifiques à l'aide de clés de `aws:SourceVpce` condition, l'opération de restauration de table échoue. Pour autoriser Amazon Keyspaces à effectuer une opération de restauration pour le compte de votre mandant, vous devez ajouter une clé de `aws:ViaAWSService` condition à votre politique IAM. La clé de `aws:ViaAWSService` condition permet l'accès lorsqu'un AWS service fait une demande en utilisant les informations d'identification du principal. Pour plus d'informations, voir Éléments de politique IAM JSON : clé de condition dans le guide de l'utilisateur IAM. La politique suivante en est un exemple.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"CassandraAccessForVPCE",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"false"
            },
            "StringEquals":{
               "aws:SourceVpce":[
                  "vpce-12345678901234567"
```

```
                ]
            }
         }
      },
      {
         "Sid":"CassandraAccessForAwsService",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"true"
            }
         }
      }
   ]
}
```

Erreurs et avertissements courants

Si vous utilisez Amazon Virtual Private Cloud et que vous vous connectez à Amazon Keyspaces, l'avertissement suivant peut s'afficher.

```
Control node cassandra.us-east-1.amazonaws.com/1.111.111.111:9142 has an entry
 for itself in system.peers: this entry will be ignored. This is likely due to a
 misconfiguration;
please verify your rpc_address configuration in cassandra.yaml on all nodes in your
 cluster.
```

Cet avertissement se produit car le `system.peers` tableau contient des entrées pour tous les points de terminaison Amazon VPC qu'Amazon Keyspaces est autorisé à consulter, y compris le point de terminaison Amazon VPC via lequel vous êtes connecté. Vous pouvez ignorer cet avertissement en toute sécurité.

Pour les autres erreurs, consultezthe section called “Erreurs de connexion au point de terminaison VPC”.

Configuration Keyspaces

AWS gère les tâches de sécurité de base comme les correctifs du système d'exploitation invité et de base de données, la configuration du pare-feu et la reprise après sinistre. Ces procédures ont

été vérifiées et certifiées par les tiers appropriés. Pour de plus amples informations, consultez les ressources suivantes.

- Modèle de responsabilité partagée Modèle
- Amazon Web Services : présentation des procédures de sécurité (livre blanc)

Bonnes pratiques de sécurité pour Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) fournit différentes fonctions de sécurité à prendre en compte lorsque vous développez et implémentez vos propres stratégies de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des remarques utiles plutôt que comme des recommandations.

Rubriques

- Bonnes pratiques de sécurité préventive pour Amazon Keyspaces
- Bonnes pratiques de sécurité de détection pour Amazon Keyspaces

Bonnes pratiques de sécurité préventive pour Amazon Keyspaces

Les bonnes pratiques de sécurité suivantes sont considérées comme préventives, car elles peuvent vous aider à anticiper et à prévenir les incidents de sécurité dans Amazon Keyspaces.

Utiliser le chiffrement au repos

Amazon Keyspaces chiffre au repos toutes les données utilisateur stockées dans des tables à l'aide de clés de chiffrement stockées dans AWS Key Management Service(AWS KMS). Cela fournit une couche supplémentaire de protection des données en sécurisant vos données contre tout accès non autorisé au stockage sous-jacent.

Par défaut, Amazon Keyspaces utilise unClé détenue par AWS pour crypter toutes vos tables. Si cette clé n'existe pas, elle est créée pour vous. Les clés par défaut du service ne peuvent pas être désactivées.

Vous pouvez également utiliser une clé gérée par le client pour le chiffrement au repos. Pour en savoir plus, consultez Amazon Keyspaces Encryption at Rest.

Utiliser Des rôles IAM pour authentifier l'accès à Amazon Keyspaces

Pour que les utilisateurs, les applications et les autresAWS services puissent accéder à Amazon Keyspaces, ils doivent inclure desAWS informations d'identification valides dans leurs demandes d'AWSAPI. Vous ne devez pas stocker d'informations d'identification AWS directement dans l'application ou dans une instance EC2. Il s'agit d'informations d'identification à long terme qui ne font pas l'objet d'une rotation automatique, et dont la compromission pourrait avoir un impact considérable sur l'activité. Un rôle IAM vous permet d'obtenir des clés d'accès temporaires qui peuvent être utilisées pour accéder aux ressources et services AWS.

Pour en savoir plus, consultez Rôles IAM.

Utiliser les politiques IAM pour l'autorisation de base Amazon Keyspaces

Lorsque vous accordez des autorisations, vous décidez qui les reçoit, les API Amazon Keyspaces auxquelles elles s'appliquent, et les actions spécifiques que vous souhaitez autoriser sur ces ressources. L'implémentation d'un privilège minimum est la clé de la réduction des risques de sécurité et de l'impact potentiel d'erreurs ou d'actes de malveillance.

Attachez des politiques d'autorisations à des identités IAM (autrement dit, des utilisateurs, des groupes et des rôles), et accordez ainsi des autorisations pour effectuer des opérations sur des ressources Amazon Keyspaces.

Pour ce faire, utilisez les ressources suivantes :

- AWSPolitiques gérées (prédéfinies)
- Politiques gérées par le client

Utiliser des conditions de politique IAM pour un contrôle d'accès précis

Lorsque vous accordez des autorisations dans Amazon Keyspaces, vous pouvez spécifier des conditions pour déterminer comment une politique d'autorisation doit prendre effet. L'implémentation d'un privilège minimum est la clé de la réduction des risques de sécurité et de l'impact potentiel d'erreurs ou d'actes de malveillance.

Vous pouvez spécifier des conditions lors de l'octroi d'autorisations à l'aide d'une politique IAM. Par exemple, vous pouvez effectuer les opérations suivantes :

- Accordez des autorisations pour permettre aux utilisateurs d'accéder en lecture seule à des espaces clés ou des tables spécifiques.
- Accordez des autorisations pour permettre à un utilisateur d'accéder en écriture à une table donnée, en fonction de son identité.

Pour en savoir plus, consultez Exemples de stratégies basées sur l'identité.

Envisager un chiffrement côté client

Si vous stockez des données sensibles ou confidentielles dans Amazon Keyspaces, il se peut que vous deviez les chiffrer aussi près que possible de leur point de génération, afin qu'elles soient protégées tout au long de leur cycle de vie. Le chiffrement de vos données sensibles en transit et au repos contribue à garantir que vos données en texte brut ne sont pas accessibles à un tiers.

Bonnes pratiques de sécurité de détection pour Amazon Keyspaces

Les bonnes pratiques de sécurité suivantes sont considérées comme de détection, car elles peuvent vous aider à détecter les vulnérabilités et les incidents de sécurité potentiels.

AWS CloudTrailÀ utiliser pour surveiller l'utilisation deAWS KMS la toucheAWS Key Management Service (AWS KMS)

Si vous utilisez une AWS KMSclé gérée par le client pour le chiffrement au repos, l'utilisation de cette clé est journalisée dansAWS CloudTrail. CloudTrail offre une visibilité de l'activité utilisateur en enregistrant les actions effectuées sur votre compte. CloudTrail enregistre des informations importantes sur chaque action, dont qui a fait la demande, les services utilisés, les actions réalisées, les paramètres pour les actions et les éléments de réponse renvoyés par leAWS service. Ces informations vous aident à suivre les modifications apportées à vosAWS ressources et à résoudre des problèmes opérationnels. CloudTrail facilite la mise en conformité avec des politiques internes et des normes réglementaires.

Vous pouvez utiliser CloudTrail pour vérifier l'utilisation de la clé. CloudTrail crée des fichiers journaux contenant un historique des appels d'AWSAPI et des événements associés pour votre compte. Ces fichiers journaux incluent toutes les demandes d'AWS KMSAPI qui ont été effectuées avec l'interface,AWS les kits SDK et les outils de ligne de commande, ainsi que viaAWS des services intégrés. Vous pouvez utiliser ces fichiers journaux pour obtenir des informations sur l'utilisation de laAWS KMS clé, sur l'opération qui a été demandée, sur l'identité du demandeur, sur l'adresse IP à partir de laquelle la demande provenait, sur l'adresse IP à partir de laquelle la demande provenait, sur l'adresse IP à partir de laquelle la demande provenait, sur l'adresse IP à partir de laquelle la demande provenait, sur l' Pour plus d'informations, consultez Journalisation des appels d'API AWS Key Management Service avec AWS CloudTrail dans le Guide de l'utilisateur AWS CloudTrail.

Utilisée CloudTrail pour surveiller les opérations en langage de définition de données (DDL) d'Amazon Keyspaces

CloudTrail offre une visibilité de l'activité utilisateur en enregistrant les actions effectuées sur votre compte. CloudTrail enregistre des informations importantes sur chaque action, dont qui a fait la demande, les services utilisés, les actions réalisées, les paramètres pour les actions et les éléments de réponse renvoyés par leAWS service. Ces informations vous aident à suivre les modifications apportées à vosAWS ressources et à résoudre des problèmes opérationnels. CloudTrail facilite la mise en conformité avec des politiques internes et des normes réglementaires.

Toutes les opérations DDL d'Amazon Keyspaces sont CloudTrail automatiquement connectées. Les opérations DDL vous permettent de créer et gérer des espaces clés et des tables Amazon Keyspaces.

Lorsqu'une activité se produit dans Amazon Keyspaces, elle est enregistrée dans un CloudTrail événement avec d'autres événements deAWS service dans l'historique des événements. Pour plus d'informations, consultez la section Journalisation des opérations Amazon Keyspaces à l'aide deAWS CloudTrail. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements dans le Guide deAWS CloudTrail l'utilisateur.

Pour un enregistrement continu des événements dans votreCompte AWS, y compris les événements pour Amazon Keyspaces, créez un journal d'activité. Un journal d'activité CloudTrail permet de livrer des fichiers journaux à un compartiment Amazon Simple Storage Service (Amazon S3). Par défaut, lorsque vous créez un journal d'activité sur la console, il s'applique à tousRégions AWS. Le journal d'activité consigne les événements de toutes les régions dans la partition AWS et livre les fichiers journaux dans le compartiment S3 de votre choix. En outre, vous pouvez configurer d'autresAWS services pour analyser en profondeur les données d'événement collectées dans les CloudTrail journaux et agir sur celles-ci.

Etiqueter vos ressources Amazon Keyspaces pour l'identification et l'automatisation

Vous pouvez attribuer des métadonnées à vos ressources AWS sous la forme d'identifications. Chaque étiquette est un libellé composé d'une clé définie par le client et d'une valeur facultative qui peut faciliter la gestion, la recherche et le filtrage de ressources.

L'étiquetage permet l'implémentation de contrôles groupés. Bien qu'il n'existe pas de types intrinsèques d'étiquettes, celles-ci vous permettent de catégoriser des ressources par objectif, par propriétaire, par environnement ou selon d'autres critères. Voici quelques exemples :

- Accès — Utilisée pour contrôler l'accès aux ressources Amazon Keyspaces en fonction des balises. Pour plus d'informations, veuillez consulter the section called “Autorisation basée sur les tags Amazon Keyspaces”.
- Sécurité — Utilisée pour déterminer des exigences telles que les paramètres de protection des données.
- Confidentialité — Identifiant pour le niveau spécifique de confidentialité des données qu'une ressource prend en charge.
- Environnement – Utilisé pour différencier les infrastructures de développement, de test et de production.

Pour plus d'informations, consultez les AWSsections Stratégies de balisage et Ajout de balises et d'étiquettes aux ressources.

Référence du langage CQL pour Amazon Keyspaces (pour Apache Cassandra)

Après vous être connecté à un point de terminaison Amazon Keyspaces (pour Apache Cassandra), vous utilisez le Cassandra Query Language (CQL) pour travailler avec votre base de données. CQL est similaire à de nombreux égards au langage SQL (Structured Query Language).

Rubriques

Éléments du langage de requête Cassandra (CQL) dans Amazon Keyspaces

Découvrez les éléments du langage de requête Cassandra (CQL) pris en charge par Amazon Keyspaces, notamment les identifiants, les constantes, les termes et les types de données.

Rubriques

Identifiants

Les identificateurs (ou noms) sont utilisés pour identifier les tables, colonnes et autres objets. Un identifiant peut être entre apostrophes ou pas. Les conditions suivantes s'appliquent :

```
identifier          ::=  unquoted_identifier | quoted_identifier
```

```
unquoted_identifier ::=  re('[a-zA-Z][a-zA-Z0-9_]*')
quoted_identifier   ::=  '"' (any character where " can appear if doubled)+ '"'
```

Constantes

Les constantes suivantes sont définies.

```
constant ::=  string | integer | float | boolean | uuid | blob | NULL
string   ::=  '\'' (any character where ' can appear if doubled)+ '\''
              '$$' (any character other than '$$') '$$'
integer  ::=  re('-?[0-9]+')
float    ::=  re('-?[0-9]+(\.[0-9]*)?([eE][+-]?[0-9+])?') | NAN | INFINITY
boolean  ::=  TRUE | FALSE
uuid     ::=  hex{8}-hex{4}-hex{4}-hex{4}-hex{12}
hex      ::=  re("[0-9a-fA-F]")
blob     ::=  '0' ('x' | 'X') hex+
```

Conditions

Un terme désigne le type de valeurs prises en charge. Les termes sont définis par ce qui suit.

```
term                ::=  constant | literal | function_call | arithmetic_operation |
 type_hint | bind_marker
literal             ::=  collection_literal | tuple_literal
function_call       ::=  identifier '(' [ term (',' term)* ] ')'
arithmetic_operation ::=  '-' term | term ('+' | '-' | '*' | '/' | '%') term
```

Types de données

Amazon Keyspaces prend en charge les types de données suivants :

Types de chaîne

Type de données	Description
`ascii`	Représente une chaîne de caractères ASCII.

Type de données	Description
`text`	Représente une chaîne encodée UTF-8.
`varchar`	Représente une chaîne encodée UTF-8 (`varchar` est un alias pour `text`).

Types numériques

Type de données	Description
`bigint`	Représente un entier long signé 64 bits.
`counter`	Représente un compteur d'entier signé 64 bits. Pour plus d'informations, consultez the section called “Compteurs”.
`decimal`	Représente une décimale de précision variable.
`double`	Représente une virgule flottante IEEE 754 64 bits.
`float`	Représente une virgule flottante IEEE 754 32 bits.
`int`	Représente un entier signé 32 bits.
`varint`	Représente un entier de précision arbitraire.

Compteurs

Une colonne `counter` contient un entier signé 64 bits. La valeur du compteur est incrémentée ou décrémentée à l'aide de l'instruction the section called “UPDATE”, et ne peut pas être définie directement. Cela rend les colonnes `counter` utiles pour le suivi des dénombrements. Par exemple, vous pouvez utiliser des compteurs pour suivre le nombre d'entrées dans un fichier journal ou le nombre de fois qu'une publication a été vue sur un réseau social. Les restrictions ci-après s'appliquent aux colonnes `counter` :

- Une colonne de type `counter` ne peut pas faire partie de la `primary key` d'une table.
- Dans une table qui contient une ou plusieurs colonnes de type `counter`, toutes les colonnes de cette table doivent être de type `counter`.

Dans les cas où la mise à jour d'un compteur échoue (par exemple, en raison d'un délai d'attente ou d'une perte de connexion avec Amazon Keyspaces), le client ne sait pas si la valeur du compteur a été mise à jour. Si la mise à jour est réessayée, la mise à jour de la valeur du compteur peut être appliquée une deuxième fois.

Type de blob

Type de données	Description
`blob`	Représente les octets arbitraires.

Type Boolean

Type de données	Description
`boolean`	Représente `true` ou `false`.

Types liés au temps

Type de données	Description
`timestamp`	Entier signé de 64 bits représentant la date et l'heure depuis l'époque (1er janvier 1970 à 00:00:00 GMT) en millisecondes.
`timeuuid`	Représente un UUID version 1.

Types de collections

Type de données	Description
`list`	Représente une collection ordonnée d'éléments littéraux.
`map`	Représente une collection non ordonnée de paires clé-valeur.
`set`	Représente une collection non ordonnée d'un ou plusieurs éléments littéraux.

Vous déclarez une colonne de collection en utilisant le type de collection suivi d'un autre type de données (par exemple, `TEXT` ou`INT`) entre crochets. Vous pouvez créer une colonne avec un `SET` de`TEXT`, ou vous pouvez créer une paire clé-valeur `TEXT` et une `MAP` paire `INT` clé-valeur, comme indiqué dans l'exemple suivant.

```
SET <TEXT>
MAP <TEXT, INT>
```

Une collection non figée vous permet de mettre à jour chaque élément de collection individuel. Les horodatages côté client et les paramètres de durée de vie (TTL) sont enregistrés pour les éléments individuels.

Lorsque vous utilisez le `FROZEN` mot clé sur un type de collection, les valeurs de la collection sont sérialisées en une seule valeur immuable, et Amazon Keyspaces les traite comme un. `BLOB` Il s'agit d'une collection surgelée. Une `UPDATE` déclaration `INSERT` OR remplace l'ensemble de la collection Frozen. Vous ne pouvez pas mettre à jour des éléments individuels d'une collection figée.

Les paramètres d'horodatage et de durée de vie (TTL) côté client s'appliquent à l'ensemble de la collection figée, et non à des éléments individuels. `Frozen`les colonnes de collection peuvent faire partie `PRIMARY KEY` d'un tableau.

Vous pouvez imbriquer des collections congelées. Par exemple, vous pouvez définir un `MAP` dans un `SET` si le `FROZEN` mot clé `MAP` est utilisé, comme illustré dans l'exemple suivant.

```
SET <FROZEN> <MAP <TEXT, INT>>>
```

Amazon Keyspaces prend en charge l'imbrication d'un maximum de cinq niveaux de collections figées par défaut. Pour plus d'informations, consultez the section called "Quotas de service Amazon Keyspaces". Pour plus d'informations sur les différences fonctionnelles avec Apache Cassandra, consultezthe section called "FROZENcollections". Pour plus d'informations sur la syntaxe CQL, consultez the section called "CREATE TABLE" etthe section called "ALTER TABLE".

Type de tuple

Le type de `tuple` données représente un groupe limité d'éléments littéraux. Vous pouvez utiliser un tuple comme alternative à un`user defined type`. Il n'est pas nécessaire d'utiliser le `FROZEN` mot-clé pour les tuples. En effet, un tuple est toujours figé et vous ne pouvez pas mettre à jour les éléments individuellement.

Autres types

Type de données	Description
`inet`	Chaîne représentant une adresse IP, au format IPv4 ou IPv6.

Statique

Dans un tableau Amazon Keyspaces avec des colonnes de regroupement, vous pouvez utiliser le `STATIC` mot clé pour créer une colonne statique de n'importe quel type.

La déclaration suivante en est un exemple.

```
my_column INT STATIC
```

Pour plus d'informations sur l'utilisation de colonnes statiques, consultezthe section called "Colonnes statiques".

Encodage JSON des types de données Amazon Keyspaces

Amazon Keyspaces propose les mêmes mappages de types de données JSON qu'Apache Cassandra. Le tableau suivant décrit les types de données qu'Amazon Keyspaces accepte dans les `INSERT JSON` instructions et les types de données qu'Amazon Keyspaces utilise lorsqu'il renvoie des données avec l'instruction. `SELECT JSON`

Pour les types de données à champ unique tels que `floatint`,`UUID`, et`date`, vous pouvez également insérer des données sous forme de`string`. Pour les types de données composés et les collections, tels que `tuplemap`, et`list`, vous pouvez également insérer des données au format JSON ou sous forme codée`JSON string`.

Type de données JSON	Types de données acceptés dans les **`INSERT JSON`** relevés	Types de données renvoyés dans les **`SELECT JSON`** relevés	Remarques
`ascii`	`string`	`string`	Utilise l'échappement de caractères JSON\u.
`bigint`	`integer, string`	`integer`	La chaîne doit être un entier de 64 bits valide.
`blob`	`string`	`string`	La chaîne doit commencer `0x` par un nombre pair de chiffres hexadécimaux.
`boolean`	`boolean, string`	`boolean`	La chaîne doit être l'une `true` ou l'autre`false`.
`date`	`string`	`string`	Date au format`YYYY-MM-DD` , fuseau horaire UTC.
`decimal`	`integer, float, string`	`float`	Peut dépasser la précision à virgule flottante IEEE-754 32 bits ou 64 bits dans le décodeur côté client.

Type de données JSON	Types de données acceptés dans les **`INSERT JSON`** relevés	Types de données renvoyés dans les **`SELECT JSON`** relevés	Remarques
`double`	`integer, float, string`	`float`	La chaîne doit être un entier ou un nombre flottant valide.
`float`	`integer, float, string`	`float`	La chaîne doit être un entier ou un nombre flottant valide.
`inet`	`string`	`string`	Adresse IPv4 ou IPv6.
`int`	`integer, string`	`integer`	La chaîne doit être un entier 32 bits valide.
`list`	`list, string`	`list`	Utilise la représent ation de liste JSON native.
`map`	`map, string`	`map`	Utilise la représent ation cartographique JSON native.
`smallint`	`integer, string`	`integer`	La chaîne doit être un entier de 16 bits valide.
`set`	`list, string`	`list`	Utilise la représent ation de liste JSON native.
`text`	`string`	`string`	Utilise l'échappe ment de caractères JSON\u.

Type de données JSON	Types de données acceptés dans les **`INSERT JSON`** relevés	Types de données renvoyés dans les **`SELECT JSON`** relevés	Remarques
`time`	`string`	`string`	Heure du jour au format`HH-MM-SS[.fffffffff]` .
`timestamp`	`integer, string`	`string`	Horodatage. Les constantes de chaîne vous permettent de stocker des horodatages sous forme de dates. Les horodatages avec format `YYYY-MM-DD HH:MM:SS.SSS` sont renvoyés.
`timeuuid`	`string`	`string`	Tapez 1 UUID. Voir constants pour le format UUID.
`tinyint`	`integer, string`	`integer`	La chaîne doit être un entier de 8 bits valide.
`tuple`	`list, string`	`list`	Utilise la représentation de liste JSON native.
`uuid`	`string`	`string`	Voir constants pour le format UUID.
`varchar`	`string`	`string`	Utilise l'échappement de caractères JSON\u.

Type de données JSON	Types de données acceptés dans les **`INSERT JSON`** relevés	Types de données renvoyés dans les **`SELECT JSON`** relevés	Remarques
`varint`	`integer, string`	`integer`	Longueur variable ; peut dépasser les nombres entiers de 32 bits ou 64 bits dans le décodeur côté client.

Instructions DDL (langage de définition des données) dans Amazon Keyspaces

Le langage de définition des données (DDL) est l'ensemble des instructions Cassandra Query Language (CQL) que vous utilisez pour gérer les structures de données dans Amazon Keyspaces (pour Apache Cassandra), telles que les espaces de touches et les tables. Vous utilisez DDL pour créer ces structures de données, les modifier après leur création et les supprimer lorsqu'elles ne sont plus utilisées. Amazon Keyspaces exécute des opérations DDL de manière asynchrone. Pour plus d'informations sur la façon de confirmer qu'une opération asynchrone est terminée, consultez. the section called “Création et suppression asynchrones d'espaces clés et de tables”

Les instructions DDL suivantes sont prises en charge :

- CREATE KEYSPACE
- ALTER KEYSPACE
- DROP KEYSPACE
- CRÉER UNE TABLE
- ALTER TABLE
- RESTAURER LA TABLE
- SUPPRIMER LA TABLE

Rubriques

- Keyspaces

- Tables

Keyspaces

Un keyspace regroupe les tables associées qui sont pertinentes pour une ou plusieurs applications. En termes de système de gestion de base de données relationnelle (SGBDR), les keyspaces sont à peu près similaires aux bases de données, tablespaces ou constructions similaires.

> **Note**
>
> Dans Apache Cassandra, les keyspaces déterminent la réplique des données entre plusieurs nœuds de stockage. Cependant, Amazon Keyspaces est un service entièrement géré : les détails de sa couche de stockage sont gérés en votre nom. Pour cette raison, les espaces clés d'Amazon Keyspaces sont uniquement des constructions logiques et ne sont pas liés au stockage physique sous-jacent.

Pour plus d'informations sur les limites de quota et les contraintes pour les espaces de clés Amazon Keyspaces, consultez. Quotas

Déclarations pour les espaces clés

- CREATE KEYSPACE
- ALTER KEYSPACE
- DROP KEYSPACE

CREATE KEYSPACE

Utilisez l'instruction `CREATE KEYSPACE` pour créer un nouveau keyspace.

Syntaxe

```
create_keyspace_statement ::=
    CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name
    WITH options
```

Où :

- *keyspace_name* est le nom du keyspace à créer.

- Les options peuvent être les suivantes :
  - `REPLICATION`— Une carte qui indique la stratégie de réplication pour le keyspace :
    - `SingleRegionStrategy`— Pour un keyspace à région unique. (Obligatoire)
    - `NetworkTopologyStrategy`— Spécifiez au moins deux et jusqu'à six Régions AWS. Le facteur de réplication pour chaque région est de trois. (Facultatif)
  - `DURABLE_WRITES`— Les écritures sur Amazon Keyspaces sont toujours durables, cette option n'est donc pas requise. Toutefois, si elle est spécifiée, la valeur doit être `true`.
  - `TAGS`— Une liste de balises de paires clé-valeur à associer à la ressource lorsque vous la créez. (Facultatif)

Exemple

Créez un keyspace comme suit.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class': 'SingleRegionStrategy'} and TAGS ={'key1':'val1',
 'key2':'val2'} ;
```

Pour créer un espace de touches multirégional, spécifiez `NetworkTopologyStrategy` et incluez au moins deux et jusqu'à six. Régions AWS Le facteur de réplication pour chaque région est de trois.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3'};
```

ALTER KEYSPACE

Utilisez `ALTER KEYSPACE` pour ajouter ou supprimer les balises d'un keyspace.

Syntaxe

```
alter_keyspace_statement ::=
    ALTER KEYSPACE keyspace_name
    [[ADD | DROP] TAGS
```

Où :

- *`keyspace_name`* est le nom du keyspace à modifier.

- TAGS— Une liste de balises de paires clé-valeur à ajouter ou à supprimer de l'espace clé.

Exemple

Modifiez un keyspace comme suit.

```
ALTER KEYSPACE "myGSGKeyspace" ADD TAGS {'key1':'val1', 'key2':'val2'};
```

DROP KEYSPACE

Utilisez l'DROP KEYSPACEinstruction pour supprimer un espace de touches, y compris l'ensemble de son contenu, tel que les tableaux.

Syntaxe

```
drop_keyspace_statement ::=
    DROP KEYSPACE [ IF EXISTS ] keyspace_name
```

Où :

- keyspace_name est le nom du keyspace à supprimer.

Exemple

```
DROP KEYSPACE "myGSGKeyspace";
```

Tables

Les tables sont les principales structures de données d'Amazon Keyspaces. Les données d'une table sont organisées en lignes et en colonnes. Un sous-ensemble de ces colonnes est utilisé pour déterminer le partitionnement (et finalement le placement des données) via la spécification d'une clé de partition.

Un autre ensemble de colonnes peut être défini en colonnes de clustering, ce qui signifie qu'elles peuvent participer en tant que prédicats à l'exécution de la requête.

Par défaut, de nouvelles tables sont créées avec une capacité de débit à la demande. Vous pouvez modifier le mode de capacité pour les tables nouvelles et existantes. Pour de plus amples

informations sur les modes de débit de capacité de lecture/écriture, reportez-vous à la section the section called “Modes de capacité de lecture/écriture”.

Pour les tables en mode provisionné, vous pouvez configurer en option`AUTOSCALING_SETTINGS`. Pour plus d'informations sur le dimensionnement automatique d'Amazon Keyspaces et les options disponibles, consultez. the section called “Utilisation de CQL”

Pour plus d'informations sur les limites de quota et les contraintes pour les tables Amazon Keyspaces, consultez. Quotas

Relevés pour tableaux

- CREATE TABLE
- ALTER TABLE
- RESTAURER LA TABLE
- DROP TABLE

CREATE TABLE

Utilisez l'instruction `CREATE TABLE` pour créer une nouvelle table.

Syntaxe

```
create_table_statement ::=  CREATE TABLE [ IF NOT EXISTS ] table_name
    '('
        column_definition
        ( ',' column_definition )*
        [ ',' PRIMARY KEY '(' primary_key ')' ]
    ')' [ WITH table_options ]

column_definition      ::=  column_name cql_type [ FROZEN ][ STATIC ][ PRIMARY KEY]

primary_key            ::=  partition_key [ ',' clustering_columns ]

partition_key          ::=  column_name
                              | '(' column_name ( ',' column_name )* ')'

clustering_columns     ::=  column_name ( ',' column_name )*

table_options          ::=  [table_options]
                              | CLUSTERING ORDER BY '(' clustering_order
 ')' [ AND table_options ]
```

```
                              | options
                              | CUSTOM_PROPERTIES
                              | AUTOSCALING_SETTINGS
                              | default_time_to_live
                              | TAGS

clustering_order       ::=  column_name (ASC | DESC) ( ',' column_name (ASC | DESC) )*
```

Où :

- *table_name* est le nom de la table à créer.
- *column_definition* comprend les éléments suivants :
  - *column_name*— Le nom de la colonne.
  - *cql_type*— Un type de données Amazon Keyspaces (voirTypes de données).
  - *FROZEN*— Désigne cette colonne de type `collection` (par exemple, `LISTSET`, ou`MAP`) comme figée. Une collection figée est sérialisée en une seule valeur immuable et traitée comme une. `BLOB` Pour plus d'informations, consultez the section called “Types de collections”.
  - *STATIC*— Désigne cette colonne comme statique. Les colonnes statiques stockent les valeurs partagées par toutes les lignes d'une même partition.
  - *PRIMARY KEY*— Désigne cette colonne comme clé primaire de la table.
- *primary_key* comprend les éléments suivants :
  - `partition_key`
  - `clustering_columns`
- *partition_key*:
  - La clé de partition peut être une seule colonne ; il peut aussi s'agir d'une valeur composée de deux colonnes ou plus. La partie clé de partition de la clé primaire est obligatoire et détermine la manière dont Amazon Keyspaces stocke vos données.
- *clustering_columns*:
  - La partie facultative de la colonne de clustering de votre clé primaire détermine la façon dont les données sont regroupées et triées dans chaque partition.
- *table_options*se composent des éléments suivants :
  - *CLUSTERING ORDER BY*— L'ORDRE DE CLUSTERING par défaut sur une table est composé de vos clés de clustering dans le sens de `ASC` tri (croissant). Spécifiez-le pour remplacer le comportement de tri par défaut.
  - *CUSTOM_PROPERTIES*— Une carte des paramètres spécifiques à Amazon Keyspaces.

- `capacity_mode`: spécifie le mode de capacité de débit en lecture/écriture pour la table. Les options sont `throughput_mode:PAY_PER_REQUEST` et `throughput_mode:PROVISIONED`. Le mode de capacité allouée nécessite `read_capacity_units` et `write_capacity_units` en tant qu'entrées. L'argument par défaut est `throughput_mode:PAY_PER_REQUEST`.
- `client_side_timestamps`: Spécifie si les horodatages côté client sont activés ou désactivés pour la table. Les options sont `{'status': 'enabled'}` et `{'status': 'disabled'}`. S'il n'est pas spécifié, la valeur par défaut est`status:disabled`. Une fois les horodatages côté client activés pour une table, ce paramètre ne peut pas être désactivé.
- `encryption_specification`: Spécifie les options de chiffrement pour le chiffrement au repos. S'il n'est pas spécifié, la valeur par défaut est`encryption_type:AWS_OWNED_KMS_KEY`. L'option de chiffrement (clé gérée par le client) nécessite la AWS KMS clé au format Amazon Resource Name (ARN) comme entrée : `kms_key_identifier:ARN` :`kms_key_identifier:ARN`.
- `point_in_time_recovery`: Spécifie si point-in-time la restauration est activée ou désactivée pour la table. Les options sont `status:enabled` et `status:disabled`. S'il n'est pas spécifié, la valeur par défaut est`status:disabled`.
- `replica_updates`: Spécifie les paramètres d'une table multirégionale spécifiques à un Région AWS. Pour une table multirégionale, vous pouvez configurer la capacité de lecture de la table différemment selon Région AWS les régions. Vous pouvez le faire en configurant les paramètres suivants. Pour plus d'informations et d'exemples, consultez the section called "Création d'une table multirégionale avec mode capacité provisionnée et mise à l'échelle automatique (CQL)".
  - `region`— La réplique Région AWS de la table avec les paramètres suivants :
    - `read_capacity_units`
- TTL: active les paramètres personnalisés Time to Live pour le tableau. Pour l'activer, utilisez`status:enabled`. L'argument par défaut est `status:disabled`. Une fois TTL activé, vous ne pouvez pas le désactiver pour le tableau.

- *`AUTOSCALING_SETTINGS`*inclut les paramètres facultatifs suivants pour les tables en mode provisionné. Pour plus d'informations et d'exemples, consultez the section called "Création d'une nouvelle table avec mise à l'échelle automatique à l'aide de CQL".
  - `provisioned_write_capacity_autoscaling_update`:

- `autoscaling_disabled`— Pour activer la mise à l'échelle automatique en fonction de la capacité d'écriture, définissez la valeur sur`false`. L'argument par défaut est `true`. (Facultatif)
- `minimum_units`— Le niveau minimum de débit d'écriture que la table doit toujours être prête à prendre en charge. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
- `maximum_units`— Le niveau maximal de débit d'écriture que la table doit toujours être prête à prendre en charge. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
- `scaling_policy`— Amazon Keyspaces soutient la politique de suivi des cibles. L'objectif de dimensionnement automatique est la capacité d'écriture allouée à la table.
  - `target_tracking_scaling_policy_configuration`— Pour définir la politique de suivi de la cible, vous devez définir la valeur cible. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.
    - `target_value`— Le taux d'utilisation cible de la table. Le dimensionnement automatique d'Amazon Keyspaces garantit que le rapport entre la capacité consommée et la capacité allouée reste égal ou proche de cette valeur. Vous définissez `target_value` en tant que pourcentage. Un double entre 20 et 90. (Obligatoire)
    - `scale_in_cooldown`— Un temps de recharge en secondes entre les activités de mise à l'échelle, qui permet à la table de se stabiliser avant qu'une autre activité d'échelle ne commence. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)
    - `scale_out_cooldown`— Un temps de recharge en secondes entre les activités de redimensionnement qui permet à la table de se stabiliser avant le début d'une autre activité de redimensionnement. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)
    - `disable_scale_in`: A `boolean` qui indique si la table `scale-in` est désactivée ou activée. Ce paramètre est désactivé par défaut. Pour l'activer`scale-in`, définissez la `boolean` valeur sur`FALSE`. Cela signifie que la capacité est automatiquement réduite pour une table en votre nom. (Facultatif)
- `provisioned_read_capacity_autoscaling_update`:

- `autoscaling_disabled`— Pour activer la mise à l'échelle automatique en fonction de la capacité de lecture, définissez la valeur sur`false`. L'argument par défaut est `true`. (Facultatif)
- `minimum_units`— Le niveau de débit minimal que la table doit toujours être prête à supporter. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
- `maximum_units`— Le niveau de débit maximal que la table doit toujours être prête à supporter. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
- `scaling_policy`— Amazon Keyspaces soutient la politique de suivi des cibles. L'objectif de dimensionnement automatique est la capacité de lecture allouée à la table.
  - `target_tracking_scaling_policy_configuration`— Pour définir la politique de suivi de la cible, vous devez définir la valeur cible. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.
    - `target_value`— Le taux d'utilisation cible de la table. Le dimensionnement automatique d'Amazon Keyspaces garantit que le rapport entre la capacité consommée et la capacité allouée reste égal ou proche de cette valeur. Vous définissez `target_value` en tant que pourcentage. Un double entre 20 et 90. (Obligatoire)
    - `scale_in_cooldown`— Un temps de recharge en secondes entre les activités de mise à l'échelle, qui permet à la table de se stabiliser avant qu'une autre activité d'échelle ne commence. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)
    - `scale_out_cooldown`— Un temps de recharge en secondes entre les activités de redimensionnement qui permet à la table de se stabiliser avant le début d'une autre activité de redimensionnement. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)
    - `disable_scale_in`: A `boolean` qui indique si la table `scale-in` est désactivée ou activée. Ce paramètre est désactivé par défaut. Pour l'activer`scale-in`, définissez la `boolean` valeur sur`FALSE`. Cela signifie que la capacité est automatiquement réduite pour une table en votre nom. (Facultatif)
- `replica_updates`: Spécifie les paramètres de mise à l'échelle automatique Région AWS spécifiques d'une table multirégionale. Pour une table multirégionale, vous pouvez configurer la capacité de lecture de la table différemment selon Région AWS les régions. Vous pouvez le

faire en configurant les paramètres suivants. Pour plus d'informations et d'exemples, consultez the section called "Création d'une table multirégionale avec mode capacité provisionnée et mise à l'échelle automatique (CQL)".

- `region`— La réplique Région AWS de la table avec les paramètres suivants :
  - `provisioned_read_capacity_autoscaling_update`
    - `autoscaling_disabled`— Pour activer la mise à l'échelle automatique en fonction de la capacité de lecture de la table, définissez la valeur sur`false`. L'argument par défaut est `true`. (Facultatif)

      > **Note**
      >
      > Le dimensionnement automatique d'une table multirégionale doit être activé ou désactivé pour toutes les répliques de la table.

    - `minimum_units`— Le niveau minimum de débit de lecture que la table doit toujours être prête à prendre en charge. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
    - `maximum_units`— Le niveau maximal de débit de lecture que la table doit toujours être prête à prendre en charge. La valeur doit être comprise entre 1 et le quota de débit maximal par seconde pour votre compte (40 000 par défaut).
    - `scaling_policy`— Amazon Keyspaces soutient la politique de suivi des cibles. L'objectif de dimensionnement automatique est la capacité de lecture allouée à la table.
      - `target_tracking_scaling_policy_configuration`— Pour définir la politique de suivi de la cible, vous devez définir la valeur cible. Pour plus d'informations sur le suivi des cibles et les périodes de recharge, consultez les politiques de dimensionnement de Target Tracking dans le guide de l'utilisateur d'Application Auto Scaling.
        - `target_value`— Le taux d'utilisation cible de la table. Le dimensionnement automatique d'Amazon Keyspaces garantit que le rapport entre la capacité de lecture consommée et la capacité de lecture allouée reste égal ou proche de cette valeur. Vous définissez `target_value` en tant que pourcentage. Un double entre 20 et 90. (Obligatoire)
        - `scale_in_cooldown`— Un temps de recharge en secondes entre les activités de mise à l'échelle, qui permet à la table de se stabiliser avant qu'une autre activité

d'échelle ne commence. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)

- `scale_out_cooldown`— Un temps de recharge en secondes entre les activités de redimensionnement qui permet à la table de se stabiliser avant le début d'une autre activité de redimensionnement. Si aucune valeur n'est fournie, la valeur par défaut est 0. (Facultatif)
- `disable_scale_in`: A `boolean` qui indique si la table `scale-in` est désactivée ou activée. Ce paramètre est désactivé par défaut. Pour l'activer`scale-in`, définissez la `boolean` valeur sur`FALSE`. Cela signifie que la capacité de lecture est automatiquement réduite pour une table en votre nom. (Facultatif)

- `default_time_to_live`— Le paramètre Durée de vie par défaut en secondes pour le tableau.
- `TAGS`— Une liste de balises de paires clé-valeur à associer à la ressource lors de sa création.

- *`clustering_order`* comprend les éléments suivants :
  - *`column_name`*— Le nom de la colonne.
  - *`ASC | DESC`*— Définit le modificateur d'ordre ascendant (`ASC`) ou descendant (`DESC`). S'il n'est pas spécifié, l'ordre par défaut est ASC.

Exemple

```
CREATE TABLE IF NOT EXISTS "my_keyspace".my_table (
                                            id text,
                                            name text,
                                            region text,
                                            division text,
                                            project text,
                                            role text,
                                            pay_scale int,
                                            vacation_hrs float,
                                            manager_id text,
                                            PRIMARY KEY (id,division))
                                            WITH CUSTOM_PROPERTIES={
                                                'capacity_mode':{
                                                        'throughput_mode':
 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units': 20
                                                    },
                                                'point_in_time_recovery':{'status':
 'enabled'},
```

```
                                                'encryption_specification':{
                                                        'encryption_type':
 'CUSTOMER_MANAGED_KMS_KEY',

 'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                                                            }
                                                }
                                                AND CLUSTERING ORDER BY (division ASC)
                                                AND TAGS={'key1':'val1', 'key2':'val2'}
                                                AND default_time_to_live = 3024000;
```

Dans une table qui utilise des colonnes de clustering, les colonnes non clusterisées peuvent être déclarées statiques dans la définition de la table. Pour plus d'informations sur les colonnes statiques, consultezthe section called “Colonnes statiques”.

Exemple

```
CREATE TABLE "my_keyspace".my_table (
                                                id int,
                                                name text,
                                                region text,
                                                division text,
                                                project text STATIC,
                                                PRIMARY KEY (id,division));
```

ALTER TABLE

Utilisez l'ALTER TABLEinstruction pour ajouter de nouvelles colonnes, ajouter des balises ou modifier les propriétés personnalisées du tableau.

Syntaxe

```
alter_table_statement ::=  ALTER TABLE table_name

        [ ADD ( column_definition | column_definition_list)  ]
        [[ADD | DROP] TAGS {'key1':'val1', 'key2':'val2'}]
        [ WITH table_options [ , ... ] ] ;

column_definition      ::=  column_name cql_type
```

Où :

- *table_name* est le nom de la table à modifier.
- *column_definition* est le nom de la colonne et le type de données à ajouter.
- *column_definition_list* est une liste de colonnes séparées par des virgules placées entre parenthèses.
- *table_options*se composent des éléments suivants :
  - *CUSTOM_PROPERTIES*— Une carte des paramètres spécifiques à Amazon Keyspaces.
    - `capacity_mode`: spécifie le mode de capacité de débit en lecture/écriture pour la table. Les options sont `throughput_mode:PAY_PER_REQUEST` et `throughput_mode:PROVISIONED`. Le mode de capacité allouée nécessite `read_capacity_units` et `write_capacity_units` en tant qu'entrées. L'argument par défaut est `throughput_mode:PAY_PER_REQUEST`.
    - `client_side_timestamps`: Spécifie si les horodatages côté client sont activés ou désactivés pour la table. Les options sont `{'status': 'enabled'}` et `{'status': 'disabled'}`. S'il n'est pas spécifié, la valeur par défaut est`status:disabled`. Une fois les horodatages côté client activés pour une table, ce paramètre ne peut pas être désactivé.
    - `encryption_specification`: Spécifie l'option de chiffrement pour le chiffrement au repos. Les options sont `encryption_type:AWS_OWNED_KMS_KEY` et `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. L'option de chiffrement (clé gérée par le client) nécessite la AWS KMS clé au format Amazon Resource Name (ARN) comme entrée :`kms_key_identifier:ARN`.
    - `point_in_time_recovery`: Spécifie si point-in-time la restauration est activée ou désactivée pour la table. Les options sont `status:enabled` et `status:disabled`. L'argument par défaut est `status:disabled`.
    - `replica_updates`: Spécifie les paramètres Région AWS spécifiques d'une table multirégionale. Pour une table multirégionale, vous pouvez configurer la capacité de lecture de la table différemment selon Région AWS les régions. Vous pouvez le faire en configurant les paramètres suivants. Pour plus d'informations et d'exemples, consultez the section called "Mise à jour de la capacité allouée et des paramètres de dimensionnement automatique d'une table multirégionale (CQL)".
      - `region`— La réplique Région AWS de la table avec les paramètres suivants :
        - `read_capacity_units`

    - `ttl`: active les paramètres personnalisés Time to Live pour le tableau. Pour l'activer, utilisez`status:enabled`. L'argument par défaut est `status:disabled`. Une fois `ttl` activé, vous ne pouvez pas le désactiver pour le tableau.
  - *`AUTOSCALING_SETTINGS`*inclut les paramètres de mise à l'échelle automatique facultatifs pour les tables provisionnées. Pour la syntaxe et les descriptions détaillées, voirthe section called "CREATE TABLE". Pour obtenir des exemples, consultez the section called "Activer la mise à l'échelle automatique sur une table existante à l'aide de CQL".
- `default_time_to_live`: paramètre de durée de vie par défaut en secondes pour le tableau.
- *`TAGS`* est une liste de balises de paire clé-valeur à attacher à la ressource.

> **Note**
>
> Avec ALTER TABLE, vous ne pouvez modifier qu'une seule propriété personnalisée. Vous ne pouvez pas combiner plusieurs commandes ALTER TABLE dans la même instruction.

Exemples

L'instruction suivante indique comment ajouter une colonne à une table existante.

```
ALTER TABLE mykeyspace.mytable ADD (ID int);
```

Cette instruction indique comment ajouter deux colonnes de collection à une table existante :

- Colonne de collection figée `col_frozen_list` contenant une collection figée imbriquée
- Colonne de collection non figée `col_map` contenant une collection figée imbriquée

```
ALTER TABLE my_Table ADD(col_frozen_list FROZEN<LIST<FROZEN<SET<TEXT>>>>, col_map
 MAP<INT, FROZEN<SET<INT>>>);
```

Pour modifier le mode de capacité d'une table et spécifier les unités de capacité de lecture et d'écriture, vous pouvez utiliser l'instruction suivante.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units':
 20}};
```

L'instruction suivante indique une clé KMS gérée par le client pour la table.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={
              'encryption_specification':{
                     'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                     'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                   }
         };
```

Pour activer point-in-time la restauration d'une table, vous pouvez utiliser l'instruction suivante.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'point_in_time_recovery':
 {'status': 'enabled'}};
```

Pour définir une valeur Time to Live par défaut en secondes pour une table, vous pouvez utiliser l'instruction suivante.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Cette instruction active les paramètres Time to Live personnalisés pour une table.

```
ALTER TABLE mytable WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

RESTAURER LA TABLE

Utilisez l'`RESTORE TABLE`instruction pour rétablir une table à un point précis dans le temps. Cette instruction nécessite que la point-in-time restauration soit activée sur une table. Pour plus d'informations, consultez oint-in-time Récupération de données.

Syntaxe

```
restore_table_statement ::=
    RESTORE TABLE restored_table_name FROM TABLE source_table_name
                  [ WITH table_options [ , ... ] ];
```

Où :

- *restored_table_name*est le nom de la table restaurée.

- *source_table_name*est le nom de la table source.
- *table_options* comprend les éléments suivants :
  - *restore_timestamp*est l'heure du point de restauration au format ISO 8601. S'il n'est pas spécifié, l'horodatage actuel est utilisé.
  - *CUSTOM_PROPERTIES*— Une carte des paramètres spécifiques à Amazon Keyspaces.
    - `capacity_mode`: spécifie le mode de capacité de débit en lecture/écriture pour la table. Les options sont `throughput_mode:PAY_PER_REQUEST` et `throughput_mode:PROVISIONED`. Le mode de capacité allouée nécessite `read_capacity_units` et `write_capacity_units` en tant qu'entrées. La valeur par défaut est le paramètre actuel de la table source.
    - `encryption_specification`: Spécifie l'option de chiffrement pour le chiffrement au repos. Les options sont `encryption_type:AWS_OWNED_KMS_KEY` et `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. L'option de chiffrement (clé gérée par le client) nécessite la AWS KMS clé au format Amazon Resource Name (ARN) comme entrée :`kms_key_identifier:ARN`. Pour restaurer une table chiffrée avec une clé gérée par le client vers une table chiffrée avec un Clé détenue par AWS, Amazon Keyspaces a besoin d'accéder à la AWS KMS clé de la table source.
    - `point_in_time_recovery`: Spécifie si point-in-time la restauration est activée ou désactivée pour la table. Les options sont `status:enabled` et `status:disabled`. Contrairement à ce qui se passe lorsque vous créez de nouvelles tables, le statut par défaut des tables restaurées est `status:enabled` dû au fait que le paramètre est hérité de la table source. Pour désactiver le PITR pour les tables restaurées, vous devez définir `status:disabled` explicitement.
    - `replica_updates`: Spécifie les paramètres Région AWS spécifiques d'une table multirégionale. Pour une table multirégionale, vous pouvez configurer la capacité de lecture de la table différemment selon Région AWS les régions. Vous pouvez le faire en configurant les paramètres suivants.
      - `region`— La réplique Région AWS de la table avec les paramètres suivants :
        - `read_capacity_units`
- *AUTOSCALING_SETTINGS*inclut les paramètres de mise à l'échelle automatique facultatifs pour les tables provisionnées. Pour une syntaxe et des descriptions détaillées, voirthe section called “CREATE TABLE”.
- *TAGS* est une liste de balises de paire clé-valeur à attacher à la ressource.

> **Note**
>
> Les tables supprimées ne peuvent être restaurées qu'au moment de leur suppression.

Exemple

```
RESTORE TABLE mykeyspace.mytable_restored from table mykeyspace.my_table
WITH restore_timestamp = '2020-06-30T04:05:00+0000'
AND custom_properties = {'point_in_time_recovery':{'status':'disabled'},
 'capacity_mode':{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10,
 'write_capacity_units': 20}}
AND TAGS={'key1':'val1', 'key2':'val2'};
```

DROP TABLE

Utilisez l'instruction `DROP TABLE` pour supprimer une table du keyspace.

Syntaxe

```
drop_table_statement ::=
    DROP TABLE [ IF EXISTS ] table_name
```

Où :

- `IF EXISTS` empêche `DROP TABLE` d'échouer si la table n'existe pas. (Facultatif)
- *`table_name`* est le nom de la table à supprimer.

Exemple

```
DROP TABLE "myGSGKeyspace".employees_tbl;
```

Instructions DML (langage de manipulation de données) dans Amazon Keyspaces

Le langage de manipulation de données (DML) est l'ensemble des instructions CQL (Cassandra Query Language) que vous utilisez pour gérer les données dans les tables Amazon Keyspaces (pour

Apache Cassandra). Vous utilisez des instructions DML pour ajouter, modifier ou supprimer des données dans une table.

Vous utilisez également les instructions DML pour interroger les données d'une table. (Notez que CQL ne prend pas en charge les jointures ou les sous-requêtes.)

Rubriques

SELECT

Utilisez une instruction SELECT pour interroger les données.

Syntaxe

```
select_statement ::=  SELECT  [ JSON ] ( select_clause | '*' )
                      FROM table_name
                      [ WHERE 'where_clause' ]
                      [ ORDER BY 'ordering_clause' ]
                      [ LIMIT (integer | bind_marker) ]
                      [ ALLOW FILTERING ]
select_clause    ::=  selector [ AS identifier ] ( ',' selector [ AS identifier ] )
selector         ::=  column_name
                      | term
                      | CAST '(' selector AS cql_type ')'
                      | function_name '(' [ selector ( ',' selector )* ] ')'
where_clause     ::=  relation ( AND relation )*
relation         ::=  column_name operator term
                      TOKEN
operator         ::=  '=' | '<' | '>' | '<=' | '>=' | IN | CONTAINS | CONTAINS KEY
ordering_clause  ::=  column_name [ ASC | DESC ] ( ',' column_name [ ASC | DESC ] )*
```

Exemples

```
SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

```
SELECT JSON name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

Pour un tableau qui fait correspondre les types de données codés en JSON aux types de données Amazon Keyspaces, consultez. the section called “Encodage JSON des types de données Amazon Keyspaces”

Utilisation du **`IN`** mot clé

Le `IN` mot clé indique l'égalité pour une ou plusieurs valeurs. Il peut être appliqué à la clé de partition et à la colonne de clustering. Les résultats sont renvoyés dans l'ordre dans lequel les clés sont présentées dans le `SELECT` relevé.

Exemples

```
SELECT * from mykeyspace.mytable WHERE primary.key1 IN (1,2) and clustering.key1 = 2;
SELECT * from mykeyspace.mytable WHERE primary.key1 IN (1,2) and clustering.key1 <= 2;
SELECT * from mykeyspace.mytable WHERE primary.key1 = 1 and clustering.key1 IN (1, 2);
SELECT * from mykeyspace.mytable WHERE primary.key1 <= 2 and clustering.key1 IN (1, 2)
 ALLOW FILTERING;
```

Pour plus d'informations sur le `IN` mot clé et sur la manière dont Amazon Keyspaces traite l'instruction, consultez. the section called “INSELECTDéclaration”

Résultats de commande

La `ORDER BY` clause indique l'ordre de tri des résultats renvoyés. Il prend comme arguments une liste de noms de colonnes ainsi que l'ordre de tri de chaque colonne. Vous ne pouvez spécifier des colonnes de clustering que dans les clauses de commande. Les colonnes non groupées ne sont pas autorisées. Les options d'ordre de tri concernent `ASC` l'ordre de tri croissant et `DESC` décroissant. Si l'ordre de tri est omis, l'ordre par défaut de la colonne de clustering est utilisé. Pour les ordres de tri possibles, voirthe section called “Résultats de commande”.

Exemple

```
SELECT name, id, division, manager_id FROM "myGSGKeyspace".employees_tbl WHERE id =
 '012-34-5678' ORDER BY division;
```

Lorsque vous utilisez `ORDER BY` le `IN` mot clé, les résultats sont classés dans une page. La réorganisation complète avec pagination désactivée n'est pas prise en charge.

JETON

Vous pouvez appliquer la TOKEN fonction à la PARTITION KEY colonne dans SELECT et WHERE aux clauses. Avec cette TOKEN fonction, Amazon Keyspaces renvoie des lignes en fonction de la valeur du jeton mappé PARTITION_KEY plutôt que de la valeur du. PARTITION KEY

TOKENles relations ne sont pas prises en charge avec le IN mot clé.

Exemples

```
SELECT TOKEN(id) from my_table;

SELECT TOKEN(id) from my_table WHERE TOKEN(id) > 100 and TOKEN(id) < 10000;
```

Fonction TTL

Vous pouvez utiliser la TTL fonction avec l'SELECTinstruction pour récupérer le délai d'expiration en secondes enregistré pour une colonne. Si aucune TTL valeur n'est définie, la fonction revientnull.

Exemple

```
SELECT TTL(my_column) from my_table;
```

La TTL fonction ne peut pas être utilisée sur des colonnes à plusieurs cellules telles que des collections.

WRITETIMEfonction

Vous pouvez utiliser la WRITETIME fonction avec l'SELECTinstruction pour récupérer l'horodatage stocké sous forme de métadonnées pour la valeur d'une colonne uniquement si la table utilise des horodatages côté client. Pour plus d'informations, consultez Horodatages côté client.

```
SELECT WRITETIME(my_column) from my_table;
```

La WRITETIME fonction ne peut pas être utilisée sur des colonnes à plusieurs cellules telles que des collections.

> **Note**
>
> Pour des raisons de compatibilité avec le comportement établi du pilote Cassandra, les politiques d'autorisation basées sur les balises ne sont pas appliquées lorsque vous effectuez des opérations sur les tables système à l'aide d'appels d'API Cassandra Query Language (CQL) via les pilotes Cassandra et les outils de développement. Pour plus d'informations,

consultez the section called “ Accès aux ressources Amazon Keyspaces basé sur des balises”.

INSERT

Utilisez l'instruction `INSERT` pour ajouter une ligne à une table.

Syntaxe

```
insert_statement ::=  INSERT INTO table_name ( names_values | json_clause )
                      [ IF NOT EXISTS ]
                      [ USING update_parameter ( AND update_parameter )* ]
names_values     ::=  names VALUES tuple_literal
json_clause      ::=  JSON string [ DEFAULT ( NULL | UNSET ) ]
names            ::=  '(' column_name ( ',' column_name )* ')'
```

Exemple

```
INSERT INTO "myGSGKeyspace".employees_tbl (id, name, project, region, division, role,
 pay_scale, vacation_hrs, manager_id)
VALUES ('012-34-5678','Russ','NightFlight','US','Engineering','IC',3,12.5,
 '234-56-7890') ;
```

Paramètres de mise à jour

`INSERT`prend en charge les valeurs suivantes comme suit `update_parameter` :

- `TTL`— Une valeur temporelle en secondes. La valeur maximale configurable est de 630 720 000 secondes, soit l'équivalent de 20 ans.
- `TIMESTAMP`— Une `bigint` valeur représentant le nombre de microsecondes écoulées depuis l'heure de base standard connue sous le nom de epoch : 1er janvier 1970 à 00h00 GMT. Dans Amazon Keyspaces, un horodatage doit être compris entre 2 jours dans le passé et 5 minutes dans le futur.

Exemple

```
INSERT INTO my_table (userid, time, subject, body, user)
```

```
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello','205.212.123.123')
        USING TTL 259200;
```

Support JSON

Pour un tableau qui fait correspondre les types de données codés en JSON aux types de données Amazon Keyspaces, consultez. the section called “Encodage JSON des types de données Amazon Keyspaces”

Vous pouvez utiliser le `JSON` mot clé pour insérer une carte `JSON` codée sous forme de ligne unique. Pour les colonnes qui existent dans le tableau mais qui sont omises dans l'instruction d'insertion JSON, utilisez-les `DEFAULT UNSET` pour conserver les valeurs existantes. `DEFAULT NULL`À utiliser pour écrire une valeur NULL dans chaque ligne de colonnes omises et remplacer les valeurs existantes (les frais d'écriture standard s'appliquent). `DEFAULT NULL`est l'option par défaut.

Exemple

```
INSERT INTO "myGSGKeyspace".employees_tbl JSON '{"id":"012-34-5678",
                                                 "name": "Russ",
                                                 "project": "NightFlight",
                                                 "region": "US",
                                                 "division": "Engineering",
                                                 "role": "IC",
                                                 "pay_scale": 3,
                                                 "vacation_hrs": 12.5,
                                                 "manager_id": "234-56-7890"}';
```

Si les données JSON contiennent des clés dupliquées, Amazon Keyspaces stocke la dernière valeur de la clé (similaire à Apache Cassandra). Dans l'exemple suivant, où se trouve la clé dupliquée`id`, la valeur `234-56-7890` est utilisée.

Exemple

```
INSERT INTO "myGSGKeyspace".employees_tbl JSON '{"id":"012-34-5678",
                                                 "name": "Russ",
                                                 "project": "NightFlight",
                                                 "region": "US",
                                                 "division": "Engineering",
                                                 "role": "IC",
                                                 "pay_scale": 3,
```

```
                                                    "vacation_hrs": 12.5,
                                                    "id": "234-56-7890"}';
```

UPDATE

Utilisez l'instruction UPDATE pour modifier une ligne d'une table.

Syntaxe

```
update_statement ::=  UPDATE table_name
                      [ USING update_parameter ( AND update_parameter )* ]
                      SET assignment ( ',' assignment )*
                      WHERE where_clause
                      [ IF ( EXISTS | condition ( AND condition )*) ]
update_parameter ::=  ( integer | bind_marker )
assignment       ::=  simple_selection '=' term
                     | column_name '=' column_name ( '+' | '-' ) term
                     | column_name '=' list_literal '+' column_name
simple_selection ::=  column_name
                     | column_name '[' term ']'
                     | column_name '.' `field_name
condition        ::=  simple_selection operator term
```

Exemple

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale = 5 WHERE id = '567-89-0123' AND
 division = 'Marketing' ;
```

Pour incrémenter un `counter`, utilisez la syntaxe suivante. Pour plus d'informations, consultez the section called “Compteurs”.

```
UPDATE ActiveUsers SET counter = counter + 1  WHERE user = A70FE1C0-5408-4AE3-
BE34-8733E5K09F14 AND action = 'click';
```

Paramètres de mise à jour

UPDATEprend en charge les valeurs suivantes comme suit `update_parameter` :

- TTL— Une valeur temporelle en secondes. La valeur maximale configurable est de 630 720 000 secondes, soit l'équivalent de 20 ans.

- `TIMESTAMP`— Une `bigint` valeur représentant le nombre de microsecondes écoulées depuis l'heure de base standard connue sous le nom de epoch : 1er janvier 1970 à 00h00 GMT. Dans Amazon Keyspaces, un horodatage doit être compris entre 2 jours dans le passé et 5 minutes dans le futur.

Exemple

```
UPDATE my_table (userid, time, subject, body, user)
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello again','205.212.123.123')
        USING TIMESTAMP '2022-11-03 13:30:54+0400';
```

DELETE

Utilisez l'instruction `DELETE` pour supprimer une ligne d'une table.

Syntaxe

```
delete_statement ::=  DELETE [ simple_selection ( ',' simple_selection ) ]
                      FROM table_name
                      [ USING update_parameter ( AND update_parameter )* ]
                      WHERE where_clause
                      [ IF ( EXISTS | condition ( AND condition )*) ]

simple_selection ::=  column_name
                     | column_name '[' term ']'
                     | column_name '.' `field_name

condition        ::=  simple_selection operator term
```

Où :

- *`table_name`* est la table qui contient la ligne à supprimer.

Exemple

```
DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive' ;
```

`DELETE`prend en charge la valeur suivante en tant que `update_parameter` :

- `TIMESTAMP`— Une `bigint` valeur représentant le nombre de microsecondes écoulées depuis l'heure de base standard connue sous le nom de epoch : 1er janvier 1970 à 00h00 GMT.

Fonctions intégrées dans Amazon Keyspaces

Amazon Keyspaces (pour Apache Cassandra) prend en charge diverses fonctions intégrées que vous pouvez utiliser dans les instructions CQL (Cassandra Query Language).

Rubriques

Fonctions scalaires

Une fonction scalaire effectue un calcul sur une valeur unique et renvoie le résultat sous forme de valeur unique. Amazon Keyspaces prend en charge les fonctions scalaires suivantes.

Fonction	Description
`blobAsType`	Renvoie une valeur du type de données spécifié.
`cast`	Convertit un type de données natif en un autre type de données natif.
`currentDate`	Renvoie la date/heure actuelle sous forme de date.
`currentTime`	Renvoie la date/heure actuelle sous forme d'heure.
`currentTimestamp`	Renvoie la date/heure actuelle sous forme d'horodatage.
`currentTimeUUID`	Renvoie la date/heure actuelle sous la forme d'un `timeuuid`.
`fromJson`	Convertit la chaîne JSON dans le type de données de la colonne sélectionnée.

Fonction	Description
`maxTimeuuid`	Renvoie la plus grande valeur possible `timeuuid` pour l'horodatage ou la chaîne de date.
`minTimeuuid`	Renvoie le plus petit possible `timeuuid` pour l'horodatage ou la chaîne de date.
`now`	Renvoie un nouveau `timeuuid` unique. Supporté pour les `DELETE` instructions `INSERTUPDATE`, et dans le cadre de la `WHERE` clause figurant dans les `SELECT` instructions.
`toDate`	Convertit un `timeuuid` ou un horodatage en un type de date.
`toJson`	Renvoie la valeur de colonne de la colonne sélectionnée au format JSON.
`token`	Renvoie la valeur de hachage de la clé de partition.
`toTimestamp`	Convertit un `timeuuid` ou une date en horodatage.
`TTL`	Renvoie le délai d'expiration en secondes pour une colonne.
`typeAsBlob`	Convertit le type de données spécifié en`blob`.
`toUnixTimestamp`	Convertit un `timeuuid` ou un horodatage en `bigInt`.
`uuid`	Renvoie un UUID de version 4 aléatoire . Supporté pour les `DELETE` instructions `INSERTUPDATE`, et dans le cadre de la `WHERE` clause figurant dans les `SELECT` instructions.

Fonction	Description
`writetime`	Renvoie l'horodatage de la valeur de la colonne spécifiée.
`dateOf`	(Obsolète) Extrait l'horodatage d'un `timeuuid` et renvoie la valeur sous forme de date.
`unixTimestampOf`	(Obsolète) Extrait l'horodatage d'un `timeuuid` et renvoie la valeur sous la forme d'un horodatage entier brut 64 bits.

Quotas pour Amazon Keyspaces (pour Apache Cassandra)

Cette section décrit les quotas actuels et les valeurs par défaut pour Amazon Keyspaces (pour Apache Cassandra).

Rubriques

Quotas de service Amazon Keyspaces

Le tableau suivant contient les quotas Amazon Keyspaces (pour Apache Cassandra) et les valeurs par défaut. Les informations sur les quotas pouvant être ajustés sont disponibles dans la console Service Quotas, où vous pouvez également demander des augmentations de quotas. Pour plus d'informations sur les quotas, contactez AWS Support.

Quota	Description	Amazon Keyspaces par défaut
Nombre maximum d'espaces de touches par Région AWS	Nombre maximal de keyspaces pour cet abonné par région. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	256
Nombre maximum de tables par Région AWS	Nombre maximal de tables dans tous les keyspaces pour cet abonné par région. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	256
Taille maximale du schéma de table	Taille maximale d'un schéma de table.	350 KO

Quota	Description	Amazon Keyspaces par défaut
Nombre maximal d'opérations DDL simultanées	Le nombre maximum d'opérations DDL simultanées autorisées pour cet abonné par région.	50
Nombre maximal de requêtes par connexion	Nombre maximal de requêtes CQL pouvant être traitées par une seule connexion TCP client par seconde.	3000
Taille maximale des lignes	Taille maximale d'une ligne, à l'exclusion des données de colonne statiques. Pour plus de détails, consultez the section called “Calcul de la taille des lignes”.	1 Mo
Nombre maximal de colonnes dans `INSERT` et de `UPDATE` relevés	Le nombre maximum de colonnes autorisées dans le CQL `INSERT` ou `UPDATE` les instructions. Une `UPDATE` instruction `INSERT` or prend en charge jusqu'à 225 colonnes normales lorsque Time to Live (TTL) est désactivé. Si le TTL est activé, il est possible de modifier jusqu'à 166 colonnes normales en une seule opération.	225/166

Quota	Description	Amazon Keyspaces par défaut
Nombre maximal de données statiques par partition logique	Taille agrégée maximale des données statiques dans une partition logique. Pour plus de détails, consultez the section called “Calcul de la taille de colonne statique par partition logique”.	1 Mo
Nombre maximum de sous-requêtes par instruction `IN` `SELECT`	Nombre maximal de sous-requêtes que vous pouvez utiliser pour le `IN` mot clé dans une `SELECT` instruction. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	100
Nombre maximum de collections congelées imbriquées par Région AWS	Le nombre maximum de collections imbriquées prises en charge lorsque vous utilisez le `FROZEN` mot clé pour une colonne contenant un type de données de collection. Pour plus d'informations sur les collections congelées, consultezthe section called “Types de collections”. Pour augmenter le niveau de nidification, contactez AWS Support.	5

Quota	Description	Amazon Keyspaces par défaut
Débit de lecture maximal par seconde	Débit de lecture maximal par seconde (unités de demande de lecture (RRU) ou unités de capacité de lecture (RCU) pouvant être allouées à une table par région. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	40 000
Débit d'écriture maximal par seconde	Débit d'écriture maximal par seconde (unités de demande d'écriture (WRU) ou unités de capacité d'écriture (WCU)) pouvant être alloué à une table par région. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	40 000
Débit de lecture au niveau du compte (alloué)	Le nombre maximum d'unités de capacité de lecture agrégée (RCU) allouées au compte par région. Cela s'applique uniquement aux tables en mode de capacité de lecture/écriture provision née. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	80 000

Quota	Description	Amazon Keyspaces par défaut
Débit d'écriture au niveau du compte (alloué)	Le nombre maximum d'unités de capacité d'écriture agrégée (WCU) allouées au compte par région. Cela s'applique uniquement aux tables en mode de capacité de lecture/écriture provisionnée. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	80 000
Nombre maximum de cibles évolutives par région et par compte	Le nombre maximum d'objectifs évolutifs pour le compte par région. Une table Amazon Keyspaces est considérée comme une cible évolutive si le dimensionnement automatique est activé pour la capacité de lecture, et comme une autre cible évolutive si le dimensionnement automatique est activé pour la capacité d'écriture. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas pour Application Auto Scaling en choisissant des cibles évolutives pour Amazon Keyspaces.	1 500

Quota	Description	Amazon Keyspaces par défaut
Taille maximale de la clé de partition	Taille maximale de la clé de partition composée. Jusqu'à 3 octets de stockage supplémentaire sont ajoutés à la taille brute de chaque colonne incluse dans la clé de partition pour les métadonnées.	2048 bytes
Taille maximale de la clé de clustering	Taille combinée maximale de toutes les colonnes de mise en cluster. Jusqu'à 4 octets de stockage supplémentaire sont ajoutés à la taille brute de chaque colonne de clustering pour les métadonnées.	850 octets
Nombre maximal de restaurations simultanées de tables à l'aide oint-in-time de P Recovery (PITR)	Le nombre maximum de restaurations de tables simultanées à l'aide du PITR par abonné est de 4. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	4
Quantité maximale de données restaurées à l'aide point-in-time de la restauration (PITR)	Taille maximale des données pouvant être restaurées à l'aide du PITR dans les 24 heures. Vous pouvez ajuster cette valeur par défaut dans la console Service Quotas.	5 To

Augmentation ou diminution du débit (pour les tables allouées)

Augmentation du débit alloué

Vous pouvez augmenter `ReadCapacityUnits` ou `WriteCapacityUnits` aussi souvent que nécessaire à l'aide de la console ou de l'`ALTER TABLE`instruction. Les nouveaux paramètres ne prennent effet qu'après que l'opération `ALTER TABLE` est achevée.

Vous ne pouvez pas dépasser vos quotas par compte lorsque vous ajoutez de la capacité provisionnée. Et vous pouvez augmenter la capacité allouée à vos tables autant que vous le souhaitez. Pour de plus amples informations sur les quotas par compte, veuillez consulter la section précédente, the section called “Quotas de service Amazon Keyspaces”.

Diminution du débit alloué

Pour chaque tableau d'un `ALTER TABLE` relevé, vous pouvez diminuer `ReadCapacityUnits` ou `WriteCapacityUnits` (ou les deux). Les nouveaux paramètres ne prennent effet qu'après que l'opération `ALTER TABLE` est achevée.

Vous pouvez effectuer jusqu'à quatre diminutions à n'importe quel moment de la journée. Une journée est définie conformément à l'heure UTC (Universal Coordinated Time). De plus, si aucune réduction n'a été effectuée au cours de la dernière heure, une réduction supplémentaire est autorisée. Cela porte effectivement à 27 le nombre maximum de diminutions par jour (4 diminutions au cours de la première heure et 1 diminution pour chacune des fenêtres d'une heure suivantes par jour).

Le chiffrement d'Amazon Keyspaces est au repos

Vous pouvez modifier les options de chiffrement entre une AWS KMS clé AWS détenue et une AWS KMS clé gérée par le client jusqu'à quatre fois par période de 24 heures, par table, à compter de la création de la table. Et si aucune modification n'est intervenue au cours des six dernières heures, une modification supplémentaire est autorisée. Cela porte ainsi à huit le nombre maximum de changements par jour (quatre changements au cours des six premières heures et un changement pour chacune des fenêtres de six heures suivantes de la journée).

Vous pouvez modifier l'option de chiffrement pour utiliser une AWS KMS clé AWS détenue aussi souvent que nécessaire, même si le quota précédent a été épuisé.

Voici les quotas, à moins que vous ne demandiez un seuil supérieur. Pour demander une augmentation du quota de service, consultez AWS Support.

Historique du document pour Amazon Keyspaces (pour Apache Cassandra)

Le tableau suivant décrit les modifications importantes apportées à la documentation depuis la dernière version d'Amazon Keyspaces (pour Apache Cassandra). Pour recevoir les notifications de mise à jour de cette documentation, abonnez-vous à un flux RSS.

- Dernière mise à jour de la documentation : 7 février 2024

Modification	Description	Date
Connectez-vous à Amazon Keyspaces depuis Amazon Elastic Kubernetes Service	Vous pouvez désormais suivre un step-by-step didacticiel pour vous connecter à Amazon Keyspaces depuis Amazon EKS.	7 février 2024
API de dimensionnement automatique Amazon Keyspaces pour les tables provisionnées	Amazon Keyspaces propose désormais un CQL support AWS API pour configurer le dimensionnement automatique avec le mode de capacité provisionnée.	23 janvier 2024
Support de réplication multirégionale d'Amazon Keyspaces pour les tables provisionnées	Amazon Keyspaces prend désormais en charge le mode de capacité provisionnée pour les tables multirégionales.	23 janvier 2024
Activité DML d'Amazon Keyspaces incluse dans les journaux CloudTrail	Vous pouvez désormais auditer les appels d'API Amazon Keyspaces Data Manipulation Language (DML) entrants. AWS CloudTrail	20 décembre 2023

Support d'Amazon Keyspaces pour le mot clé FROZEN	Amazon Keyspaces prend désormais en charge le `FROZEN` mot clé pour les types de données de collecte.	15 novembre 2023
Mise à jour de la politique gérée par Amazon Keyspaces	Amazon Keyspaces a ajouté de nouvelles autorisations à la politique `AmazonKeyspacesFullAccess` gérée afin de permettre aux clients se connectant à Amazon Keyspaces via des points de terminaison VPC d'interface d'accéder à l'instance Amazon EC2 afin de mettre à jour la table Amazon `system.peers` Keyspaces avec les informations réseau provenant du VPC.	3 octobre 2023
Mise à jour de la politique gérée par Amazon Keyspaces	Amazon Keyspaces a créé une nouvelle politique `AmazonKeyspacesReadOnlyAccess_v2` gérée pour permettre aux clients se connectant à Amazon Keyspaces via des points de terminaison VPC d'interface d'accéder à l'instance Amazon EC2 afin de mettre à jour la table Amazon `system.peers` Keyspaces avec les informations réseau provenant du VPC.	12 septembre 2023

Bonnes pratiques pour créer des connexions dans Amazon Keyspaces	Découvrez comment améliorer et optimiser les configurations des pilotes clients dans Amazon Keyspaces.	30 juin 2023
Les espaces de touches du système sont désormais documentés pour Amazon Keyspaces	Découvrez ce qui est stocké dans les espaces de touches du système et comment leur demander des informations utiles dans Amazon Keyspaces.	21 juin 2023
Amazon Keyspaces prend désormais en charge la réplication multirégionale	Amazon Keyspaces Multi-Region Replication vous aide à gérer des applications distribuées dans le monde entier en améliorant la tolérance aux pannes, la stabilité et la résilience.	5 juin 2023
Mise à jour de la politique gérée par Amazon Keyspaces	Amazon Keyspaces a ajouté de nouvelles autorisations à la politique `AmazonKeyspacesFullAccess` gérée afin de permettre à Amazon Keyspaces de créer un rôle lié à un service lorsqu'un administrateur crée un espace clé multirégional.	5 juin 2023
Support d'Amazon Keyspaces pour le mot clé `IN`	Amazon Keyspaces prend désormais en charge le `IN` mot clé dans `SELECT` les relevés.	25 avril 2023

Accès entre comptes pour Amazon Keyspaces et points de terminaison VPC d'interface	Découvrez comment implémenter l'accès entre comptes pour Amazon Keyspaces avec des points de terminaison VPC.	20 avril 2023
Support d'Amazon Keyspaces pour les horodatages côté client	Les horodatages côté client d'Amazon Keyspaces sont des horodatages au niveau des cellules compatibles avec Cassandra qui aident les applications distribuées à déterminer l'ordre des opérations d'écriture lorsque différents clients modifient les mêmes données.	14 mars 2023
Commencer à utiliser Amazon Keyspaces et les points de terminaison VPC d'interface	Dans ce step-by-step didacticiel, découvrez comment vous connecter à Amazon Keyspaces depuis un VPC.	1er mars 2023
Optimisation des coûts des tables Amazon Keyspaces	Les meilleures pratiques et conseils sont disponibles pour vous aider à identifier des stratégies permettant d'optimiser les coûts de vos tables Amazon Keyspaces existantes.	17 février 2023
`Murmur3Partitioner` C'est désormais la valeur par défaut	`Murmur3Partitioner` Il s'agit désormais du partitionneur par défaut dans Amazon Keyspaces.	17 novembre 2022

Amazon Keyspaces prend désormais en charge `Murmur3Partitioner`	`Murmur3Partitioner` Il est désormais disponible sur Amazon Keyspaces.	9 novembre 2022
Support mis à jour pour les chaînes vides et les valeurs blob	Amazon Keyspaces prend désormais également en charge les chaînes vides et les valeurs blob en tant que valeurs de colonnes de clustering.	19 octobre 2022
Amazon Keyspaces est désormais disponible dans AWS GovCloud (US)	Amazon Keyspaces est désormais disponible dans le cadre de la norme FedRAMP AWS GovCloud (US) Region et est soumis aux normes strictes de FedRAMP. Pour plus d'informations sur les points de terminaison disponibles, consultez la section Points de terminaison AWS GovCloud (US) Region FIPS.	4 août 2022
Surveillez les coûts de stockage des tables Amazon Keyspaces avec Amazon CloudWatch	Amazon Keyspaces vous permet désormais de surveiller et de suivre les coûts de stockage des tables au fil du `BillableTableSizeInBytes` CloudWatch temps grâce à cette métrique.	14 juin 2022
Amazon Keyspaces prend désormais en charge Terraform	Vous pouvez désormais utiliser Terraform pour effectuer des opérations de langage de définition de données (DDL) dans Amazon Keyspaces.	9 juin 2022

Support de la fonction Amazon Keyspaces `token`	Amazon Keyspaces vous aide désormais à optimiser les requêtes des applications en utilisant cette fonction. `token`	19 avril 2022
Intégration d'Amazon Keyspaces à Apache Spark	Amazon Keyspaces vous permet désormais de lire et d'écrire des données dans Apache Spark plus facilement en utilisant le connecteur open source Spark Cassandra.	19 avril 2022
Référence de l'API Amazon Keyspaces	Amazon Keyspaces prend en charge les opérations du plan de contrôle pour gérer les espaces de touches et les tables à l'aide du AWS SDK et. AWS CLI Le guide de référence de l'API décrit en détail les opérations du plan de contrôle prises en charge.	2 mars 2022
Comment résoudre les problèmes de configuration courants lors de l'utilisation d'Amazon Keyspaces.	Découvrez comment résoudre les problèmes de configura tion courants que vous pouvez rencontrer lors de l'utilisation d'Amazon Keyspaces.	22 novembre 2021
Amazon Keyspaces prend en charge Time to Live (TTL).	Amazon Keyspaces Time to Live (TTL) vous aide à simplifier la logique de votre application et à optimiser le prix du stockage en expirant automatiquement les données des tables.	18 octobre 2021

Migration de données vers Amazon Keyspaces à l'aide de DSBulk.	tep-by-step Tutoriel pour migrer des données d'Apache Cassandra vers Amazon Keyspaces à l'aide du DataStax Bulk Loader (DSBulk).	9 août 2021
Amazon Keyspaces prend en charge les entrées de point de terminaison VPC dans le tableau. `system.peers`	Amazon Keyspaces vous permet de renseigner le `system.peers` tableau avec les informations disponibles sur les points de terminais on VPC de l'interface afin d'améliorer l'équilibrage de charge et d'augmenter le débit de lecture/écriture.	29 juillet 2021
Mise à jour des politiques gérées par IAM pour prendre en charge les AWS KMS clés gérées par le client.	Les politiques gérées par IAM pour Amazon Keyspaces incluent désormais des autorisations permettant de répertorier et de consulter les clés AWS KMS gérées par les clients disponibles stockées dans. AWS KMS	1er juin 2021
Support d'Amazon Keyspaces pour les clés gérées par AWS KMS le client.	Amazon Keyspaces vous permet de prendre le contrôle des AWS KMS clés gérées par les clients qui sont stockées à des AWS KMS fins de chiffrement au repos.	1er juin 2021

Support d'Amazon Keyspaces pour la syntaxe JSON	Amazon Keyspaces vous aide à lire et à écrire des documents JSON plus facilement en prenant en charge la syntaxe JSON pour les opérations INSERT et SELECT.	21 janvier 2021
Support d'Amazon Keyspaces pour les colonnes statiques	Amazon Keyspaces vous permet désormais de mettre à jour et de stocker efficacement les données communes entre plusieurs lignes en utilisant des colonnes statiques.	9 novembre 2020
Publication générale du support de NoSQL Workbench pour Amazon Keyspaces	NoSQL Workbench est une application côté client qui vous permet de concevoir et de visualiser plus facilemen t des modèles de données non relationnels pour Amazon Keyspaces. Les clients NoSQL Workbench sont disponibl es pour Windows, macOS et Linux.	28 octobre 2020
Version préliminaire du support de NoSQL Workbench pour Amazon Keyspaces	NoSQL Workbench est une application côté client qui vous permet de concevoir et de visualiser plus facilemen t des modèles de données non relationnels pour Amazon Keyspaces. Les clients NoSQL Workbench sont disponibl es pour Windows, macOS et Linux.	5 octobre 2020

Nouveaux exemples de code pour l'accès programmatique à Amazon Keyspaces	Nous continuons d'ajouter des exemples de code pour l'accès programmatique à Amazon Keyspaces. Des exemples sont désormais disponibles pour les pilotes Java, Python, Go, C# et Perl Cassandra compatibles avec la version 3.11.2 d'Apache Cassandra.	17 juillet 2020
point-in-time Récupération d'Amazon Keyspaces (PITR)	Amazon Keyspaces propose désormais la point-in-time restauration (PITR) pour protéger vos tables contre les opérations d'écriture ou de suppression accidentelles en vous fournissant des sauvegardes continues des données de vos tables.	9 juillet 2020
Disponibilité générale d'Amazon Keyspaces	Avec Amazon Keyspaces, anciennement connu lors de la version préliminaire sous le nom d'Amazon Managed Apache Cassandra Service (MCS), vous pouvez utiliser le code Cassandra Query Language (CQL), les pilotes Cassandra sous licence Apache 2.0 et les outils de développement que vous utilisez déjà aujourd'hui.	23 avril 2020

Mise à l'échelle automatique d'Amazon Keyspaces	Amazon Keyspaces (pour Apache Cassandra) s'intègre à Application Auto Scaling pour vous aider à fournir une capacité de débit efficace pour des charges de travail variables en réponse au trafic réel des applications en ajustant automatiquement la capacité de débit.	23 avril 2020
Interface : points de terminaison de cloud privé virtuel (VPC) pour Amazon Keyspaces	Amazon Keyspaces propose une communication privée entre le service et votre VPC afin que le trafic réseau ne quitte pas le réseau Amazon.	16 avril 2020
Politiques d'accès basées sur des balises	Vous pouvez désormais utiliser des balises de ressources dans les politiques IAM pour gérer l'accès à Amazon Keyspaces.	8 avril 2020
Type de données du compteur	Amazon Keyspaces vous permet désormais de coordonner les incréments et les diminutions par rapport aux valeurs des colonnes à l'aide de compteurs.	7 avril 2020
Balisage des ressources	Amazon Keyspaces vous permet désormais d'étiqueter et de classer les ressources à l'aide de balises.	31 mars 2020

AWS CloudFormation soutien	Amazon Keyspaces vous aide désormais à automatiser la création et la gestion des ressources en utilisant. AWS CloudFormation	25 mars 2020
Support des rôles et politiques IAM et de l'authentification SigV4	Ajout d'informations sur la façon dont vous pouvez utiliser AWS Identity and Access Management (IAM) pour gérer les autorisations d'accès et mettre en œuvre des politiques de sécurité pour Amazon Keyspaces, ainsi que sur l'utilisation du plug-in d'authentification pour le pilote Java pour DataStax Cassandra afin d'accéder par programmation à Amazon Keyspaces à l'aide de rôles IAM et d'identités fédérées.	17 mars 2020
Mode de capacité de lecture/écriture	Amazon Keyspaces prend désormais en charge deux modes de capacité de débit de lecture/écriture. Le mode de capacité de lecture/écriture contrôle la façon dont le débit de lecture et d'écriture vous est facturé et la manière dont la capacité de débit des tables est gérée.	20 février 2020
Première version	Cette documentation couvre la version initiale d'Amazon Keyspaces (pour Apache Cassandra).	3 décembre 2019

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.