aws

Guia do Desenvolvedor

# Amazon Keyspaces (para Apache Cassandra)

# Amazon Keyspaces (para Apache Cassandra): Guia do Desenvolvedor

# Table of Contents

O que é o Amazon Keyspaces? ..... 1
Como funciona ..... 2
Arquitetura de alto nível ..... 2
Modelo de dados do Cassandra ..... 5
Como acessar o Amazon Keyspaces ..... 6
Casos de uso ..... 7
O que é CQL? ..... 8
Compare o Amazon Keyspaces com o Cassandra ..... 9
Diferenças funcionais com o Apache Cassandra ..... 10
APIs, operações e tipos de dados do Apache Cassandra ..... 11
Criação e exclusão assíncronas de espaços de chave e tabelas ..... 11
Autenticação e autorização ..... 11
Lote ..... 11
Configuração do cluster ..... 11
Conexões ..... 12
Palavra-chave `IN` ..... 12
Ajuste do throughput máximo de consultas CQL ..... 13
Coleções `FROZEN` ..... 13
Transações leves ..... 14
Balanceamento de carga ..... 14
Paginação ..... 14
Particionadores ..... 15
Instruções preparadas ..... 15
Intervalo de exclusão ..... 15
Tabelas de sistema ..... 15
Carimbos de data/hora ..... 16
APIs, operações, funções e tipos de dados compatíveis do Cassandra ..... 16
Suporte à API do Cassandra ..... 17
Suporte à API do ambiente de gerenciamento do Cassandra ..... 18
Suporte à API do plano de dados do Cassandra ..... 19
Suporte às funções do Cassandra ..... 19
Suporte ao tipo de dados do Cassandra ..... 20
Níveis de consistência suportados do Cassandra ..... 21
Níveis de consistência de gravação ..... 22

Níveis de consistência de leitura ... 23
Níveis de consistência sem suporte ... 24
Como acessar o Amazon Keyspaces ... 25
Conf AWS Identity and Access Management iguração ... 25
Inscreva-se para um Conta da AWS ... 25
Criar um usuário com acesso administrativo ... 26
Configurando o Amazon Keyspaces ... 27
Como usar o console ... 28
Usando AWS CloudShell ... 28
Obtendo permissões do IAM para AWS CloudShell ... 29
Interagindo com o Amazon Keyspaces usando AWS CloudShell ... 30
Conexão programática ... 31
Criação de credenciais ... 32
Service endpoints (Endpoints de serviço) ... 44
Utilizar o `cqlsh` ... 49
Usar a AWS CLI ... 55
Uso da API ... 60
Trabalhando com AWS SDKs ... 61
Uso de um driver de cliente Cassandra ... 62
Conectando-se a partir do Amazon EKS ... 92
Conexão com VPC endpoints ... 113
Pré-requisitos ... 113
Etapa 1: iniciar uma instância do Amazon EC2 ... 114
Etapa 2: configurar a instância do Amazon EC2 ... 116
Etapa 3: criar um endpoint da VPC para o Amazon Keyspaces ... 119
Etapa 4: configurar permissões para a conexão do endpoint da VPC ... 124
Etapa 5: configurar o monitoramento ... 127
Etapa 6: (opcional) práticas recomendadas para conexões ... 128
Etapa 7: (opcional) limpeza ... 131
Acesso entre contas ... 132
Acesso entre contas em uma VPC compartilhada ... 133
Acesso entre contas sem uma VPC compartilhada ... 136
Conceitos básicos ... 139
Pré-requisitos ... 139
Etapa 1: criar um espaço de chave e uma tabela ... 139
Como criar um espaço de chaves ... 140

Criar uma tabela ... 142
Etapa 2: Operações de CRUD ... 146
Criar ... 147
Leitura ... 148
Atualizar ... 151
Delete ... 152
Etapa 3: Limpar (opcional) ... 153
Excluir uma tabela ... 154
Como excluir um espaço de chave ... 155
Migração para o Amazon Keyspaces ... 157
Migrando de Cassandra ... 159
Compatibilidade ... 159
Estime os preços ... 160
Estratégia de migração ... 169
Migração offline ... 169
Ferramentas de migração ... 172
Como carregar dados usando cqlsh ... 172
Carregamento de dados usando o DSBulk ... 185
Exemplos de código ... 197
Ações ... 202
`CreateKeyspace` ... 203
`CreateTable` ... 207
`DeleteKeyspace` ... 214
`DeleteTable` ... 217
`GetKeyspace` ... 221
`GetTable` ... 225
`ListKeyspaces` ... 230
`ListTables` ... 234
`RestoreTable` ... 238
`UpdateTable` ... 242
Cenários ... 247
Conceitos básicos de keyspaces e tabelas ... 247
Bibliotecas e ferramentas ... 311
Bibliotecas e exemplos ... 311
Kit de ferramentas para desenvolvedores do Amazon Keyspaces (para Apache Cassandra) ... 311

Exemplos do Amazon Keyspaces (para Apache Cassandra) ..... 311
AWS Plugins de autenticação Signature Version 4 (SigV4) ..... 312
Repositórios de amostras e ferramentas para desenvolvedores em destaque ..... 312
Buffers de protocolo do Amazon Keyspaces ..... 312
Modelo AWS CloudFormation para criar o painel do Amazon CloudWatch para as métricas do Amazon Keyspaces (para Apache Cassandra) ..... 312
Como usar o Amazon Keyspaces (para Apache Cassandra) com AWS Lambda ..... 313
Como usar o Amazon Keyspaces (para Apache Cassandra) com Spring ..... 313
Como usar o Amazon Keyspaces (para Apache Cassandra) com Scala ..... 313
Como usar o Amazon Keyspaces (para Apache Cassandra) com AWS Glue ..... 313
Amazon Keyspaces (para Apache Cassandra) Cassandra Query Language (CQL) para conversor AWS CloudFormation ..... 314
Auxiliares do Amazon Keyspaces (para Apache Cassandra) para driver do Apache Cassandra para Java ..... 314
Demonstração de compressão rápida do Amazon Keyspaces (para Apache Cassandra ..... 314
Demonstração do codec do Amazon Keyspaces (para Apache Cassandra) e Amazon S3 ... 314
Integração com Apache Spark ..... 315
Pré-requisitos ..... 316
Etapa 1: Configurar o Amazon Keyspaces ..... 316
Etapa 2: Configurar o Apache Cassandra Spark Connector ..... 318
Etapa 3: Criar o arquivo de configuração do aplicativo ..... 320
Conecte-se com autenticação SigV4 ..... 320
Conecte-se com credenciais específicas do serviço ..... 321
Conecte-se com uma taxa fixa ..... 322
Etapa 4: Preparar os dados de origem e a tabela de destino ..... 323
Etapa 5: Gravar e ler dados do Amazon Keyspaces ..... 324
Solução de problemas ..... 327
Erros e avisos comuns ..... 329
Solução de problemas ..... 330
Erros gerais ..... 330
Erros gerais ..... 330
Conexões ..... 332
Erros na conexão com um endpoint do Amazon Keyspaces ..... 332
Gerenciamento de capacidade ..... 344
Erros de capacidade de tecnologia sem servidor ..... 345
Linguagem de definição de dados ..... 350

Erros de linguagem de definição de dados ..... 350
Gerenciamento de recursos de tecnologia sem servidor ..... 356
Armazenamento ..... 356
Modos de capacidade de leitura/gravação ..... 357
Modo de capacidade sob demanda ..... 357
Modo de capacidade de throughput provisionada ..... 360
Gerenciamento e visualização de modos de capacidade ..... 362
Considerações ao mudar os modos de capacidade ..... 363
Gerencie a capacidade de produção com escalabilidade automática ..... 363
Como funciona o ajuste de escala automático do Amazon Keyspaces ..... 364
Como o escalonamento automático funciona para tabelas multirregionais ..... 366
Observações de uso ..... 367
Como usar o console ..... 368
Usar SSL ..... 373
Uso da CLI ..... 379
Capacidade de expansão ..... 385
Estime o consumo de capacidade ..... 386
Intervalo de consultas ..... 387
Limitar consultas ..... 387
Digitalizações de tabela ..... 388
Transações leves ..... 389
Estime o consumo da capacidade de leitura e gravação com a Amazon CloudWatch ..... 389
Como trabalhar com o Amazon Keyspaces ..... 390
Como trabalhar com espaços de chaves ..... 390
espaços de chaves do sistema ..... 390
Como criar espaços de chaves ..... 397
Trabalhar com tabelas ..... 398
Criar tabelas ..... 398
Tabelas multirregionais ..... 399
Colunas estáticas ..... 400
Como trabalhar com linhas ..... 404
Como calcular o tamanho da linha ..... 405
Trabalhar com consultas ..... 409
`IN SELECT` Instrução ..... 409
Como ordenar resultados ..... 413
Como paginar resultados ..... 414

Como trabalhar com particionadores ... 415
Trabalhando com tags ... 417
Restrições de marcação ... 418
Operações de marcação ... 419
Relatórios de alocação de custos para Amazon Keyspaces ... 424
Práticas recomendadas ... 426
Design em NoSQL ... 427
NoSQL vs. RDBMS ... 428
Dois conceitos-chave ... 428
Abordagem geral ... 429
Conexões ... 430
Como funcionam ... 430
Como configurar conexões ... 431
Conexões de endpoint da VPC ... 433
Como monitorar conexões ... 434
Como lidar com erros de conexão ... 435
Modelagem de dados ... 436
Design de chave de partição ... 436
Otimização de custo ... 439
Avaliar seus custos no nível da tabela ... 439
Avaliar o modo de capacidade de sua tabela ... 441
Avalie as configurações do Application Auto Scaling da sua tabela ... 446
Identifique seus recursos não utilizados ... 453
Avaliar seus padrões de uso de tabelas ... 459
Avaliar sua capacidade provisionada para o provisionamento do tamanho certo ... 460
NoSQL Workbench ... 470
Baixar ... 471
Conceitos básicos ... 471
Modelador de dados ... 473
Como criar um modelo de dados ... 474
Como editar um modelo de dados ... 476
Visualizador de dados ... 478
Visualização de um modelo de dados ... 478
Exibição agregada ... 480
Confirmar um modelo de dados ... 482
Antes de começar ... 483

Como conectar-se com credenciais específicas do serviço ... 484
Como conectar-se com credenciais do IAM ... 487
Como usar uma conexão salva ... 490
Apache Cassandra ... 490
Modelos de dados de amostra ... 493
Modelo de dados de funcionários ... 493
Modelo de dados de transações com cartão de crédito ... 493
Modelo de dados de operações de companhias aéreas ... 494
Histórico de versões ... 494
Replicação multirregional ... 496
Benefícios ... 496
Modos de capacidade e preços ... 497
Como funciona ... 498
Como funciona ... 498
Resolução de conflitos ... 500
Recuperação de desastres ... 500
Permissões do IAM ... 501
Integração com o PITR ... 502
Integração com AWS serviços ... 502
Observações de uso ... 503
Como usar a replicação multirregional ... 505
Usar o console ... 506
Usar SSL ... 512
Usando o AWS CLI ... 520
Recuperação pontual ... 530
Como funciona ... 530
Como habilitar a PITR ... 531
Restaurar permissões ... 534
Backups contínuos ... 536
Restaurar as configurações ... 537
PITR e tabelas criptografadas ... 539
PITR e tabelas multirregionais ... 539
Tempo de restauração da tabela ... 540
Integração com serviços da AWS ... 502
Restaurar uma tabela para um ponto no tempo ... 541
Antes de começar ... 541

Como restaurar uma tabela para um ponto no tempo (console) ..... 541
Como restaurar uma tabela para um ponto no tempo com o AWS CLI ..... 543
Como restaurar uma tabela para um ponto no tempo com CQL ..... 545
Como restaurar uma tabela excluída com o AWS CLI ..... 548
Como restaurar uma tabela excluída com CQL ..... 548
Dados expirados com a vida útil ..... 550
Como funciona ..... 551
Valor de TTL padrão ..... 551
Valores de TTL personalizados ..... 552
Como habilitar a TTL ..... 553
Integração com serviços da AWS ..... 553
Como usar a vida útil ..... 554
Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas (console) ..... 554
Para atualizar as configurações de vida útil (TTL) padrão em tabelas existentes (console) .. 555
Para desabilitar as configurações de vida útil (TTL) padrão em tabelas existentes (console) ..... 555
Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas usando CQL ..... 556
Para usar `ALTER TABLE` para editar as configurações de tempo de vida (TTL) padrão usando CQL ..... 556
Como habilitar a vida útil (TTL) em novas tabelas usando propriedades personalizadas ..... 557
Como habilitar a vida útil (TTL) em tabelas existentes usando propriedades personalizadas ..... 557
Para usar `INSERT` para editar as configurações de tempo de vida (TTL) personalizadas usando CQL ..... 557
Para usar `UPDATE` para editar as configurações de tempo de vida (TTL) personalizadas usando CQL ..... 558
Carimbos de data/hora do lado do cliente ..... 560
Como funciona ..... 561
Carimbos de data/hora do lado do cliente no Amazon Keyspaces ..... 561
Integração com serviços da AWS ..... 562
Como usar carimbos de data/hora no lado do cliente ..... 562
Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (console) ..... 563
Como ativar os carimbos de data/hora do lado do cliente em tabelas existentes (console) .. 564

Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CQL) ... 564
Como ativar carimbos de data/hora do lado do cliente para tabelas existentes usando ALTER TABLE (CQL) ... 565
Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CLI) ... 565
Como ativar carimbos de data/hora do lado do cliente em uma tabela existente (CLI) ... 567
Como usar carimbos de data/hora do lado do cliente em instruções de Linguagem de Manipulação de Dados (DML) ... 569
Recursos da AWS CloudFormation ... 570
Amazon Keyspaces e modelos AWS CloudFormation ... 570
Saiba mais sobre o AWS CloudFormation ... 570
Monitoramento do Amazon Keyspaces ... 571
Monitoramento com CloudWatch ... 572
Uso de métricas do ... 573
Métricas e dimensões ... 574
Criar alarmes ... 596
Fazendo login com CloudTrail ... 597
Configurando entradas do arquivo de log no CloudTrail ... 598
Informações sobre DDL em CloudTrail ... 599
Informações de DML em CloudTrail ... 599
Noções básicas sobre entradas de arquivos de log do ... 600
Segurança ... 612
Proteção de dados ... 613
Criptografia em repouso ... 614
Criptografia em trânsito ... 635
Privacidade do tráfego entre redes ... 636
AWS Identity and Access Management ... 637
Público ... 638
Autenticando com identidades ... 639
Gerenciando acesso usando políticas ... 642
Como o Amazon Keyspaces funciona com o IAM ... 644
Exemplos de políticas baseadas em identidade ... 650
Políticas gerenciadas pela AWS ... 657
Solução de problemas ... 666
Usar perfis vinculados ao serviço ... 670
Validação de conformidade ... 677

Resiliência ..... 679
Segurança da infraestrutura ..... 680
Usar VPC endpoints da interface do ..... 681
Análise de configuração e vulnerabilidade no Amazon Keyspaces ..... 687
Práticas recomendadas de segurança ..... 688
Práticas recomendadas de segurança preventiva ..... 688
Práticas recomendadas de segurança de detecção ..... 690
Referência da linguagem CQL ..... 693
Elementos de idiomas ..... 693
Identificadores ..... 693
Constantes ..... 694
Termos ..... 694
Tipos de dados ..... 694
Codificação JSON dos tipos de dados do Amazon Keyspaces ..... 698
Instruções DDL ..... 702
Keyspaces ..... 703
Tabelas ..... 705
Instruções DML ..... 718
SELECT ..... 719
INSERT ..... 722
UPDATE ..... 724
DELETE ..... 725
Funções incorporadas ..... 726
Funções escalares ..... 726
Cotas ..... 729
Service Quotas do Amazon Keyspaces ..... 729
Aumentar ou diminuir throughput (para tabelas provisionadas) ..... 735
Aumentar throughput provisionado ..... 735
Diminuir throughput provisionado ..... 735
Criptografia em repouso do Amazon Keyspaces ..... 735
Histórico do documento ..... 736
..... dccxlvii

# O que é o Amazon Keyspaces (para Apache Cassandra)?

O Amazon Keyspaces (para Apache Cassandra) é um serviço de banco de dados compatível com Apache Cassandra, escalável, de alta disponibilidade e gerenciado. Com o Amazon Keyspaces, você não precisa provisionar, corrigir ou gerenciar servidores, nem instalar, manter ou operar software.

O Amazon Keyspaces possui tecnologia sem servidor, então você paga somente pelos recursos que usa, e o serviço aumenta e diminui automaticamente as tabelas em resposta ao tráfego do aplicativo. Você pode criar aplicativos que atendem a milhares de solicitações por segundo com throughput e armazenamento praticamente ilimitados.

> **Note**
>
> O Apache Cassandra é um datastore de coluna ampla de código aberto, projetado para lidar com grandes quantidades de dados. Para obter mais informações, consulte Apache Cassandra.

O Amazon Keyspaces facilita a migração, a execução e a escalabilidade das workloads do Cassandra no Nuvem AWS. Com apenas alguns cliques no AWS Management Console ou algumas linhas de código, você pode criar espaços chave e tabelas no Amazon Keyspaces, sem implantar nenhuma infraestrutura ou instalar software.

Com o Amazon Keyspaces, você pode executar suas cargas de trabalho existentes do Cassandra AWS usando o mesmo código do aplicativo Cassandra e as mesmas ferramentas de desenvolvedor que você usa atualmente.

Para obter uma lista de endpoints disponíveis Regiões da AWS , consulte Endpoints de serviço para Amazon Keyspaces.

Recomendamos que você comece lendo as seguintes seções:

Tópicos

- Amazon Keyspaces: como funciona
- Casos de uso do Amazon Keyspaces
- O que é Cassandra Query Language (CQL)?

# Amazon Keyspaces: como funciona

O Amazon Keyspaces remove a sobrecarga administrativa do gerenciamento do Cassandra. Para entender por que, é útil começar com a arquitetura do Cassandra e depois compará-la com o Amazon Keyspaces.

Tópicos

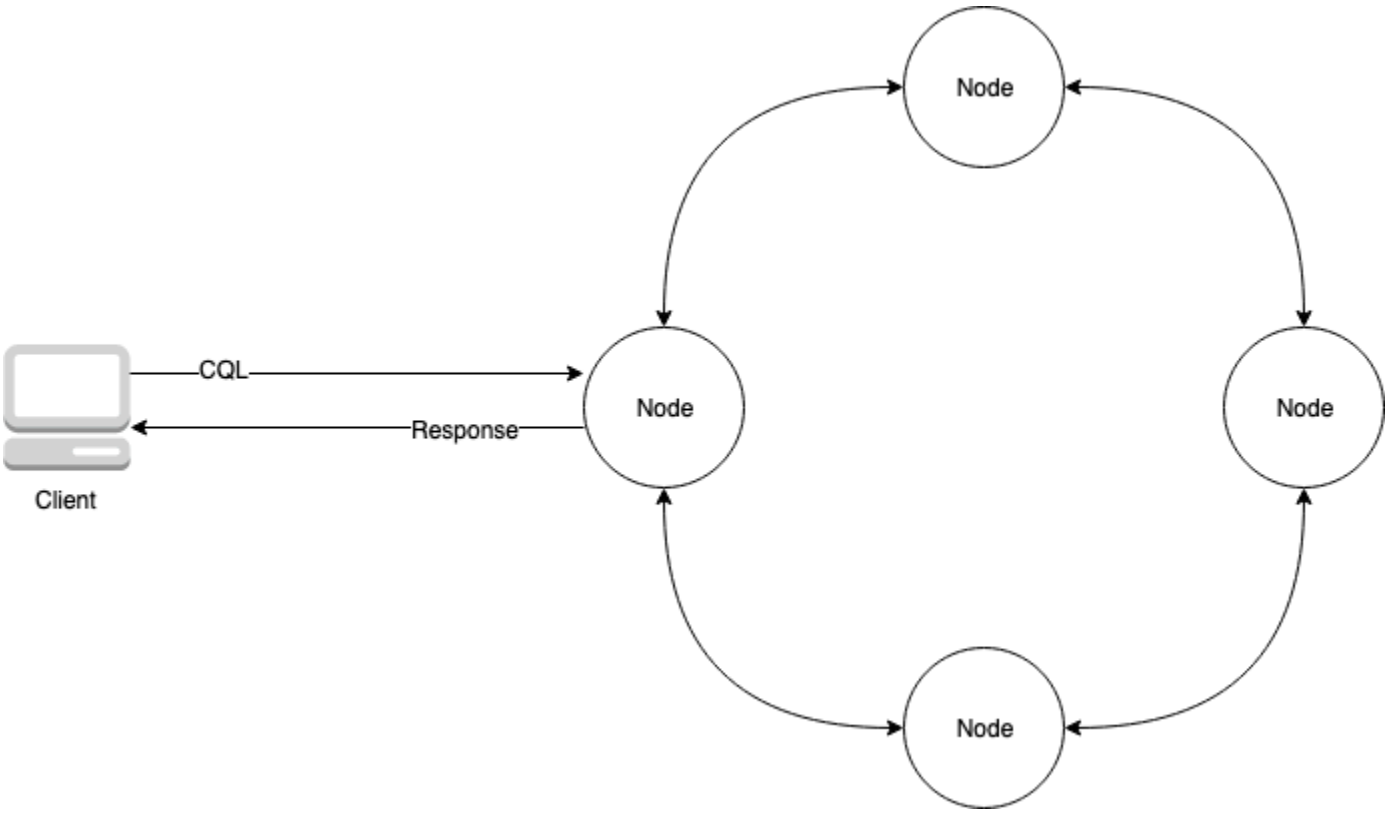

- Arquitetura de alto nível: Apache Cassandra versus Amazon Keyspaces
- Modelo de dados do Cassandra
- Como acessar o Amazon Keyspaces a partir de um aplicativo


## Arquitetura de alto nível: Apache Cassandra versus Amazon Keyspaces

O Apache Cassandra tradicional é implantado em um cluster composto de um ou mais nós. Você é responsável por gerenciar cada nó e adicionar e remover nós à medida que seu cluster é escalado.

Um programa cliente acessa o Cassandra conectando-se a um dos nós e emitindo instruções Cassandra Query Language (CQL). O CQL é semelhante ao SQL, a linguagem popular usada em bancos de dados relacionais. Embora o Cassandra não seja um banco de dados relacional, o CQL fornece uma interface familiar para consultar e manipular dados no Cassandra.

O diagrama a seguir mostra um cluster Apache Cassandra simples, com quatro nós.




Uma implantação de produção do Cassandra pode consistir em centenas de nós, executados em centenas de computadores físicos em um ou mais datacenters físicos. Isso pode causar uma sobrecarga operacional para os desenvolvedores de aplicativos que precisam provisionar, corrigir e gerenciar servidores, além de instalar, manter e operar o software.

Com o Amazon Keyspaces (para Apache Cassandra), você não precisa provisionar, corrigir ou gerenciar servidores, para que possa se concentrar na criação de aplicativos melhores. O Amazon Keyspaces oferece dois modos de capacidade de throughput para leituras e gravações: sob demanda e provisionada. Você pode escolher o modo de capacidade de throughput da sua tabela para otimizar o preço das leituras e gravações com base na previsibilidade e na variabilidade da sua workload.

Com o modo sob demanda, você paga apenas pelas leituras e gravações que seu aplicativo realmente executa. Você não precisa especificar com antecedência a capacidade de throughput de sua tabela. O Amazon Keyspaces acomoda o tráfego do seu aplicativo quase instantaneamente à medida que aumenta ou diminui, o que o torna uma boa opção para aplicativos com tráfego imprevisível.

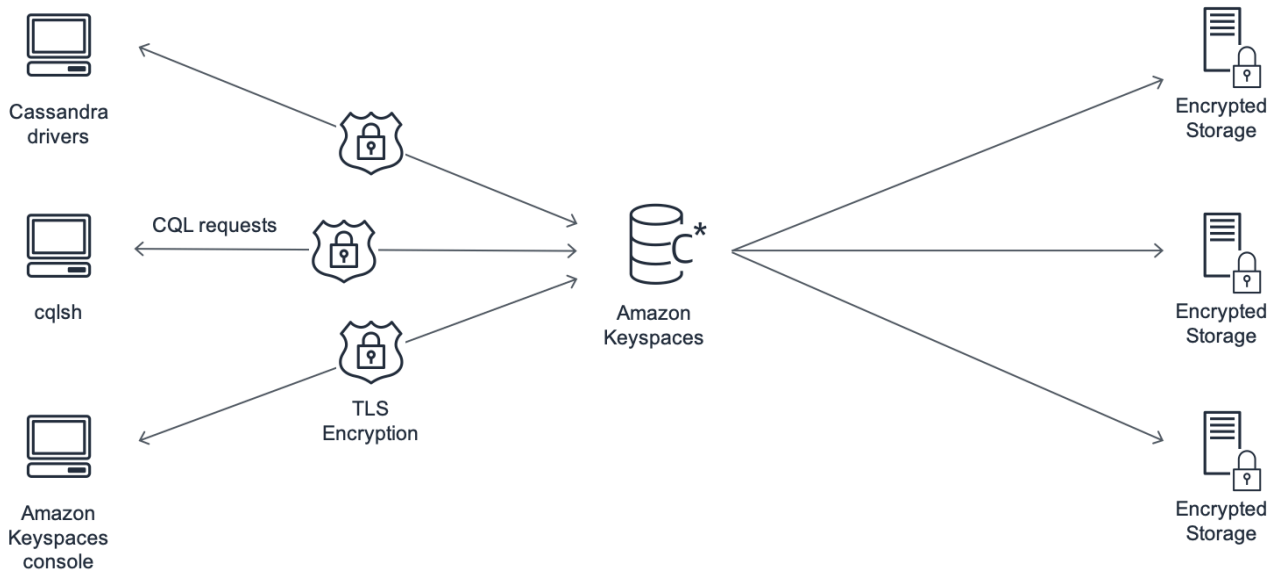
O modo de capacidade provisionada ajuda você a otimizar o preço do throughput se você tiver tráfego previsível de aplicativos e puder prever os requisitos de capacidade da sua tabela com

antecedência. Com o modo de capacidade provisionada, você especifica o número de leituras e gravações por segundo que espera que seu aplicativo execute. Você pode aumentar e diminuir automaticamente a capacidade provisionada de sua tabela ativando o escalonamento automático.

Você pode alterar o modo de capacidade da sua tabela uma vez por dia à medida que aprende mais sobre os padrões de tráfego da sua workload ou se espera ter um grande aumento no tráfego, como devido a um grande evento que, segundo sua previsão, gerará muito tráfego na tabela. Para obter mais informações sobre unidades de capacidade de leitura e gravação, consulte the section called "Modos de capacidade de leitura/gravação".

O Amazon Keyspaces (para Apache Cassandra) armazena três cópias dos seus dados em várias Zonas de disponibilidade para maior durabilidade e alta disponibilidade. Além disso, você se contará com um datacenter e uma arquitetura de rede criados para atender aos requisitos das organizações com as maiores exigências de segurança. A criptografia em repouso é ativada automaticamente quando você cria uma nova tabela do Amazon Keyspaces e todas as conexões de clientes exigem Transport Layer Security (TLS). Recursos adicionais AWS de segurança incluem monitoramento e AWS Identity and Access Managementendpoints de nuvem privada virtual (VPC). Para obter uma visão geral de todos os recursos de segurança disponíveis, consulte Segurança.

O diagrama a seguir mostra a arquitetura do Amazon Keyspaces.




Um programa cliente acessa o Amazon Keyspaces conectando-se a um endpoint predeterminado (nome do host e número da porta) e emitindo instruções CQL. Para obter uma lista de endpoints disponíveis, consulte the section called "Service endpoints (Endpoints de serviço)".

# Modelo de dados do Cassandra

A forma como você modela seus dados para seu caso de negócios é fundamental para alcançar o desempenho ideal do Amazon Keyspaces. Um modelo de dados ruim pode degradar significativamente o desempenho.

Embora o CQL seja semelhante ao SQL, os backends do Cassandra e dos bancos de dados relacionais são muito diferentes e devem ser abordados de forma diferente. A seguir estão algumas das questões mais importantes a serem consideradas:

Armazenamento

Você pode visualizar seus dados do Cassandra em tabelas, com cada linha representando um registro e cada coluna um campo dentro desse registro.

Design da tabela: consulta primeiro

Não há `JOIN`s no CQL. Portanto, você deve criar suas tabelas com a forma dos seus dados e como você precisa acessá-los para seus casos de uso comercial. Isso pode resultar em desnormalização com dados duplicados. Você deve projetar cada uma de suas tabelas especificamente para um padrão de acesso específico.

Partições

Seus dados são armazenados em partições no disco. O número de partições em que seus dados são armazenados e como eles são distribuídos entre as partições é determinado pela sua chave de partição. A forma como você define sua chave de partição pode ter um impacto significativo no desempenho de suas consultas. Para ver as práticas recomendadas, consulte the section called “Design de chave de partição”.

Chave primária

No Cassandra, os dados são armazenados como um par de chave/valor. Para esse fim, cada tabela do Cassandra deve ter uma chave primária, que é a chave para cada linha na tabela. A chave primária é composta por uma chave de partição necessária e uma ou mais colunas de clustering opcionais. Os dados que compõem a chave primária devem ser exclusivos em todos os registros de uma tabela.

- Chave de partição — A parte da chave de partição da chave primária é necessária e determina em qual partição do cluster os dados são armazenados. A chave de partição pode ser uma única coluna ou um valor composto formado por duas ou mais colunas. Você usaria uma chave

de partição composta se uma chave de partição de coluna única resultasse em uma única partição ou em poucas partições com a maioria dos dados e, portanto, suportasse a maioria das operações de E/S do disco.

- Coluna de clustering — A parte opcional da coluna de clustering de sua chave primária determina como os dados são agrupados e classificados em cada partição. Se você incluir uma coluna de clustering em sua chave primária, a coluna de clustering poderá ter uma ou mais colunas. Se houver várias colunas na coluna de clustering, a ordem de classificação será determinada pela ordem em que as colunas são listadas na coluna de clustering, da esquerda para a direita.

Para obter mais informações sobre o design do NoSQL e o Amazon Keyspaces, consulte. the section called “Design em NoSQL” Para obter mais informações sobre Amazon Keyspaces e modelagem de dados, consulte. the section called “Modelagem de dados”

## Como acessar o Amazon Keyspaces a partir de um aplicativo

O Amazon Keyspaces (para Apache Cassandra) implementa a API Apache Cassandra Query Language (CQL), para que você possa usar os drivers CQL e Cassandra que você já usa. Atualizar seu aplicativo é tão fácil quanto atualizar o driver ou a configuração `cqlsh` do Cassandra para apontar para o endpoint do serviço Amazon Keyspaces. Para obter mais informações sobre as credenciais necessárias, consultethe section called “Credenciais do IAM para autenticação AWS ”.

> **Note**
>
> Para ajudar você a começar, você pode encontrar exemplos de end-to-end código de conexão com o Amazon Keyspaces usando vários drivers de cliente do Cassandra no repositório de exemplos de código do Amazon Keyspaces em. GitHub

Considere o seguinte programa Python, que se conecta a um cluster do Cassandra e consulta uma tabela.

```
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)
```

Para executar o mesmo programa no Amazon Keyspaces, você precisa:

- Adicionar o endpoint e a porta do cluster: por exemplo, o host pode ser substituído por um endpoint de serviço, como `cassandra.us-east-2.amazonaws.com` e o número da porta por: `9142`.
- Adicionar a configuração TLS/SSL: para obter mais informações sobre como adicionar a configuração TLS/SSL para se conectar ao Amazon Keyspaces usando um driver Python do cliente Cassandra, consulte Como usar um driver de cliente Cassandra Python para acessar o Amazon Keyspaces programaticamente.

# Casos de uso do Amazon Keyspaces

A seguir estão apenas algumas das maneiras pelas quais você pode usar o Amazon Keyspaces:

- Crie aplicativos que exijam baixa latência — Processe dados em alta velocidade para aplicativos que exigem single-digit-millisecond latência, como manutenção de equipamentos industriais, monitoramento comercial, gerenciamento de frotas e otimização de rotas.
- Crie aplicativos usando tecnologias de código aberto — Crie aplicativos AWS usando APIs e drivers de código aberto do Cassandra que estão disponíveis para uma ampla variedade de linguagens de programação, como Java, Python, Ruby, Microsoft.NET, Node.js, PHP, C++, Perl e Go. Para obter exemplos de código, consulte Bibliotecas e ferramentas.
- Mova suas workloads do Cassandra para a nuvem – Gerenciar as tabelas do Cassandra sozinho é demorado e caro. Com o Amazon Keyspaces, você pode configurar, proteger e escalar tabelas do Cassandra sem gerenciar a Nuvem AWS infraestrutura. Para ter mais informações, consulte Gerenciamento de recursos de tecnologia sem servidor.

# O que é Cassandra Query Language (CQL)?

O Cassandra Query Language (CQL) é a linguagem principal para comunicação com o Apache Cassandra. O Amazon Keyspaces (para Apache Cassandra) é compatível com a API CQL 3.x (com compatibilidade com versões anteriores à versão 2.x).

Para executar consultas CQL, você pode realizar um dos seguintes procedimentos:

- Use o editor CQL no AWS Management Console.
- Uso AWS CloudShell e expansão de dinheiro.
- Execute-os no cliente `cqlsh`.
- Execute-os programaticamente usando um driver de cliente Cassandra licenciado pelo Apache 2.0.

Além disso, você pode acessar o Amazon Keyspaces usando o AWS SDK e o. AWS Command Line Interface

Para obter mais informações sobre como usar esses métodos para acessar o Amazon Keyspaces, consulte Como acessar o Amazon Keyspaces (para Apache Cassandra).

Para obter mais informações sobre CQL, consulte Referência de idiomas CQL para Amazon Keyspaces (para Apache Cassandra).

# Como o Amazon Keyspaces (para Apache Cassandra) se compara ao Apache Cassandra?

Para estabelecer uma conexão com o Amazon Keyspaces, você pode usar um endpoint de AWS serviço público ou um endpoint privado usando endpoints de interface VPC () na Amazon AWS PrivateLink Virtual Private Cloud. Dependendo do endpoint usado, o Amazon Keyspaces pode aparecer para o cliente de uma das seguintes formas.

AWS conexão de endpoint de serviço

> Essa é uma conexão estabelecida em qualquer endpoint público. Nesse caso, o Amazon Keyspaces aparece como um cluster Apache Cassandra 3.11.2 de nove nós para o cliente.

Interface: conexão de endpoint VPC

> Essa é uma conexão privada estabelecida usando uma interface VPC endpoint. Nesse caso, o Amazon Keyspaces aparece como um cluster Apache Cassandra 3.11.2 de três nós para o cliente.

Independentemente do tipo de conexão e do número de nós visíveis para o cliente, o Amazon Keyspaces fornece taxa de transferência e armazenamento praticamente ilimitados. Para fazer isso, o Amazon Keyspaces mapeia os nós para balanceadores de carga que direcionam suas consultas para uma das muitas partições de armazenamento subjacentes. Para obter mais informações sobre conexões, consulte the section called “Como funcionam”.

O Amazon Keyspaces armazena dados em partições. Uma partição é uma alocação de armazenamento para uma tabela, apoiada por unidades de estado sólido (SSDs). O Amazon Keyspaces replica automaticamente seus dados em várias zonas de disponibilidade dentro de uma Região da AWS para maior durabilidade e alta disponibilidade. Conforme sua taxa de transferência ou necessidades de armazenamento aumentam, o Amazon Keyspaces gerencia o gerenciamento de partições para você e provisiona automaticamente as partições adicionais necessárias. Para ter mais informações, consulte the section called “Armazenamento”.

O Amazon Keyspaces oferece suporte a todas as operações de plano de dados do Cassandra comumente usadas, como criar espaços de chaves e tabelas, ler dados e gravar dados. O Amazon Keyspaces não tem servidor, então você não precisa provisionar, corrigir ou gerenciar servidores. Você também não precisa instalar, manter nem operar software. Como resultado, no Amazon

Keyspaces, você não precisa usar as operações de API do plano de controle do Cassandra para gerenciar configurações de clusters e nós.

O Amazon Keyspaces configura automaticamente configurações como fator de replicação e nível de consistência para oferecer alta disponibilidade, durabilidade e desempenho. single-digit-millisecond Para obter ainda mais resiliência e leituras locais de baixa latência, o Amazon Keyspaces oferece replicação em várias regiões.

Tópicos

- Diferenças funcionais: Amazon Keyspaces versus Apache Cassandra
- APIs, operações, funções e tipos de dados compatíveis do Cassandra no Amazon Keyspaces
- Níveis de consistência compatíveis do Apache Cassandra no Amazon Keyspaces

# Diferenças funcionais: Amazon Keyspaces versus Apache Cassandra

A seguir estão as diferenças funcionais entre o Amazon Keyspaces e o Apache Cassandra.

Tópicos

- APIs, operações e tipos de dados do Apache Cassandra
- Criação e exclusão assíncronas de espaços de chave e tabelas
- Autenticação e autorização
- Lote
- Configuração do cluster
- Conexões
- Palavra-chave IN
- Ajuste do throughput máximo de consultas CQL
- Coleções FROZEN
- Transações leves
- Balanceamento de carga
- Paginação
- Particionadores
- Instruções preparadas

- Intervalo de exclusão
- Tabelas de sistema
- Carimbos de data/hora

## APIs, operações e tipos de dados do Apache Cassandra

O Amazon Keyspaces oferece suporte a todas as operações de plano de dados do Cassandra comumente usadas, como criar espaços de chaves e tabelas, ler dados e gravar dados. Para ver o que é compatível no momento, consulte APIs, operações, funções e tipos de dados compatíveis do Cassandra no Amazon Keyspaces.

## Criação e exclusão assíncronas de espaços de chave e tabelas

O Amazon Keyspaces executa operações de linguagem de definição de dados (DDL), como criar e excluir espaços de chave e tabelas, de forma assíncrona. Para saber como monitorar o status de criação de recursos, consulte the section called “Como criar espaços de chaves” e the section called “Criar tabelas”. Para obter uma lista de instruções DDL na referência da linguagem CQL, consulte the section called “Instruções DDL”.

## Autenticação e autorização

O Amazon Keyspaces (para Apache Cassandra) usa AWS Identity and Access Management (IAM) para autenticação e autorização de usuários e oferece suporte às políticas de autorização equivalentes às do Apache Cassandra. Dessa forma, o Amazon Keyspaces não oferece suporte aos comandos de configuração de segurança do Apache Cassandra.

## Lote

O Amazon Keyspaces oferece suporte a comandos em lote não registrados com até 30 comandos no lote. Somente comandos INSERT, UPDATE ou DELETE incondicionais são permitidos em um lote. Os lotes registrados não são compatíveis.

## Configuração do cluster

O Amazon Keyspaces é de tecnologia sem servidor, portanto, não há clusters, hosts ou máquinas virtuais Java (JVMs) para configurar. As configurações do Cassandra de compactação, compressão, armazenamento em cache, coleta de resíduos e filtragem de bloom não são aplicáveis ao Amazon Keyspaces e são ignoradas quando especificadas.

## Conexões

Você pode usar os drivers existentes do Cassandra para se comunicar com o Amazon Keyspaces, mas precisa configurar os drivers de forma diferente. O Amazon Keyspaces suporta até 3.000 consultas CQL por conexão TCP por segundo, mas não há limite no número de conexões que um driver pode estabelecer.

A maioria dos drivers de código aberto do Cassandra estabelece um pool de conexões com o Cassandra e balanceia a carga das consultas sobre esse pool de conexões. O Amazon Keyspaces expõe 9 endereços IP de mesmo nível aos drivers, e o comportamento padrão da maioria dos drivers é estabelecer uma única conexão com cada endereço IP de mesmo nível. Portanto, o throughput máximo de consultas CQL de um driver usando as configurações padrão é de 27.000 consultas CQL por segundo.

Para aumentar esse número, recomendamos aumentar o número de conexões por endereço IP que o driver mantém no pool de conexões. Por exemplo, definir o máximo de conexões por endereço IP como 2 dobra o throughput máximo do driver para 54.000 consultas CQL por segundo.

Como prática recomendada, recomendamos configurar os drivers para usar 500 consultas CQL por segundo por conexão para permitir a sobrecarga e melhorar a distribuição. Nesse cenário, o planejamento de 18.000 consultas CQL por segundo requer 36 conexões. A configuração do driver para 4 conexões em 9 endpoints fornece 36 conexões executando 500 solicitações por segundo. Para obter mais informações sobre as práticas recomendadas para conexões, consulte the section called “Conexões”.

Ao se conectar com endpoints da VPC, pode haver menos endpoints disponíveis. Isso significa que você precisa aumentar o número de conexões na configuração do driver. Para obter mais informações sobre práticas recomendadas para conexões de VPC, consulte. the section called “Conexões de endpoint da VPC”

## Palavra-chave `IN`

O Amazon Keyspaces suporta a palavra-chave `IN` na instrução `SELECT`. O `IN` não é compatível com `UPDATE` e `DELETE`. Ao usar a palavra-chave `IN` na instrução `SELECT`, os resultados da consulta são retornados na ordem em que as chaves são apresentadas na instrução `SELECT`. No Cassandra, os resultados são ordenados lexicograficamente.

Ao usar `ORDER BY`, a reordenação completa com paginação desativada não é compatível e os resultados são ordenados em uma página. As consultas slice não são compatíveis com a

palavra-chave `IN`. Os `TOKENS` não são compatíveis com a palavra-chave `IN`. O Amazon Keyspaces processa consultas com a palavra-chave `IN` criando subconsultas. Cada subconsulta conta como uma conexão para o limite de 3.000 consultas CQL por conexão TCP por segundo. Para ter mais informações, consulte the section called “`IN SELECT` Instrução”.

## Ajuste do throughput máximo de consultas CQL

O Amazon Keyspaces suporta até 3.000 consultas CQL por conexão TCP por segundo, mas não há limite no número de conexões que um driver pode estabelecer.

A maioria dos drivers de código aberto do Cassandra estabelece um pool de conexões com o Cassandra e balanceia a carga das consultas sobre esse pool de conexões. O Amazon Keyspaces expõe 9 endereços IP de mesmo nível aos drivers, e o comportamento padrão da maioria dos drivers é estabelecer uma única conexão com cada endereço IP de mesmo nível. Portanto, o throughput máximo de consultas CQL de um driver usando as configurações padrão será de 27.000 consultas CQL por segundo.

Para aumentar esse número, recomendamos aumentar o número de conexões por endereço IP que o driver mantém no pool de conexões. Por exemplo, definir o máximo de conexões por endereço IP como 2 dobrará o throughput máximo do seu driver para 54.000 consultas CQL por segundo.

Para obter mais informações sobre as práticas recomendadas para conexões, consulte the section called “Conexões”.

Ao se conectar com VPC endpoints, menos endpoints estão disponíveis. Isso significa que você precisa aumentar o número de conexões na configuração do driver. Para obter mais informações sobre as melhores práticas para conexões de endpoints VPC, consulte. the section called “Conexões de endpoint da VPC”

## Coleções `FROZEN`

A palavra-chave `FROZEN` no Cassandra serializa vários componentes de um tipo de dados de coleção em um único valor imutável que é tratado como um `BLOB`. As instruções `INSERT` e `UPDATE` sobrescrevem toda a coleção.

Por padrão, o Amazon Keyspaces suporta até cinco níveis de aninhamento para coleções congeladas. Para ter mais informações, consulte the section called “Service Quotas do Amazon Keyspaces”.

O Amazon Keyspaces não suporta comparações de desigualdade que usam toda a coleção congelada em uma condição `UPDATE` ou instrução `SELECT`. O comportamento de coleções e coleções congeladas é o mesmo no Amazon Keyspaces.

Quando você usa coleções congeladas com timestamps no lado do cliente, caso o timestamp de uma operação de gravação seja o mesmo que o timestamp de uma coluna existente não expirada ou não delimitada, o Amazon Keyspaces não realiza comparações. Em vez disso, ele permite que o servidor determine o gravador mais recente, e o gravador mais recente vence.

Para obter mais informações sobre coleções congeladas, consulte the section called “Tipos de coleção”.

## Transações leves

O Amazon Keyspaces (para Apache Cassandra) oferece suporte total à funcionalidade de comparação e configuração nos comandos INSERT, UPDATE e DELETE, conhecidos como transações leves (LWTs) no Apache Cassandra. Como uma oferta de tecnologia sem servidor, o Amazon Keyspaces (para Apache Cassandra) fornece desempenho consistente em qualquer escala, inclusive para transações leves. Com o Amazon Keyspaces, não há penalidade de desempenho pelo uso de transações leves.

## Balanceamento de carga

As entradas da tabela `system.peers` correspondem aos balanceadores de carga do Amazon Keyspaces. Para obter melhores resultados, recomendamos usar uma política de balanceamento de carga round robin e ajustar o número de conexões por IP para atender às necessidades do seu aplicativo.

## Paginação

O Amazon Keyspaces pagina os resultados com base no número de linhas que lê para processar uma solicitação, não no número de linhas retornadas no conjunto de resultados. Como resultado, algumas páginas podem conter menos linhas do que você especifica em TAMANHO DA PÁGINA para consultas filtradas. Além disso, o Amazon Keyspaces pagina os resultados automaticamente após a leitura de 1 MB de dados para fornecer aos clientes um desempenho de leitura consistente de um dígito em milissegundos. Para ter mais informações, consulte the section called “Como paginar resultados”.

## Particionadores

O particionador padrão no Amazon Keyspaces é o `Murmur3Partitioner` compatível com o Cassandra. Além disso, você tem a opção de usar o `DefaultPartitioner` do Amazon Keyspaces ou o `RandomPartitioner` compatível com Cassandra.

Com o Amazon Keyspaces, você pode alterar com segurança o particionador da sua conta sem precisar recarregar seus dados do Amazon Keyspaces. Depois que a alteração da configuração for concluída, o que leva aproximadamente 10 minutos, os clientes verão a nova configuração do particionador automaticamente na próxima vez que se conectarem. Para ter mais informações, consulte the section called “Como trabalhar com particionadores”.

## Instruções preparadas

O Amazon Keyspaces oferece suporte ao uso de instruções preparadas para operações de linguagem de manipulação de dados (DML), como leitura e gravação de dados. Atualmente, o Amazon Keyspaces não oferece suporte ao uso de instruções preparadas para operações de linguagem de definição de dados (DDL), como a criação de tabelas e espaços de chaves. As operações de DDL devem ser executadas fora das instruções preparadas.

## Intervalo de exclusão

O Amazon Keyspaces oferece suporte à exclusão de linhas em um intervalo. Um intervalo é um conjunto contíguo de linhas dentro de uma partição. Você especifica um intervalo em uma operação DELETE usando uma cláusula WHERE. Você pode especificar o intervalo para ser uma partição inteira.

Além disso, você pode especificar um intervalo como um subconjunto de linhas contíguas em uma partição usando operadores relacionais (por exemplo, '**>**', '**<**') ou incluindo a chave de partição e omitindo uma ou mais colunas de agrupamento. Com o Amazon Keyspaces, você pode excluir até 1.000 linhas dentro de um intervalo em uma única operação. Além disso, as exclusões de intervalos são atômicas, mas não isoladas.

## Tabelas de sistema

O Amazon Keyspaces preenche as tabelas do sistema que são exigidas pelos drivers open source do Apache 2.0 para o Cassandra. As tabelas do sistema que são visíveis para um cliente contêm informações exclusivas do usuário autenticado. As tabelas do sistema são totalmente controladas pelo Amazon Keyspaces e são somente para leitura.

O acesso somente para leitura às tabelas do sistema é necessário, e você pode controlá-lo com as políticas de acesso do IAM. Para ter mais informações, consulte the section called “Gerenciando acesso usando políticas”. Você deve definir políticas de controle de acesso baseadas em tags para tabelas do sistema de forma diferente, dependendo se você usa o AWS SDK ou as chamadas de API da Cassandra Query Language (CQL) por meio de drivers e ferramentas de desenvolvedor do Cassandra. Para saber mais sobre controle de acesso baseado em tags para tabelas de sistema, consulte the section called “ Acesso a recursos do Amazon Keyspaces com base em tags”.

Se você acessar o Amazon Keyspaces usando endpoints da VPC Amazon, verá entradas na tabela `system.peers` para cada endpoint da VPC Amazon que o Amazon Keyspaces tem permissão para ver. Como resultado, seu driver do Cassandra pode emitir uma mensagem de aviso sobre o próprio nó de controle na tabela `system.peers`. Você pode ignorar esse aviso com segurança.

### Carimbos de data/hora

No Amazon Keyspaces, os timestamps em nível de célula que são compatíveis com os timestamps padrão no Apache Cassandra são um atributo opcional.

A cláusula `USING TIMESTAMP` e a função `WRITETIME` só estão disponíveis quando os timestamps no lado do cliente estão ativados para uma tabela. Para saber mais sobre os timestamps no lado do cliente no Amazon Keyspaces, consulte Carimbos de data/hora do lado do cliente.

## APIs, operações, funções e tipos de dados compatíveis do Cassandra no Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) é compatível com a API Cassandra Query Language (CQL) 3.11 (compatível com versões anteriores da versão 2.x).

O Amazon Keyspaces oferece suporte a todas as operações de plano de dados do Cassandra comumente usadas, como criar espaços de chaves e tabelas, ler dados e gravar dados.

As seções a seguir listam as funcionalidades com suporte.

Tópicos

- Suporte à API do Cassandra
- Suporte à API do ambiente de gerenciamento do Cassandra
- Suporte à API do plano de dados do Cassandra
- Suporte às funções do Cassandra

- Suporte ao tipo de dados do Cassandra

## Suporte à API do Cassandra

| Operação de API | Compatível |
|---|---|
| `CREATE KEYSPACE` | Sim |
| `ALTER KEYSPACE` | Sim |
| `DROP KEYSPACE` | Sim |
| `CREATE TABLE` | Sim |
| `ALTER TABLE` | Sim |
| `DROP TABLE` | Sim |
| `CREATE INDEX` | Não |
| `DROP INDEX` | Não |
| `UNLOGGED BATCH` | Sim |
| `LOGGED BATCH` | Não |
| `SELECT` | Sim |
| `INSERT` | Sim |
| `DELETE` | Sim |
| `UPDATE` | Sim |
| `USE` | Sim |
| `CREATE TYPE` | Não |
| `ALTER TYPE` | Não |
| `DROP TYPE` | Não |

| Operação de API | Compatível |
| --- | --- |
| `CREATE TRIGGER` | Não |
| `DROP TRIGGER` | Não |
| `CREATE FUNCTION` | Não |
| `DROP FUNCTION` | Não |
| `CREATE AGGREGATE` | Não |
| `DROP AGGREGATE` | Não |
| `CREATE MATERIALIZED VIEW` | Não |
| `ALTER MATERIALIZED VIEW` | Não |
| `DROP MATERIALIZED VIEW` | Não |
| `TRUNCATE` | Não |

## Suporte à API do ambiente de gerenciamento do Cassandra

Como o Amazon Keyspaces é gerenciado, as operações de API do ambiente de gerenciamento do Cassandra para gerenciar configurações de clusters e nós não são necessárias. Como resultado, os seguintes atributo do Cassandra não são aplicáveis.

| Atributo | Motivo |
| --- | --- |
| Botão de gravação durável | Todas as gravações são duráveis |
| Leia as configurações de restauração | Não aplicável |
| Segundos de graça do GC | Não aplicável |
| Configurações do filtro Bloom | Não aplicável |
| Configurações de compactação | Não aplicável |

| Atributo | Motivo |
| --- | --- |
| Compression settings (Configurações de compactação) | Não aplicável |
| Configurações de armazenamento em cache | Não aplicável |
| Configurações de segurança | Substituído pelo IAM |

## Suporte à API do plano de dados do Cassandra

| Atributo | Compatível |
| --- | --- |
| Suporte a JSON para instruções SELECT e INSERT | Sim |
| Colunas estáticas | Sim |
| Vida útil (TTL) | Sim |

## Suporte às funções do Cassandra

Para obter mais informações sobre as funções suportadas, consulte o the section called “Funções incorporadas”.

| Função | Compatível |
| --- | --- |
| Funções do `Aggregate` | Não |
| `Blob` conversão | Sim |
| `Cast` | Sim |
| Funções do `Datetime` | Sim |
| Funções de conversão de tempo | Sim |
| Funções do `TimeUuid` | Sim |

| Função | Compatível |
| --- | --- |
| `Token` | Sim |
| `User defined functions (UDF)` | Não |
| `Uuid` | Sim |

## Suporte ao tipo de dados do Cassandra

| Tipo de dados | Compatível | Observação |
| --- | --- | --- |
| `ascii` | Sim | |
| `bigint` | Sim | |
| `blob` | Sim | |
| `boolean` | Sim | |
| `counter` | Sim | |
| `date` | Sim | |
| `decimal` | Sim | |
| `double` | Sim | |
| `float` | Sim | |
| `frozen` | Sim | |
| `inet` | Sim | |
| `int` | Sim | |
| `list` | Sim | |
| `map` | Sim | |

| Tipo de dados | Compatível | Observação |
|---|---|---|
| `set` | Sim | |
| `smallint` | Sim | |
| `text` | Sim | |
| `time` | Sim | |
| `timestamp` | Sim | |
| `timeuuid` | Sim | |
| `tinyint` | Sim | |
| `tuple` | Sim | |
| `user-defined types` (UDT) | Não | Para refatorar UDTs com buffers de protocolo, consulte Buffers de Protocolo do Amazon Keyspaces. |
| `uuid` | Sim | |
| `varchar` | Sim | |
| `varint` | Sim | |

# Níveis de consistência compatíveis do Apache Cassandra no Amazon Keyspaces

Os tópicos desta seção descrevem quais níveis de consistência do Apache Cassandra são suportados para operações de leitura e gravação no Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Níveis de consistência de gravação
- Níveis de consistência de leitura

- Níveis de consistência sem suporte

## Níveis de consistência de gravação

O Amazon Keyspaces replica todas as operações de gravação três vezes em várias zonas de disponibilidade para maior durabilidade e alta disponibilidade. As gravações são armazenadas de forma duradoura antes de serem confirmadas usando o nível de consistência `LOCAL_QUORUM`. Para cada gravação de 1 KB, você recebe a cobrança de 1 unidade de capacidade de gravação (WCU) para tabelas usando o modo de capacidade provisionada ou 1 unidade de solicitação de gravação (WRU) para tabelas usando o modo sob demanda.

Você pode usar `cqlsh` para definir a consistência de todas as consultas na sessão atual para `LOCAL_QUORUM` usando o código a seguir.

```
CONSISTENCY LOCAL_QUORUM;
```

Para configurar o nível de consistência programaticamente, você pode definir a consistência com os drivers de cliente Cassandra apropriados. Por exemplo, os drivers Java da versão 4.x permitem que você defina o nível de consistência no arquivo `app config`, conforme mostrado abaixo.

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

Se você estiver usando um driver Java Cassandra versão 3.x, poderá especificar o nível de consistência da sessão adicionando `.withQueryOptions(new QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)` conforme mostrado no exemplo de código a seguir.

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

Para configurar o nível de consistência para operações de gravação específicas, você pode definir a consistência ao chamar `QueryBuilder.insertInto` com um argumento `setConsistencyLevel` ao usar o driver Java.

## Níveis de consistência de leitura

O Amazon Keyspaces oferece suporte a três níveis de consistência de leitura: `ONE`, `LOCAL_ONE` e `LOCAL_QUORUM`. Durante uma leitura `LOCAL_QUORUM`, o Amazon Keyspaces retorna uma resposta que reflete as atualizações mais recentes de todas as operações anteriores de gravação bem-sucedidas. Usar o nível `ONE` ou `LOCAL_ONE` de consistência pode melhorar o desempenho e a disponibilidade de suas solicitações de leitura, mas a resposta pode não refletir os resultados de uma gravação recém-concluída.

Para cada leitura de 4 KB usando consistência `ONE` ou `LOCAL_ONE`, você recebe uma cobrança de 0,5 unidades de capacidade de leitura (RCUs) para tabelas usando o modo de capacidade provisionada ou 0,5 unidades de solicitação de leitura (RRUs) para tabelas usando o modo sob demanda. Para cada leitura de 4 KB usando consistência `LOCAL_QUORUM`, você recebe a cobrança de 1 unidade de capacidade de leitura (RCU) para tabelas usando o modo de capacidade provisionada ou 1 unidade de solicitação de leitura (RRU) para tabelas usando o modo sob demanda.

Cobrança com base na consistência de leitura e no modo de throughput da capacidade de leitura por tabela para cada 4 KB de leituras

| Níveis de consistência | Provisionada | Sob demanda |
|---|---|---|
| `ONE` | 0,5 RCUs | 0,5 RRUs |
| `LOCAL_ONE` | 0,5 RCUs | 0,5 RRUs |
| `LOCAL_QUORUM` | 1 RCU | 1 RRU |

Para especificar uma consistência diferente para operações de leitura, chame `QueryBuilder.select` com um argumento `setConsistencyLevel` quando estiver usando o driver Java.

# Níveis de consistência sem suporte

Os seguintes níveis de consistência não são suportados pelo Amazon Keyspaces e resultarão em exceções.

Níveis de consistência sem suporte

| Apache Cassandra | Amazon Keyspaces |
|---|---|
| `EACH_QUORUM` | Não suportado |
| `QUORUM` | Não suportado |
| `ALL` | Não suportado |
| `TWO` | Não suportado |
| `THREE` | Não suportado |
| `ANY` | Não suportado |
| `SERIAL` | Não suportado |
| `LOCAL_SERIAL` | Não suportado |

# Como acessar o Amazon Keyspaces (para Apache Cassandra)

Você pode acessar o Amazon Keyspaces usando o console, AWS CloudShell, programaticamente executando um `cqlsh` cliente, o AWS SDK ou usando um driver Cassandra licenciado pelo Apache 2.0. O Amazon Keyspaces tem suporte para os drivers e os clientes que são compatíveis com o Apache Cassandra versão 3.11.2. Antes de acessar o Amazon Keyspaces, você deve concluir a configuração AWS Identity and Access Management e, em seguida, conceder a uma identidade do IAM permissões de acesso ao Amazon Keyspaces.

## Conf AWS Identity and Access Management iguração

### Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

# Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Signing in as the root user (Fazer login como usuário-raiz) no Guia do usuário do Início de Sessão da AWS .
2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .
2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Create a permission set no Guia do usuário do AWS IAM Identity Center .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Add groups no Guia do usuário do AWS IAM Identity Center .

# Configurando o Amazon Keyspaces

O acesso aos recursos do Amazon Keyspaces é gerenciado usando o IAM. Usando o IAM, você pode anexar políticas a usuários, funções e identidades federadas do IAM que concedem permissões de leitura e gravação a recursos específicos no Amazon Keyspaces.

Para começar a conceder permissões a uma identidade do IAM, você pode usar uma das políticas AWS gerenciadas do Amazon Keyspaces:

- AmazonKeyspacesFullAccess— esta política concede permissões para acessar todos os recursos no Amazon Keyspaces com acesso total a todos os recursos.
- AmazonKeyspacesReadOnlyAccess_v2 — essa política concede permissões somente de leitura ao Amazon Keyspaces.

Para obter uma explicação detalhada das ações definidas nas políticas gerenciadas, consultethe section called “Políticas gerenciadas pela AWS”.

Para limitar o escopo das ações que uma identidade do IAM pode realizar ou limitar os recursos que a identidade pode acessar, você pode criar uma política personalizada que usa a política `AmazonKeyspacesFullAccess` gerenciada como modelo e remover todas as permissões desnecessárias. Você também pode limitar o acesso a tabelas ou espaços de teclas específicos. Para obter mais informações sobre como restringir ações ou limitar o acesso a recursos específicos no Amazon Keyspaces, consulte. the section called “Como o Amazon Keyspaces funciona com o IAM”

Para acessar o Amazon Keyspaces depois de criar Conta da AWS e criar uma política que conceda a uma identidade do IAM acesso ao Amazon Keyspaces, continue em uma das seguintes seções:

- Como usar o console
- Usando AWS CloudShell
- Conexão programática


# Como acessar o Amazon Keyspaces usando o console

Você pode acessar o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home. Para obter mais informações sobre AWS Management Console acesso, consulte Como controlar o acesso dos usuários do IAM ao AWS Management Console no Guia do usuário do IAM.

É possível usar o console para fazer o seguinte no Amazon Keyspaces:

- Crie, exclua e gerencie tabelas e espaços de chave.
- Monitore métricas importantes da tabela na guia Monitor de uma tabela:
  - Tamanho da tabela faturável (bytes)
  - Métricas de capacidade
- Execute consultas usando o editor CQL, por exemplo, insira, atualize e exclua dados.
- Altere a configuração do particionador da conta.
- Veja as métricas de desempenho e erro da conta no painel.

Para saber como criar um keyspace e uma tabela do Amazon Keyspaces e configurá-los com exemplos de dados do aplicativo, consulte Conceitos básicos do Amazon Keyspaces (para Apache Cassandra).

# Usando AWS CloudShell para acessar o Amazon Keyspaces

AWS CloudShell é um shell pré-autenticado baseado em navegador que você pode iniciar diretamente do. AWS Management Console Você pode executar AWS CLI comandos em AWS serviços usando seu shell preferido (Bash PowerShell ou Z shell). Para trabalhar com o Amazon Keyspaces usando`cqlsh`, você deve instalar o. `cqlsh-expansion` Para obter instruções de `cqlsh-expansion` instalação, consultethe section called “Usar a `cqlsh-expansion`”.

Você inicia a AWS CloudShell partir do AWS Management Console, e AWS as credenciais que você usou para entrar no console estão automaticamente disponíveis em uma nova sessão de shell.

Essa pré-autenticação de AWS CloudShell usuários permite que você ignore a configuração de credenciais ao interagir com serviços AWS como o Amazon Keyspaces usando `cqlsh` ou a AWS CLI versão 2 (pré-instalada no ambiente computacional do shell).

## Obtendo permissões do IAM para AWS CloudShell

Usando os recursos de gerenciamento de acesso fornecidos por AWS Identity and Access Management, os administradores podem conceder permissões aos usuários do IAM para que eles possam acessar AWS CloudShell e usar os recursos do ambiente.

A maneira mais rápida de um administrador conceder acesso aos usuários é por meio de uma política AWS gerenciada. Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. A seguinte política AWS gerenciada para CloudShell pode ser anexada às identidades do IAM:

- `AWSCloudShellFullAccess`: concede permissão para uso AWS CloudShell com acesso total a todos os recursos.

Se você quiser limitar o escopo das ações que um usuário do IAM pode realizar AWS CloudShell, crie uma política personalizada que use a política `AWSCloudShellFullAccess` gerenciada como modelo. Para obter mais informações sobre como limitar as ações que estão disponíveis para os usuários em CloudShell, consulte Gerenciamento de AWS CloudShell acesso e uso com políticas do IAM no Guia do AWS CloudShell usuário.

> **Note**
>
> Sua identidade do IAM também exige uma política que conceda permissão para fazer chamadas para o Amazon Keyspaces.

Você pode usar uma política AWS gerenciada para dar à sua identidade do IAM acesso ao Amazon Keyspaces ou começar com a política gerenciada como modelo e remover as permissões que você não precisa. Você também pode limitar o acesso a tabelas e espaços de teclas específicos para criar uma política personalizada. A seguinte política gerenciada para Amazon Keyspaces pode ser anexada às identidades do IAM:

- AmazonKeyspacesFullAccess— Essa política concede permissão para usar o Amazon Keyspaces com acesso total a todos os recursos.

Para obter uma explicação detalhada das ações definidas na política gerenciada, consultethe section called “Políticas gerenciadas pela AWS”.

Para obter mais informações sobre como restringir ações ou limitar o acesso a recursos específicos no Amazon Keyspaces, consulte. the section called “Como o Amazon Keyspaces funciona com o IAM”

## Interagindo com o Amazon Keyspaces usando AWS CloudShell

Depois AWS CloudShell de iniciar a partir do AWS Management Console, você pode começar imediatamente a interagir com o Amazon Keyspaces usando nossa interface `cqlsh` de linha de comando. Se você ainda não instalou `ocqlsh-expansion`, consulte the section called “Usar a `cqlsh-expansion`” as etapas detalhadas.

> **Note**
>
> Ao usar o `cqlsh-expansion` in AWS CloudShell, você não precisa configurar as credenciais antes de fazer chamadas, porque você já está autenticado no shell.

Conecte-se ao Amazon Keyspaces e crie um novo keyspace. Em seguida, leia uma tabela do sistema para confirmar se o keyspace foi criado usando AWS CloudShell

1. A partir do AWS Management Console, você pode iniciar CloudShell escolhendo as seguintes opções disponíveis na barra de navegação:
   - Escolha o CloudShell ícone.
   - Comece a digitar “cloudshell” na caixa de pesquisa e escolha a opção. CloudShell
2. Você pode estabelecer uma conexão com o Amazon Keyspaces usando o comando a seguir. Certifique-se de substituir *`cassandra.us-east-1.amazonaws.com`* pelo endpoint correto para sua região.

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

   Se a conexão for bem-sucedida, você verá um resultado semelhante a este:

   ```
   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. Crie um novo espaço de teclas com o nome`mykeyspace`. Você pode usar o comando a seguir para fazer isso.

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

4. Para confirmar que o keyspace foi criado, você pode ler a partir de uma tabela do sistema usando o comando a seguir.

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

Se a chamada tiver êxito, a linha de comando exibirá uma resposta do serviço semelhante à seguinte saída:

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

(1 rows)
```

# Conexão programática ao Amazon Keyspaces

Este tópico descreve as etapas necessárias para se conectar programaticamente ao Amazon Keyspaces. Ele orienta você na criação de credenciais do IAM e lista os endpoints AWS de serviço disponíveis. A última seção mostra como se conectar ao Amazon Keyspaces usando cqlsh. Para step-by-step tutoriais sobre como se conectar ao Amazon Keyspaces usando diferentes drivers do Apache Cassandra, consulte. the section called “Uso de um driver de cliente Cassandra” Para ver um step-by-step tutorial que mostra como se conectar ao Amazon Keyspaces a partir de um endpoint da Amazon VPC, consulte. the section called “Conexão com VPC endpoints”

> **Note**
>
> Para ajudar você a começar, você pode encontrar exemplos de end-to-end código de conexão com o Amazon Keyspaces usando vários drivers de cliente do Cassandra no repositório de exemplos de código do Amazon Keyspaces em. GitHub

O Amazon Keyspaces tem suporte para os drivers e os clientes que são compatíveis com o Apache Cassandra versão 3.11.2. Ele pressupõe que você já tenha concluído as instruções AWS de configuração emComo acessar o Amazon Keyspaces.

Se você já tem um Conta da AWS, consulte os tópicos a seguir para saber como acessar o Amazon Keyspaces usando cqlsh programaticamente:

Tópicos

- Criação de credenciais para acessar o Amazon Keyspaces programaticamente
- Endpoints de serviço para Amazon Keyspaces
- Usar cqlsh para se conectar ao Amazon Keyspaces
- Usar a AWS CLI
- Uso da API
- Usando o Amazon Keyspaces com um SDK AWS
- Como usar um driver de cliente Cassandra para acessar o Amazon Keyspaces programaticamente
- Tutorial: Conectando-se ao Amazon Keyspaces a partir do Amazon Elastic Kubernetes Service

## Criação de credenciais para acessar o Amazon Keyspaces programaticamente

Para fornecer aos usuários e aplicativos credenciais para acesso programático aos recursos do Amazon Keyspaces, você pode executar uma das seguintes ações:

- Crie credenciais específicas do serviço que sejam semelhantes ao nome de usuário e senha tradicionais que o Cassandra usa para autenticação e gerenciamento de acesso. AWS as credenciais específicas do serviço estão associadas a um usuário específico AWS Identity and Access Management (IAM) e só podem ser usadas para o serviço para o qual foram criadas. Para obter mais informações, consulte Como usar o IAM com o Amazon Keyspaces (para Apache Cassandra) no Guia do usuário do IAM.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

- Para aumentar a segurança, recomendamos criar identidades do IAM que sejam usadas em todos os AWS serviços e usar credenciais temporárias. O plug-in de autenticação SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra permite que você autentique chamadas para o Amazon Keyspaces usando chaves de acesso do IAM em vez de nome de usuário e senha. Para saber mais sobre como o plug-in SigV4 do Amazon Keyspaces permite que usuários do IAM, perfis e identidades federadas do IAM se autentiquem nas solicitações da API do Amazon Keyspaces, consulte o processo AWS Signature Version 4 (SigV4).

  É possível fazer download dos plugins SigV4 nos seguintes locais.

  - 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.
  - Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

  Para exemplos de código que mostram como estabelecer conexões usando o plug-in de autenticação SigV4, consulte the section called “Uso de um driver de cliente Cassandra”.

Tópicos

- Gerar credenciais específicas do serviço
- Como criar e configurar AWS credenciais para o Amazon Keyspaces

## Gerar credenciais específicas do serviço

As credenciais específicas de serviço são semelhantes ao nome de usuário e à senha que o Cassandra usa para autenticação e gerenciamento de acesso. As credenciais específicas do serviço permitem que os usuários do IAM acessem um serviço AWS específico. Essas credenciais de longo prazo não podem ser usadas para acessar outros AWS serviços. Eles estão associados a um usuário específico do IAM e não podem ser usados por outros usuários do IAM.

> **Important**
>
> As credenciais específicas do serviço são credenciais de longo prazo associadas a um usuário específico do IAM e só podem ser usadas para o serviço para o qual foram criadas. Para conceder permissões às funções do IAM ou identidades federadas para acessar todos os seus AWS recursos usando credenciais temporárias, você deve usar a AWS autenticação com o plug-in de autenticação SigV4 para Amazon Keyspaces.

Use um dos procedimentos a seguir para gerar credenciais específicas do serviço.

Gerar credenciais específicas do serviço usando o console

Para gerar credenciais específicas do serviço usando o console

1. Faça login no AWS Management Console e abra o AWS Identity and Access Management console emhttps://console.aws.amazon.com/iam/home.
2. No painel de navegação, escolha Usuários e, em seguida, escolha o usuário que você criou anteriormente e que tem permissões do Amazon Keyspaces (política anexada).
3. Selecione Credenciais de segurança. Em Credenciais para Amazon Keyspaces, escolha Gerar credenciais para gerar as credenciais específicas do serviço.

   As credenciais específicas do seu serviço agora estão disponíveis. Esta é a única vez que a senha pode ser visualizada ou baixada. Não será possível recuperá-la posteriormente. No entanto, é possível redefinir a senha a qualquer momento. Salve o usuário e a senha em um local seguro, pois você precisará deles mais tarde.

Gere credenciais específicas do serviço usando o AWS CLI

Para gerar credenciais específicas do serviço usando o AWS CLI

Antes de gerar credenciais específicas do serviço, você precisa baixar, instalar e configurar o AWS Command Line Interface ():AWS CLI

1. Faça o download AWS CLI em http://aws.amazon.com/cli.

> **Note**
>
> AWS CLI É executado em Windows, macOS ou Linux.

2. Siga as instruções para instalar a AWS CLI e configurar a AWS CLI no Guia do usuário.AWS Command Line Interface
3. Usando o AWS CLI, execute o comando a seguir para gerar credenciais específicas do serviço para a usuáriaalice, para que ela possa acessar o Amazon Keyspaces.

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

A saída é semelhante à seguinte.

```
{
    "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"
    }
}
```

Na saída, observe os valores de `ServiceUserName` e `ServicePassword`. Salve os valores em um local seguro, pois você precisará deles mais tarde.

> **Important**
>
> Este é o único momento em que o `ServicePassword` estará disponível para você.

# Como criar e configurar AWS credenciais para o Amazon Keyspaces

Para acessar o Amazon Keyspaces programaticamente com o AWS SDK ou com os AWS CLI drivers do cliente Cassandra e o plug-in SigV4, você precisa de um usuário ou função do IAM com chaves de acesso. Ao usar AWS programaticamente, você fornece suas chaves de AWS acesso para que AWS possa verificar sua identidade em chamadas programáticas. Suas chaves de acesso consistem em um ID de chave de acesso (por exemplo, AKIAIOSFODNN7EXAMPLE) e uma chave de acesso secreta (por exemplo, wjalrxutnfemi/k7mdeng/ CYEXAMPLEKEY). bPxRfi Este tópico o guia pelas etapas necessárias nesse processo.

As melhores práticas de segurança recomendam que você crie usuários do IAM com permissões limitadas e, em vez disso, associe as funções do IAM às permissões necessárias para realizar tarefas específicas. Os usuários do IAM podem então assumir temporariamente as funções do IAM para realizar as tarefas necessárias. Por exemplo, usuários do IAM em sua conta usando o console do Amazon Keyspaces podem mudar para uma função para usar temporariamente as permissões da função no console. Os usuários cedem suas permissões originais e assumem as permissões atribuídas à função. Quando os usuários saem da função, suas permissões originais são restauradas. As credenciais que os usuários usam para assumir a função são temporárias. Pelo contrário, os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança se, em vez de assumir funções, eles tiverem permissões diretamente atribuídas a eles. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários. Para obter mais informações sobre funções, consulte Cenários comuns para funções: usuários, aplicativos e serviços no Guia do usuário do IAM.

Tópicos

- Credenciais exigidas pelo AWS CLIAWS SDK ou pelo plug-in SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra
- Criação de um usuário do IAM para acesso programático ao Amazon Keyspaces em sua conta AWS
- Criar chaves de acesso para um usuário do IAM
- Como gerenciar chaves de acesso para usuários do IAM
- Usar credenciais temporárias para se conectar ao Amazon Keyspaces usando um perfil do IAM e o plug-in SigV4

Credenciais exigidas pelo AWS CLIAWS SDK ou pelo plug-in SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra

As seguintes credenciais são necessárias para autenticar o usuário ou o perfil do IAM:

`AWS_ACCESS_KEY_ID`

Especifica uma chave de AWS acesso associada a um usuário ou função do IAM.

A chave de acesso `aws_access_key_id` é necessária para se conectar programaticamente ao Amazon Keyspaces.

`AWS_SECRET_ACCESS_KEY`

Especifica a chave secreta associada à chave de acesso. Essencialmente, essa é a “senha” para a chave de acesso.

A `aws_secret_access_key` é necessária para se conectar programaticamente ao Amazon Keyspaces.

`AWS_SESSION_TOKEN`: opcional

Especifica o valor de token de sessão que é necessário se você estiver usando credenciais de segurança temporárias recuperadas diretamente das operações do AWS Security Token Service . Para ter mais informações, consulte the section called “Usar credenciais temporárias para se conectar ao Amazon Keyspaces”.

Se você estiver se conectando com um usuário do IAM, o `aws_session_token` não é obrigatório.

Criação de um usuário do IAM para acesso programático ao Amazon Keyspaces em sua conta AWS

Para obter credenciais para acesso programático ao Amazon Keyspaces com o plug-in AWS CLI, AWS SDK ou SigV4, você precisa primeiro criar um usuário ou uma função do IAM. O processo de criar um usuário do IAM e configurar esse usuário do IAM para ter acesso programático ao Amazon Keyspaces é mostrado nas seguintes etapas:

1. Crie o usuário no AWS Management Console Tools for Windows PowerShell ou usando uma operação de AWS API. AWS CLI Se você criar o usuário no AWS Management Console, as credenciais serão criadas automaticamente.
2. Caso crie o usuário programaticamente, você deverá criar uma chave de acesso (uma ID de chave de acesso e uma chave de acesso secreta) para esse usuário em uma etapa adicional.

3. Conceda ao usuário permissões para acessar o Amazon Keyspaces.

Para obter informações sobre as permissões de que você precisa para criar um usuário, consulte Permissões necessárias para acessar recursos do IAM.

Criação de usuários do IAM (console)

Você pode usar o AWS Management Console para criar usuários do IAM.

Para criar um usuário do IAM com acesso programático (console)

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Users (Usuários) e Add users (Adicionar usuários).
3. Digite o nome para o novo usuário. Esse é o nome de login do. AWS

   > **Note**
   >
   > Os nomes de usuário podem ser uma combinação de até 64 letras, dígitos e estes caracteres: adição (+), igual (=), vírgula (,), ponto (.), arroba (@), sublinhado (_) e hífen (-). Os nomes devem ser exclusivos dentro de uma conta. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, você não pode criar dois usuários denominados TESTUSER e testuser.

4. Selecione Chave de acesso - Acesso programático para criar uma chave de acesso para o novo usuário. Você pode visualizar ou fazer download das chaves de acesso ao acessar a página Final.

   Escolha Próximo: permissões.

5. Na página Set permissions (Definir permissões), selecione Attach existing policies to user directly (Anexar políticas existentes diretamente ao usuário).

   Essa opção exibe a lista de políticas AWS gerenciadas e gerenciadas pelo cliente disponíveis em sua conta. Você pode inserir `keyspaces` no campo de pesquisa para exibir apenas políticas relacionadas ao Amazon Keyspaces.

   Para o Amazon Keyspaces, as políticas gerenciadas disponíveis são `AmazonKeyspacesFullAccess` e `AmazonKeyspacesReadOnlyAccess`. Para obter mais informações sobre cada política, consulte the section called “Políticas gerenciadas pela AWS”.

Para fins de teste e para seguir os tutoriais de conexão, selecione a `AmazonKeyspacesReadOnlyAccess` política para o novo usuário do IAM. Observação: como uma prática recomendada, sugerimos que você siga o princípio do privilégio mínimo e crie políticas personalizadas que limitem o acesso a recursos específicos e permitam apenas as ações necessárias. Para obter mais informações sobre políticas do IAM e para ver políticas de exemplo para Amazon Keyspaces, consulte the section called “Políticas baseadas em identidade do Amazon Keyspaces”. Depois de criar políticas de permissão personalizadas, anexe suas políticas às funções e permita que os usuários assumam temporariamente as funções apropriadas.

Escolha Próximo: etiquetas.

6. Na página Adicionar tags (opcional), você pode adicionar tags para o usuário ou escolher Avançar: Revisão.
7. Na página Revisão, você poder ver todas as escolhas feitas até esse ponto. Quando estiver pronto para continuar, escolha Create user (Criar usuário).
8. Para exibir as chaves de acesso do usuário (IDs de chave de acesso e chaves de acesso secretas), selecione Show (Mostrar) ao lado da senha e da chave de acesso. Para salvar as chaves de acesso, escolha Fazer download de .csv e, em seguida, salve o arquivo em um local seguro.

> **Important**
>
> Esta é a única oportunidade de visualizar ou fazer download das chaves de acesso secretas, e você precisa desta informação antes de poder usar o plug-in SigV4. Salve a nova ID da chave de acesso do usuário e a chave de acesso secreta em um local seguro e protegido. Você não terá acesso às chaves secretas novamente depois dessa etapa.

### Criação de usuários do IAM (AWS CLI)

Você pode usar o AWS CLI para criar um usuário do IAM.

Para criar um usuário do IAM com acesso programático (AWS CLI)

1. Crie um usuário com o AWS CLI código a seguir.

- aws iam create-user

2. Forneça ao usuário acesso programático. Isso requer chaves de acesso, que podem ser geradas como se segue.

   - AWS CLI: aws iam create-access-key
   - Ferramentas para Windows PowerShell: New-IAMAccessKey
   - API do IAM: CreateAccessKey

   > **Important**
   >
   > Esta é a única oportunidade de visualizar ou fazer download das chaves de acesso secretas, e você precisa desta informação antes de poder usar o plug-in SigV4. Salve a nova ID da chave de acesso do usuário e a chave de acesso secreta em um local seguro e protegido. Você não terá acesso às chaves secretas novamente depois dessa etapa.

3. Anexe uma política `AmazonKeyspacesReadOnlyAccess` ao usuário que defina as permissões do usuário. Observação: como prática recomendada, recomendamos que você gerencie as permissões de usuário adicionando o usuário a um grupo e anexando uma política ao grupo, em vez de anexá-la diretamente a um usuário.

   - AWS CLI: aws iam attach-user-policy

Criar chaves de acesso para um usuário do IAM

Caso já tenha um usuário do IAM, você poderá criar novas chaves de acesso a qualquer momento. Para obter informações sobre como gerenciar chaves, por exemplo, como alternar as chaves de acesso, consulte Gerenciar chaves de acesso para usuários do IAM.

Para criar chaves de acesso para um usuário do IAM (console)

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Users.
3. Escolha o nome do usuário para o qual você deseja criar uma chave de acesso.
4. Na página Resumo, clique na guia Credenciais de segurança.

5. Na seção Access keys (Chaves de acesso), escolha Create access key (Criar chave de acesso).

   Para ver o novo par de chaves de acesso, escolha Show (Mostrar). Suas credenciais terão a seguinte aparência:

   - ID da chave de acesso: AKIAIOSFODNN7EXAMPLE
   - Chave de acesso secreta: wjalrxutnfemi/k7mdeng/ bPxRfi CYEXAMPLEKEY

   > **Note**
   >
   > Você não terá mais acesso à chave de acesso secreta depois que essa caixa de diálogo for fechada.

6. Para baixar o par de chaves, escolha Baixar arquivo .csv. Armazene as chaves em um lugar seguro.
7. Depois de baixar o arquivo .csv, escolha Close (Fechar).

Quando você cria uma chave de acesso, o par de chaves é ativo por padrão, e você pode usar o par imediatamente.

Como gerenciar chaves de acesso para usuários do IAM

Como uma prática recomendada, sugerimos que você não incorpore chaves de acesso diretamente no código. Os AWS SDKs e as ferramentas de linha de AWS comando permitem que você coloque as chaves de acesso em locais conhecidos para que você não precise mantê-las em código. Coloque chaves de acesso em um dos seguintes locais:

- Variáveis de ambiente: em um sistema multicliente, escolha as variáveis de ambiente do usuário, não as variáveis de ambiente do sistema.
- Arquivo de credenciais da CLI: os arquivos `credentials` e `config` são atualizados quando você executa o comando `aws configure`. O arquivo `credentials` está localizado em `~/.aws/credentials` no Linux, no MacOS ou no Unix, ou ainda em `C:\Users\USERNAME\.aws\credentials` no Windows. Esse arquivo pode conter os detalhes da credencial para o perfil `default` e quaisquer perfis nomeados.
- Arquivo de configuração da CLI: os arquivos `credentials` e `config` são atualizados quando você executa o comando `aws configure`. O arquivo `config` está localizado em `~/.aws/config` no Linux, no MacOS ou no Unix, ou ainda em `C:\Users\USERNAME\.aws\config` no

Windows. Esse arquivo contém as definições de configuração para o perfil padrão e quaisquer perfis nomeados.

Armazenar chaves de acesso como variáveis de ambiente é um pré-requisito para o the section called “Plugin de autenticação para Java 4.x”. O cliente pesquisa credenciais usando a cadeia de fornecedores de credenciais padrão, e as chaves de acesso armazenadas como variáveis de ambiente têm precedência sobre todos os outros locais, por exemplo, arquivos de configuração. Para obter mais informações, consulte Definições e precedência de configuração.

Os exemplos a seguir mostram como configurar variáveis de ambiente para o usuário padrão.

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>
```

Configurar a variável de ambiente altera o valor usado até o final da sua sessão de shell ou até que você defina a variável como um valor diferente. Você pode tornar as variáveis persistentes em sessões futuras definindo-as no script de inicialização do 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>
```

O uso de `set` para definir uma variável de ambiente altera o valor usado até o final da sessão de prompt de comando atual ou até que você defina a variável como um valor diferente. O uso de `setx` para definir uma variável de ambiente altera o valor usado na sessão de prompt de comando atual e todas as sessões de prompt de comando que você criar após a execução do comando. Não afeta outros shells de comando que já estejam em execução no momento em que você executar o comando.

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>"
```

Se você definir uma variável de ambiente no PowerShell prompt, conforme mostrado nos exemplos anteriores, ela salvará o valor somente durante a sessão atual. Para tornar a configuração da variável de ambiente persistente em todas as sessões PowerShell e nas sessões do Prompt de Comando, armazene-a usando o aplicativo Sistema no Painel de Controle. Como alternativa, você pode definir a variável para todas as PowerShell sessões futuras adicionando-a ao seu PowerShell perfil. Consulte a PowerShell documentação para obter mais informações sobre como armazenar variáveis de ambiente ou persisti-las nas sessões.

### Usar credenciais temporárias para se conectar ao Amazon Keyspaces usando um perfil do IAM e o plug-in SigV4

Para aumentar a segurança, você pode usar credenciais temporárias para se autenticar com o plug-in SigV4. Em muitos casos, você não precisa de chaves de acesso de longo prazo que nunca expiram (como há no caso de um usuário do IAM). Em vez disso, você pode criar perfis do IAM e gerar credenciais de segurança temporárias. As credenciais de segurança temporárias consistem em um ID da chave de acesso e uma chave de acesso secreta, mas elas também incluem um token de segurança que indica quando as credenciais expiram. Para saber mais sobre como usar funções do IAM em vez de chaves de acesso de longo prazo, consulte Como mudar para uma função do IAM (AWS API).

Para começar com credenciais temporárias, primeiro você precisa criar um perfil do IAM.

#### Crie um perfil do IAM que conceda acesso somente leitura ao Amazon Keyspaces

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Perfis e depois Criar perfil.
3. Na página Criar perfil, em Selecionar tipo de entidade confiável, selecione AWS serviço. Em Escolher um caso de uso, escolha Amazon EC2 e escolha Próximo.
4. Na página Adicionar permissões, em Políticas de permissões, escolha Amazon Keyspaces Read Only Access na lista de políticas e escolha Próximo.
5. Na página Nome, revisão e criação, insira um nome para o perfil e revise as seções Selecionar entidades confiáveis e Adicionar permissões. Você também pode adicionar tags opcionais para o perfil nesta página. Quando concluir, selecione Criar perfil. Lembre-se desse nome, pois você precisará dele quando executar sua instância do Amazon EC2.

Para usar credenciais de segurança temporárias no código, você chama programaticamente uma AWS Security Token Service API como `AssumeRole` e extrai as credenciais e o token de sessão resultantes da sua função do IAM que você criou na etapa anterior. Em seguida, você usa esses valores como credenciais para chamadas subsequentes para AWS. O exemplo a seguir mostra o pseudocódigo de como usar credenciais de segurança temporárias:

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

Para obter um exemplo que implementa credenciais temporárias usando o driver Python para acessar o Amazon Keyspaces, consulte ???.

Para obter detalhes sobre como chamar `AssumeRole`, `GetFederationToken` e outras operações de API, consulte a Referência da API do AWS Security Token Service. Para obter informações sobre como obter as credenciais de segurança temporárias e o token de sessão provenientes do resultado, consulte a documentação do SDK com o qual você está trabalhando. Você pode encontrar a documentação de todos os AWS SDKs na página principal da AWS documentação, na seção SDKs e kits de ferramentas.

## Endpoints de serviço para Amazon Keyspaces

Tópicos

- Portas e protocolos
- Endpoints globais
- Endpoints do FIPS na AWS GovCloud (US) Region
- Endpoints da região da China

### Portas e protocolos

Você pode acessar o Amazon Keyspaces programaticamente executando um cliente `cqlsh`, usando um driver Cassandra licenciado para Apache 2.0 ou usando o AWS CLI e o SDK AWS.

A tabela a seguir mostra as portas e os protocolos dos diferentes mecanismos de acesso.

| Acesso programático | Porta | Protocolo |
|---|---|---|
| CQLSH | 9142 | TLS |
| Driver Cassandra | 9142 | TLS |
| AWS CLI | 443 | HTTPS |
| AWS SDK | 443 | HTTPS |

Para conexões TLS, o Amazon Keyspaces usa o Starfield CA para se autenticar no servidor. Para obter mais informações, consulte the section called “Como configurar manualmente as conexões `cqlsh` para o TLS” ou a seção Antes de começar do seu driver no capítulo the section called “Uso de um driver de cliente Cassandra”.

## Endpoints globais

O Amazon Keyspaces está disponível nas seguintes Regiões da AWS. Esta tabela mostra o endpoint de serviço disponível para cada região.

| Nome da região | Região | Endpoint | Protocolo | |
|---|---|---|---|---|
| Leste dos EUA (Ohio) | us-east-2 | cassandra.us-east-2.amazonaws.com | HTTPS e TLS | |
| Leste dos EUA (Norte da Virgínia) | us-east-1 | cassandra.us-east-1.amazonaws.com<br><br>cassandra-fips.us-east-1.amazonaws.com | HTTPS e TLS<br><br>TLS | |
| Oeste dos EUA (N. da Califórnia) | us-west-1 | cassandra.us-west-1.amazonaws.com | HTTPS e TLS | |

| Nome da região | Região | Endpoint | Protocolo |
|---|---|---|---|
| Oeste dos EUA (Oregon) | us-west-2 | cassandra.us-west-2.amazonaws.com<br><br>cassandra-fips.us-west-2.amazonaws.com | HTTPS e TLS<br><br>TLS |
| Ásia-Pacífico (Hong Kong) | ap-east-1 | cassandra.ap-east-1.amazonaws.com | HTTPS e TLS |
| Ásia-Pacífico (Mumbai) | ap-south-1 | cassandra.ap-south-1.amazonaws.com | HTTPS e TLS |
| Ásia-Pacífico (Seul) | ap-northeast-2 | cassandra.ap-northeast-2.amazonaws.com | HTTPS e TLS |
| Ásia-Pacífico (Singapura) | ap-southeast-1 | cassandra.ap-southeast-1.amazonaws.com | HTTPS e TLS |
| Ásia-Pacífico (Sydney) | ap-southeast-2 | cassandra.ap-southeast-2.amazonaws.com | HTTPS e TLS |
| Ásia-Pacífico (Tóquio) | ap-northeast-1 | cassandra.ap-northeast-1.amazonaws.com | HTTPS e TLS |
| Canadá (Central) | ca-central-1 | cassandra.ca-central-1.amazonaws.com | HTTPS e TLS |

| Nome da região | Região | Endpoint | Protocolo |
|---|---|---|---|
| Europa (Frankfurt) | eu-central-1 | cassandra.eu-central-1.amazonaws.com | HTTPS e TLS |
| Europa (Irlanda) | eu-west-1 | cassandra.eu-west-1.amazonaws.com | HTTPS e TLS |
| Europa (Londres) | eu-west-2 | cassandra.eu-west-2.amazonaws.com | HTTPS e TLS |
| Europa (Paris) | eu-west-3 | cassandra.eu-west-3.amazonaws.com | HTTPS e TLS |
| Europa (Estocolmo) | eu-north-1 | cassandra.eu-north-1.amazonaws.com | HTTPS e TLS |
| Oriente Médio (Barém) | me-south-1 | cassandra.me-south-1.amazonaws.com | HTTPS e TLS |
| América do Sul (São Paulo) | sa-east-1 | cassandra.sa-east-1.amazonaws.com | HTTPS e TLS |
| AWS GovCloud (Leste dos EUA) | us-gov-east-1 | cassandra.us-gov-east-1.amazonaws.com | HTTPS e TLS |
| AWS GovCloud (Oeste dos EUA) | us-gov-west-1 | cassandra.us-gov-west-1.amazonaws.com | HTTPS e TLS |

## Endpoints do FIPS na AWS GovCloud (US) Region

Endpoints FIPS disponíveis no AWS GovCloud (US) Region. Para obter mais informações, consulte Amazon Keyspaces no AWS GovCloud (US) Guia do Usuário

| Nome da região | região | Endpoint de FIPS | Protocolo |
|---|---|---|---|
| AWS GovCloud (Leste dos EUA) | us-gov-east-1 | cassandra.us-gov-east-1.amazonaws.com | HTTPS e TLS |
| AWS GovCloud (Oeste dos EUA) | us-gov-west-1 | cassandra.us-gov-west-1.amazonaws.com | HTTPS e TLS |

## Endpoints da região da China

Os seguintes endpoints do Amazon Keyspaces estão disponíveis nas regiões AWS da China.

Para acessar esses endpoints, você precisa se inscrever em um conjunto separado de credenciais de conta exclusivas para as regiões da China. Para obter mais informações, consulte Inscrição, contas e credenciais na China.

| Nome da região | Região | Endpoint | Protocolo |
|---|---|---|---|
| China (Pequim) | cn-north-1 | cassandra.cn-north-1.amazonaws.com.cn | HTTPS e TLS |
| China (Ningxia) | cn-northwest-1 | cassandra.cn-northwest-1.amazonaws.com.cn | HTTPS e TLS |

# Usar **cqlsh** para se conectar ao Amazon Keyspaces

Para se conectar ao Amazon Keyspaces usando `cqlsh`, você pode usar o `cqlsh-expansion`. Este é um kit de ferramentas que contém ferramentas comuns do Apache Cassandra, como `cqlsh`, e auxiliares pré-configurados para o Amazon Keyspaces, mantendo total compatibilidade com o Apache Cassandra. O `cqlsh-expansion` integra o plug-in de autenticação SigV4 e permite que você se conecte usando chaves de acesso do IAM em vez de nome de usuário e senha. Você só precisa instalar os scripts `cqlsh` para fazer uma conexão e não a distribuição completa do Apache Cassandra, porque o Amazon Keyspaces é uma tecnologia sem servidor. Esse pacote de instalação leve inclui os `cqlsh-expansion` e os scripts clássicos `cqlsh` que você pode instalar em qualquer plataforma compatível com Python.

Para obter informações gerais sobre o `cqlsh`, consulte cqlsh: shell do CQL.

Tópicos


- Usar a cqlsh-expansion para se conectar ao Amazon Keyspaces
- Como configurar manualmente as conexões cqlsh para o TLS


## Usar a **cqlsh-expansion** para se conectar ao Amazon Keyspaces

Instalar e configurar a **cqlsh-expansion**

1. Para instalar o pacote `cqlsh-expansion` Python, você pode executar um comando `pip`. Isso instala os scripts `cqlsh-expansion` em sua máquina usando uma instalação pip e um arquivo contendo uma lista de dependências. O `--user flag` instrui `pip` a usar o diretório de instalação do usuário do Python para sua plataforma. Em um sistema baseado em Unix, esse deve ser o diretório `~/.local/`.

   Você precisa do Python 3 para instalar o `cqlsh-expansion`. Para descobrir sua versão do Python, use `Python --version`. Para instalar, você pode executar o seguinte comando.

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

   A saída deve ser semelhante a esta.

   ```
   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
```

Se o diretório de instalação não estiver noPATH, você precisará adicioná-lo seguindo as instruções do seu sistema operacional. Abaixo está um exemplo para o Ubuntu Linux.

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

Para confirmar se o pacote está instalado, você pode executar o comando a seguir.

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

A saída deve ser algo parecido com isso.

```
cqlsh 6.1.0
```

2. Para configurar o `cqlsh-expansion`, você pode executar um script de pós-instalação para concluir automaticamente as seguintes etapas:

   1. Crie o diretório `.cassandra` no diretório inicial do usuário se ele ainda não existir.
   2. Copie um arquivo `cqlshrc` de configuração pré-configurado no diretório `.cassandra`.
   3. Copie o certificado digital Starfield no diretório `.cassandra`. O Amazon Keyspaces usa esse certificado para configurar a conexão segura com o Transport Layer Security (TLS). A criptografia em trânsito fornece uma camada adicional de proteção de dados ao criptografar seus dados à medida que eles viajam de e para o Amazon Keyspaces.

   Para revisar o script primeiro, você pode acessá-lo no repositório do Github em post_install.py.

   Para usar o script, você pode executar o seguinte comando.

   ```
   cqlsh-expansion.init
   ```

   > **Note**
   >
   > O diretório e o arquivo criados pelo script de pós-instalação não são removidos quando você desinstala o `cqlsh-expansion` usando `pip uninstall`, e precisam ser excluídos manualmente.

Conectar-se ao Amazon Keyspaces usando **`cqlsh-expansion`**

1. Configure seu Região da AWS e adicione-o como uma variável de ambiente do usuário.

   Para adicionar sua região padrão como uma variável de ambiente em um sistema baseado em Unix, você pode executar o comando a seguir. Para este exemplo, usamos Leste dos EUA (Norte da Virgínia).

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

   Para obter mais informações sobre como definir variáveis de ambiente, inclusive para outras plataformas, consulte Como definir variáveis de ambiente.

2. Encontre seu endpoint.

   Escolha o endpoint de serviço apropriado para sua região. Para analisar os endpoints disponíveis para o Amazon Keyspaces, consulte the section called “Service endpoints (Endpoints de serviço)”. Para este exemplo, usamos o endpoint `cassandra.us-east-1.amazonaws.com`.

3. Configurar o modo de autenticação.

   A conexão com as chaves de acesso do IAM (usuários, perfis e identidades federadas do IAM) é o método recomendado para aumentar a segurança.

   Antes de se conectar com as chaves de acesso do IAM, você precisa concluir as seguintes etapas:

   a. Crie um usuário do IAM ou siga as melhores práticas e crie um perfil do IAM que os usuários do IAM possam assumir. Para obter mais informações sobre como criar uma chave de acesso do IAM, consulte the section called “Credenciais do IAM para autenticação AWS”.

   b. Crie uma política do IAM que conceda ao perfil (ou ao usuário do IAM) no mínimo o acesso somente leitura ao Amazon Keyspaces. Para obter mais informações sobre as permissões necessárias para que o usuário ou o perfil do IAM se conecte ao Amazon Keyspaces, consulte the section called “Como acessar as tabelas do Amazon Keyspaces”.

   c. Adicione as chaves de acesso do usuário do IAM às variáveis de ambiente do usuário, conforme mostrado no exemplo a seguir.

      ```
      export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
      ```

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

Para obter mais informações sobre como definir variáveis de ambiente, inclusive para outras plataformas, consulte Como definir variáveis de ambiente.

> **Note**
>
> Se você estiver se conectando a partir de uma instância do Amazon EC2, também precisará configurar uma regra de saída no grupo de segurança que permita o tráfego da instância para o Amazon Keyspaces. Para obter mais informações sobre como visualizar e editar regras de saída do EC2, consulte Adicionar regras a um grupo de segurança no Guia do usuário do Amazon EC2.

4. Conectar-se ao Amazon Keyspaces usando a `cqlsh-expansion` e a autenticação SigV4.

Para se conectar ao Amazon Keyspaces com a `cqlsh-expansion`, você pode usar o comando a seguir. Certifique-se de substituir o endpoint de serviço pelo endpoint correto para sua região.

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

Se a conexão for bem-sucedida, você verá um resultado semelhante a este:

```
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>
```

Se você encontrar um erro de conexão, consulte the section called “Erros de conexão cqlsh” para obter informações sobre solução de problemas.

- Conectar-se ao Amazon Keyspaces com credenciais específicas do serviço.

  Para se conectar à combinação tradicional de nome de usuário e senha que o Cassandra usa para autenticação, você deve primeiro criar credenciais específicas do serviço para o Amazon Keyspaces, conforme descrito em the section called “Credenciais específicas do serviço”. Você também precisa conceder a esse usuário permissões para acessar o Amazon Keyspaces. Para obter mais informações, consulte the section called “Como acessar as tabelas do Amazon Keyspaces”.

Depois de criar credenciais e permissões específicas do serviço para o usuário, você deve atualizar o arquivo `cqlshrc`, normalmente encontrado no caminho do diretório do usuário `~/.cassandra/`. No arquivo `cqlshrc`, vá para a seção `[authentication]` do Cassandra e comente o módulo e a classe SigV4 em `[auth_provider]` usando o caractere ";", conforme mostrado no exemplo a seguir.

```
[auth_provider]

; module = cassandra_sigv4.auth

; classname = SigV4AuthProvider
```

Depois de atualizar o arquivo `cqlshrc`, você pode se conectar ao Amazon Keyspaces com credenciais específicas do serviço usando o comando a seguir.

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

Limpeza

- Para remover o pacote `cqlsh-expansion`, é possível usar o comando `pip uninstall`.

  ```
  pip3 uninstall cqlsh-expansion
  ```

  O comando `pip3 uninstall` não remove o diretório e os arquivos relacionados criados pelo script de pós-instalação. Para remover a pasta e os arquivos criados pelo script de pós-instalação, você pode excluir o diretório `.cassandra`.

## Como configurar manualmente as conexões **`cqlsh`** para o TLS

O Amazon Keyspaces só aceita conexões seguras usando Transport Layer Security (TLS). Você pode usar o utilitário `cqlsh-expansion` que baixa automaticamente o certificado para você e instala um arquivo `cqlshrc` de configuração pré-configurado. Para obter mais informações sobre isso, consulte the section called "Usar a `cqlsh-expansion`" nesta página.

Se quiser baixar o certificado e configurar a conexão manualmente, você pode fazer isso usando as etapas a seguir.

1. Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.

2. Abra o arquivo `cqlshrc` de configuração no diretório inicial do Cassandra, por exemplo, `${HOME}/.cassandra/cqlshrc`, e adicione as linhas a seguir.

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

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

## Usar a AWS CLI

Você pode usar a AWS Command Line Interface (AWS CLI) para controlar vários serviços da AWS via linha de comando e automatizá-los usando scripts. Com o Amazon Keyspaces, você pode usar o AWS CLI para operações de definição de linguagem de dados (DDL), como criar um espaço de chave ou uma tabela. Além disso, você pode usar serviços e ferramentas de infraestrutura como código (IaC), como AWS CloudFormation e o Terraform.

Antes de usar a AWS CLI com o Amazon Keyspaces, você deve obter um ID de chave de acesso e uma chave de acesso secreta. Para obter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Para obter uma listagem completa de todos os comandos disponíveis para o Amazon Keyspaces na AWS CLI, consulte a Referência de comandos da AWS CLI.

Tópicos


- Download e configuração da AWS CLI
- Uso do AWS CLI com o Amazon Keyspaces


## Download e configuração da AWS CLI

A AWS CLI está disponível em https://aws.amazon.com/cli. Ela é executada no Windows, macOS ou Linux. Depois de fazer download da AWS CLI, siga estas etapas para instalá-la e configurá-la:

1. Acesse o Guia do usuário do AWS Command Line Interface.
2. Siga as instruções de Instalação da AWS CLI e Configuração da AWS CLI

## Uso do AWS CLI com o Amazon Keyspaces

O formato de linha de comando consiste em um nome de operação do Amazon Keyspaces, seguido pelos parâmetros dessa operação. O AWS CLI suporta uma sintaxe abreviada para os valores de parâmetro, assim como JSON. Os seguintes exemplos do Amazon Keyspaces usam sintaxe AWS CLI abreviada. Para obter mais informações, consulte Uso da sintaxe abreviada com a CLI da AWS.

O comando a seguir cria um espaço de teclas com o catálogo de nomes.

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

O comando retorna o recurso nome do recurso da Amazon (ARN) na saída.

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

Para confirmar que o catálogo do espaço de chave existe, você pode usar o seguinte comando.

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

A saída do comando retorna os seguintes valores.

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

```
}
```

O comando a seguir cria uma tabela com o nome book_awards. A chave de partição da tabela consiste nas colunas `year` e `award` e a chave de agrupamento consiste nas colunas `category` de `rank`. As duas colunas de agrupamento usam a ordem de classificação crescente. (Para facilitar a leitura, comandos longos nesta seção são divididos em linhas separadas.)

```
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}]'
```

Este comando resulta na seguinte saída.

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

Para confirmar os metadados e as propriedades da tabela, você pode usar o seguinte comando:

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

Este comando retorna a seguinte saída.

```
{
    "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": ""
    }
}
```

Ao criar tabelas com esquemas complexos, pode ser útil carregar a definição do esquema da tabela a partir de um arquivo JSON. A seguir, temos um exemplo. Faça o download do arquivo JSON de exemplo de definição de esquema em schema_definition.zip e extraia `schema_definition.json`, anotando o caminho para o arquivo. Neste exemplo, o arquivo JSON de definição do esquema está localizado no diretório atual. Para diferentes opções de caminho de arquivo, consulte Como carregar parâmetros de um arquivo.

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

Os exemplos a seguir mostram como criar uma tabela simples com o nome myTable com opções adicionais. Observe que os comandos estão divididos em linhas separadas para melhorar a legibilidade. Esse comando mostra como criar uma tabela e:

- definir o modo de capacidade da tabela
- ativar a recuperação pontual para a tabela
- definir a vida útil (TTL) padrão para a tabela como um ano
- adicionar duas tags para a tabela

```
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'
```

Este exemplo mostra como criar uma nova tabela que usa uma chave gerenciada pelo cliente para criptografia e tem o TTL ativado para permitir que você defina datas de expiração para colunas e linhas. Para executar essa amostra, você deve substituir o ARN do recurso da chave AWS KMS gerenciada pelo cliente pela sua própria chave e garantir que o Amazon Keyspaces tenha acesso a ela.

```
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'
```

## Uso da API

Você pode usar o AWS SDK e a AWS Command Line Interface (AWS CLI) para trabalhar de forma interativa com o Amazon Keyspaces. Você pode usar a API para operações de definição de linguagem de dados (DDL), como criar um espaço de chave ou uma tabela. Além disso, você pode usar serviços e ferramentas de infraestrutura como código (IaC), como AWS CloudFormation e o Terraform.

Antes de usar a AWS CLI com o Amazon Keyspaces, você deve obter um ID de chave de acesso e uma chave de acesso secreta. Para obter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Para obter uma lista completa de todas as operações disponíveis para Amazon Keyspaces na API, consulte Amazon Keyspaces API Reference.

# Usando o Amazon Keyspaces com um SDK AWS

AWS kits de desenvolvimento de software (SDKs) estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que facilitam a criação de aplicações em seu idioma preferido pelos desenvolvedores.

| Documentação do SDK | Exemplos de código |
| --- | --- |
| AWS SDK for C++ | AWS SDK for C++ exemplos de código |
| AWS CLI | AWS CLI exemplos de código |
| AWS SDK for Go | AWS SDK for Go exemplos de código |
| AWS SDK for Java | AWS SDK for Java exemplos de código |
| AWS SDK for JavaScript | AWS SDK for JavaScript exemplos de código |
| AWS SDK para Kotlin | AWS SDK para Kotlin exemplos de código |
| AWS SDK for .NET | AWS SDK for .NET exemplos de código |
| AWS SDK for PHP | AWS SDK for PHP exemplos de código |
| AWS Tools for PowerShell | Ferramentas para exemplos PowerShell de código |
| AWS SDK for Python (Boto3) | AWS SDK for Python (Boto3) exemplos de código |
| AWS SDK for Ruby | AWS SDK for Ruby exemplos de código |
| AWS SDK para Rust | AWS SDK para Rust exemplos de código |
| SDK da AWS para SAP ABAP | SDK da AWS para SAP ABAP exemplos de código |
| AWS SDK for Swift | AWS SDK for Swift exemplos de código |

Exemplo de disponibilidade

Você não consegue encontrar o que precisa? Solicite um código de exemplo no link Fornecer feedback na parte inferior desta página.

## Como usar um driver de cliente Cassandra para acessar o Amazon Keyspaces programaticamente

Você pode usar vários drivers Cassandra de código aberto de terceiros para se conectar ao Amazon Keyspaces. O Amazon Keyspaces é compatível com os drivers do Cassandra que suportam o Apache Cassandra versão 3.11.2. Estes são os drivers e as versões mais recentes que testamos e recomendamos usar com o 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`

Para obter mais informações sobre os drivers do Cassandra, consulte Drivers do Apache Cassandra Client.

Note

Para ajudar você a começar, você pode visualizar e baixar exemplos de end-to-end códigos que estabelecem conexões com o Amazon Keyspaces com drivers populares. Veja exemplos do Amazon Keyspaces em. GitHub

Os tutoriais deste capítulo incluem uma consulta CQL simples para confirmar que a conexão com o Amazon Keyspaces foi estabelecida com sucesso. Para saber como trabalhar com espaços de chaves e tabelas depois de se conectar a um endpoint do Amazon Keyspaces, consulte Referência da linguagem CQL. Para ver um step-by-step tutorial que mostra como se conectar ao Amazon

Keyspaces a partir de um endpoint da Amazon VPC, consulte. the section called “Conexão com VPC endpoints”

Tópicos


- Como usar um driver de cliente Java Cassandra para acessar o Amazon Keyspaces programaticamente
- Como usar um driver de cliente Cassandra Python para acessar o Amazon Keyspaces programaticamente
- Como usar um driver de cliente Cassandra Node.js para acessar o Amazon Keyspaces programaticamente
- Como usar um driver de cliente Cassandra .NET Core para acessar o Amazon Keyspaces programaticamente
- Como usar um driver de cliente Cassandra Go para acessar o Amazon Keyspaces programaticamente
- Como usar um driver de cliente Cassandra Perl para acessar o Amazon Keyspaces programaticamente


## Como usar um driver de cliente Java Cassandra para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente Java.

> **Note**
>
> Atualmente, o Java 17 e o DataStax Java Driver 4.17 estão apenas em suporte Beta. Para ter mais informações, consulte https://docs.datastax.com/en/developer/java-driver/4.17/upgrade_guide/.

Para fornecer aos usuários e aplicativos credenciais para acesso programático aos recursos do Amazon Keyspaces, você pode executar uma das seguintes ações:

- Criar credenciais específicas do serviço associadas a um usuário específico AWS Identity and Access Management (IAM).
- Para aumentar a segurança, recomendamos criar chaves de acesso do IAM para identidades do IAM que são usadas em todos os AWS serviços. O plug-in de autenticação SigV4 do Amazon

Keyspaces para drivers de clientes do Cassandra permite que você autentique chamadas para o Amazon Keyspaces usando chaves de acesso do IAM em vez de nome de usuário e senha. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

> **Note**
>
> Para obter um exemplo de como usar o Amazon Keyspaces com o Spring Boot, consulte https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring.

Tópicos

- Antes de começar
- Um tep-by-step tutorial para se conectar ao Amazon Keyspaces usando o driver DataStax Java para Apache Cassandra usando credenciais específicas do serviço
- Um tep-by-step tutorial para se conectar ao Amazon Keyspaces usando o driver DataStax Java 4.x para Apache Cassandra e o plug-in de autenticação SigV4
- Conecte-se ao Amazon Keyspaces usando o driver DataStax Java 3.x para Apache Cassandra e o plug-in de autenticação SigV4

Antes de começar

Para se conectar ao Amazon Keyspaces, você precisa concluir as seguintes tarefas antes de começar.

1. O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes.

   a. Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.

b. Converta o certificado digital Starfield em um arquivo 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
```

Nesta etapa, você precisa criar uma senha para o repositório de chaves e confiar nesse certificado. O comando interativo tem a aparência a seguir.

```
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. Anexe o arquivo trustStore nos argumentos da JVM:

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

Um tep-by-step tutorial para se conectar ao Amazon Keyspaces usando o driver DataStax Java para Apache Cassandra usando credenciais específicas do serviço

O step-by-step tutorial a seguir mostra como se conectar ao Amazon Keyspaces usando um driver Java para Cassandra usando credenciais específicas do serviço. Especificamente, você usará a versão 4.0 do driver DataStax Java para o Apache Cassandra.

Tópicos

- Etapa 1: pré-requisitos
- Etapa 2: configurar o driver
- Etapa 3: executar o aplicativo de exemplo

Etapa 1: pré-requisitos

Para seguir este tutorial, você precisa gerar credenciais específicas do serviço e adicionar o driver DataStax Java para Apache Cassandra ao seu projeto Java.

- Gere credenciais específicas do serviço para seu usuário do IAM do Amazon Keyspaces concluindo as etapas em the section called “Credenciais específicas do serviço”. Se preferir usar as chaves de acesso do IAM para autenticação, consulte the section called “Plugin de autenticação para Java 4.x”.
- Adicione o driver DataStax Java para Apache Cassandra ao seu projeto Java. Verifique se está usando uma versão do driver compatível com o Apache Cassandra 3.11.2. Para obter mais informações, consulte a documentação do driver DataStax Java para Apache Cassandra.

Etapa 2: configurar o driver

Você pode especificar as configurações do driver DataStax Java Cassandra criando um arquivo de configuração para seu aplicativo. Esse arquivo de configuração substitui as configurações padrão e instrui o driver a se conectar ao endpoint do serviço Amazon Keyspaces usando a porta 9142. Para obter uma lista de endpoints de serviço, consulte the section called “Service endpoints (Endpoints de serviço)”.

Crie um arquivo de configuração e salve-o na pasta de recursos do aplicativo — por exemplo, `src/main/resources/application.conf`. Abra `application.conf` e adicione as seguintes configurações.

1. Provedor de autenticação: crie o provedor de autenticação com a `PlainTextAuthProvider` classe. ServiceUserO *`nome`* e *`ServicePassword`*deve corresponder ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas em. Gerar credenciais específicas do serviço

   > **Note**
   >
   > Você pode usar credenciais de curto prazo usando o plug-in de autenticação do driver DataStax Java para Apache Cassandra em vez de credenciais codificadas no arquivo de configuração do driver. Para saber mais, siga as instruções do the section called “Plugin de autenticação para Java 4.x”.

2. Datacenter local: defina o valor `local-datacenter` para a região à qual você está se conectando. Por exemplo, se o aplicativo estiver se conectando a `cassandra.us-east-2.amazonaws.com`, defina o datacenter local como `us-east-2`. Para saber todas as Regiões da AWS disponíveis, consulte ???. Defina `slow-replica-avoidance = false` para balancear a carga em relação a menos nós.

3. SSL/TLS — Inicialize o SSL EngineFactory adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com. `class = DefaultSslEngineFactory` Forneça o caminho para o arquivo trustStore e a senha que você criou anteriormente. O Amazon Keyspaces não oferece suporte `hostname-validation` a pares, então defina essa opção como 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**
>
> Em vez de adicionar o caminho para o trustStore no arquivo de configuração, você também pode adicioná-lo diretamente no código do aplicativo ou aos seus argumentos da JVM.

### Etapa 3: executar o aplicativo de exemplo

Este exemplo de código mostra um aplicativo de linha de comando simples que cria um pool de conexões com o Amazon Keyspaces usando o arquivo de configuração que criamos anteriormente. Ele confirma que a conexão foi estabelecida executando uma consulta simples.

```
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**
>
> Use um `try` bloco para estabelecer a conexão e garantir que ela esteja sempre fechada. Se você não usa um `try` bloco, lembre-se de fechar sua conexão para evitar o vazamento de recursos.

Um tep-by-step tutorial para se conectar ao Amazon Keyspaces usando o driver DataStax Java 4.x para Apache Cassandra e o plug-in de autenticação SigV4

A seção a seguir descreve como usar o plug-in de autenticação SigV4 para o driver DataStax Java 4.x de código aberto do Apache Cassandra para acessar o Amazon Keyspaces (para o Apache Cassandra). O plug-in está disponível no GitHubrepositório.

O plug-in de autenticação SigV4 permite que você use credenciais do IAM para usuários ou perfis ao se conectar ao Amazon Keyspaces. Em vez de exigir um nome de usuário e senha, esse plug-

in assina solicitações de API usando chaves de acesso. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Etapa 1: pré-requisitos

Para seguir este tutorial, você precisa concluir as tarefas a seguir.

- Se ainda não tiver feito isso, crie credenciais para o usuário do IAM ou o perfil do IAM seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”. Este tutorial pressupõe que as chaves de acesso sejam armazenadas como variáveis de ambiente. Para ter mais informações, consulte the section called “Como gerenciar chaves de acesso”.
- Adicione o driver DataStax Java para Apache Cassandra ao seu projeto Java. Verifique se está usando uma versão do driver compatível com o Apache Cassandra 3.11.2. Para obter mais informações, consulte a documentação do driver DataStax Java para Apache Cassandra.
- Adicione o plug-in de autenticação ao seu aplicativo. O plug-in de autenticação suporta a versão 4.x do driver DataStax Java para Apache Cassandra. Se estiver usando o Apache Maven ou um sistema de compilação que pode usar dependências do Maven, adicione as seguintes dependências ao seu arquivo `pom.xml`.

  > **Important**
  >
  > Substitua a versão do plug-in pela versão mais recente, conforme mostrado no GitHub repositório.

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

Etapa 2: configurar o driver

Você pode especificar as configurações do driver DataStax Java Cassandra criando um arquivo de configuração para seu aplicativo. Esse arquivo de configuração substitui as configurações padrão e instrui o driver a se conectar ao endpoint do serviço Amazon Keyspaces usando a porta 9142. Para obter uma lista de endpoints de serviço, consulte the section called “Service endpoints (Endpoints de serviço)”.

Crie um arquivo de configuração e salve-o na pasta de recursos do aplicativo — por exemplo, `src/main/resources/application.conf`. Abra `application.conf` e adicione as seguintes configurações.

1. Provedor de autenticação: defina o `advanced.auth-provider.class` para uma nova instância de `software.aws.mcs.auth.SigV4AuthProvider`. O SigV4 AuthProvider é o manipulador de autenticação fornecido pelo plug-in para realizar a autenticação SigV4.
2. Datacenter local: defina o valor `local-datacenter` para a região à qual você está se conectando. Por exemplo, se o aplicativo estiver se conectando a `cassandra.us-east-2.amazonaws.com`, defina o datacenter local como `us-east-2`. Para todos os disponíveis Regiões da AWS, consulte???. `slow-replica-avoidance = false`Defina para balancear a carga em relação a todos os nós disponíveis.
3. Idempotência — Defina o padrão `idempotence` para que o aplicativo configure o driver para sempre repetir as solicitações `true` de leitura/gravação/preparação/execução com falha. Essa é uma prática recomendada para aplicativos distribuídos que ajuda a lidar com falhas transitórias ao tentar novamente solicitações com falha.
4. SSL/TLS — Inicialize o SSL EngineFactory adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com. `class = DefaultSslEngineFactory` Forneça o caminho para o arquivo trustStore e a senha que você criou anteriormente. O Amazon Keyspaces não oferece suporte `hostname-validation` a pares, então defina essa opção como false.
5. Conexões — Crie pelo menos 3 conexões locais por endpoint `local.size = 3` configurando. Essa é uma prática recomendada que ajuda seu aplicativo a lidar com sobrecarga e picos de tráfego. Para obter mais informações sobre como calcular quantas conexões locais por endpoint seu aplicativo precisa com base nos padrões de tráfego esperados, consultethe section called “Como configurar conexões”.
6. Política de repetição — A `AmazonKeyspacesExponentialRetryPolicy` política de repetição do Amazon Keyspaces é uma alternativa à que vem com `DefaultRetryPolicy` o driver Cassandra. A principal diferença entre as duas políticas de repetição é que você pode configurar a quantidade de tentativas de repetição para atender `AmazonKeyspacesExponentialRetryPolicy` às suas necessidades. Por padrão, o número de tentativas de repetição do `AmazonKeyspacesExponentialRetryPolicy` é definido como 3. Além disso, a política de repetição do Amazon Keyspaces não retorna o genérico. `NoHostAvailableException` Em vez disso, a política de repetição do Amazon Keyspaces devolve a exceção original retornada pelo serviço. Para obter mais exemplos de código que

implementam políticas de repetição, consulte as políticas de [repetição do Amazon Keyspaces no Github](#).

7. Declarações preparadas — `prepare-on-all-nodes` Defina como false para otimizar o uso da rede.

```
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**
>
> Em vez de adicionar o caminho para o trustStore no arquivo de configuração, você também pode adicioná-lo diretamente no código do aplicativo ou aos seus argumentos da JVM.

### Etapa 3: executar o aplicativo

Este exemplo de código mostra um aplicativo de linha de comando simples que cria um pool de conexões com o Amazon Keyspaces usando o arquivo de configuração que criamos anteriormente. Ele confirma que a conexão foi estabelecida executando uma consulta simples.

```
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**
>
> Use um `try` bloco para estabelecer a conexão e garantir que ela esteja sempre fechada. Se você não usa um `try` bloco, lembre-se de fechar sua conexão para evitar o vazamento de recursos.

Conecte-se ao Amazon Keyspaces usando o driver DataStax Java 3.x para Apache Cassandra e o plug-in de autenticação SigV4

A seção a seguir descreve como usar o plug-in de autenticação SigV4 para o driver DataStax Java de código aberto 3.x para o Apache Cassandra acessar o Amazon Keyspaces. O plug-in está disponível no GitHub repositório.

O plug-in de autenticação SigV4 permite que você use credenciais do IAM para usuários e perfis ao se conectar ao Amazon Keyspaces. Em vez de exigir um nome de usuário e senha, esse plug-in assina solicitações de API usando chaves de acesso. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Etapa 1: pré-requisitos

Para executar esse exemplo de código, primeiro você precisa concluir as tarefas a seguir.

- Crie credenciais para seu usuário do IAM ou perfil do IAM seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”. Este tutorial pressupõe que as chaves de acesso sejam armazenadas como variáveis de ambiente. Para ter mais informações, consulte the section called “Como gerenciar chaves de acesso”.
- Siga as etapas em the section called “Antes de começar” para baixar o certificado digital Starfield, convertê-lo em um arquivo trustStore e anexar o arquivo trustStore nos argumentos da JVM ao seu aplicativo.
- Adicione o driver DataStax Java para Apache Cassandra ao seu projeto Java. Verifique se está usando uma versão do driver compatível com o Apache Cassandra 3.11.2. Para obter mais informações, consulte a documentação do driver DataStax Java para Apache Cassandra.
- Adicione o plug-in de autenticação ao seu aplicativo. O plug-in de autenticação suporta a versão 3.x do driver DataStax Java para Apache Cassandra. Se estiver usando o Apache Maven ou

um sistema de compilação que pode usar dependências do Maven, adicione as seguintes dependências ao seu arquivo `pom.xml`. Substitua a versão do plug-in pela versão mais recente, conforme mostrado no GitHub repositório.

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

Etapa 2: executar o aplicativo

Este exemplo de código mostra um aplicativo de linha de comando simples que cria um pool de conexões com o Amazon Keyspaces. Ele confirma que a conexão foi estabelecida executando uma consulta simples.

```
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"));
    }
}
```

Observações de uso:

Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

Consulte o seguinte repositório para políticas úteis de drivers Java, exemplos, e melhores práticas ao usar o driver Java com o Amazon Keyspaces: https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

## Como usar um driver de cliente Cassandra Python para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente Python. Para fornecer aos usuários e aplicativos credenciais para acesso programático aos recursos do Amazon Keyspaces, você pode executar uma das seguintes ações:

- Criar credenciais específicas do serviço associadas a um usuário específico AWS Identity and Access Management (IAM).
- Para aumentar a segurança, recomendamos criar chaves de acesso do IAM para usuários ou funções do IAM que são usadas em todos os AWS serviços. O plug-in de autenticação SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra permite que você autentique chamadas para o Amazon Keyspaces usando chaves de acesso do IAM em vez de nome de usuário e senha. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Tópicos

- Antes de começar
- Conecte-se ao Amazon Keyspaces usando o driver Python para Apache Cassandra e credenciais específicas do serviço
- Conecte-se ao Amazon Keyspaces usando o driver DataStax Python para Apache Cassandra e o plug-in de autenticação SigV4

Antes de começar

Você precisa concluir a tarefa seguinte antes de iniciar.

O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes. Para se conectar ao Amazon Keyspaces usando o TLS, você precisa baixar um certificado digital da Amazon e configurar o driver do Python para usar o TLS.

Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.


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


Conecte-se ao Amazon Keyspaces usando o driver Python para Apache Cassandra e credenciais específicas do serviço

O exemplo de código a seguir mostra como se conectar ao Amazon Keyspaces com um driver de cliente Python e credenciais específicas do serviço.

```
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)
```

Observações de uso:

1. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que o *ServiceUsernome* e a senha *ServicePassword*correspondam ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. Gerar credenciais específicas do serviço
3. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

Conecte-se ao Amazon Keyspaces usando o driver DataStax Python para Apache Cassandra e o plug-in de autenticação SigV4

A seção a seguir mostra como usar o plug-in de autenticação SigV4 para o driver DataStax Python de código aberto do Apache Cassandra para acessar o Amazon Keyspaces (para o Apache Cassandra).

Se ainda não tiver feito isso, comece criando credenciais para o usuário ou o perfil do IAM seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”. Este tutorial usa credenciais temporárias, o que requer um perfil do IAM. Para obter mais informações sobre credenciais temporárias, consulte the section called “Usar credenciais temporárias para se conectar ao Amazon Keyspaces”.

Em seguida, adicione o plug-in de autenticação Python SigV4 ao seu ambiente a partir do repositório. GitHub

```
pip install cassandra-sigv4
```

O exemplo de código a seguir mostra como se conectar ao Amazon Keyspaces usando o driver DataStax Python de código aberto para Cassandra e o plug-in de autenticação SigV4. O plug-in depende do AWS SDK para Python (Boto3). Ele usa `boto3.session` para obter credenciais temporárias.

```
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)
```

Observações de uso:

1. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que *aws_access_key_id*, *aws_secret_access_key* e *aws_session_token* correspondam ao `Access Key`, `Secret Access Key` e `Session Token` que você obteve usando `boto3.session`. Para obter mais informações, consulte Credenciais no AWS SDK for Python (Boto3).
3. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

# Como usar um driver de cliente Cassandra Node.js para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente Node.js. Para fornecer aos usuários e aplicativos credenciais para acesso programático aos recursos do Amazon Keyspaces, você pode executar uma das seguintes ações:

- Criar credenciais específicas do serviço associadas a um usuário específico AWS Identity and Access Management (IAM).
- Para aumentar a segurança, recomendamos criar chaves de acesso do IAM para usuários ou funções do IAM que são usadas em todos os AWS serviços. O plug-in de autenticação SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra permite que você autentique chamadas para o Amazon Keyspaces usando chaves de acesso do IAM em vez de nome de usuário e senha. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Tópicos


- Antes de começar
- Conecte-se ao Amazon Keyspaces usando o DataStax driver Node.js para Apache Cassandra e credenciais específicas do serviço
- Conecte-se ao Amazon Keyspaces usando o driver DataStax Node.js para Apache Cassandra e o plug-in de autenticação SigV4


## Antes de começar

Você precisa concluir a tarefa seguinte antes de iniciar.

O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes. Para se conectar ao Amazon Keyspaces usando o TLS, você precisa baixar um certificado digital da Amazon e configurar o driver do Python para usar o TLS.

Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.

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

Conecte-se ao Amazon Keyspaces usando o DataStax driver Node.js para Apache Cassandra e credenciais específicas do serviço

Configure seu driver para usar o certificado digital Starfield para TLS e autenticar usando credenciais específicas do serviço. Por exemplo: .

```
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}`));
```

Observações de uso:

1. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que o *ServiceUsernome* e a senha *ServicePassword*correspondam ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. Gerar credenciais específicas do serviço
3. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

Conecte-se ao Amazon Keyspaces usando o driver DataStax Node.js para Apache Cassandra e o plug-in de autenticação SigV4

A seção a seguir mostra como usar o plug-in de autenticação SigV4 para o driver DataStax Node.js de código aberto do Apache Cassandra para acessar o Amazon Keyspaces (para o Apache Cassandra).

Se ainda não tiver feito isso, crie credenciais para o usuário do IAM ou o perfil do IAM seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”.

Adicione o plug-in de autenticação SigV4 Node.js ao seu aplicativo a partir do GitHub repositório.
O plug-in é compatível com a versão 4.x do driver DataStax Node.js para Cassandra e depende do AWS SDK para Node.js. Ele usa `AWSCredentialsProvider` para obter credenciais.

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

Este exemplo de código mostra como definir uma instância específica da região `SigV4AuthProvider` como provedor de autenticação.

```
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}`));
```

Observações de uso:

1. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que o *acesso KeyId* e a chave *secreta AccessKey* correspondam à chave de acesso e à chave de acesso secreta que você obteve usando`AWSCredentialsProvider`. Para obter mais informações, consulte Configuração de credenciais em Node.js no AWS SDK para JavaScript em Node.js.
3. Para armazenar chaves de acesso fora do código, consulte as melhores práticas em the section called “Como gerenciar chaves de acesso”.
4. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

## Como usar um driver de cliente Cassandra .NET Core para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente .NET Core. As etapas de configuração variam de acordo com o ambiente e o sistema operacional. Talvez seja necessário modificá-las adequadamente. O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes. Para se conectar ao Amazon

Keyspaces usando o TLS, você precisa baixar um certificado digital da Starfield e configurar seu driver para usar o TLS.

1. Faça o download do certificado Starfield e salve-o em um diretório local, anotando o caminho. A seguir está um exemplo de uso 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. Instale o Cassandrac SharpDriver por meio do nuget, usando o console do nuget.

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

3. O exemplo a seguir usa um projeto de console C# do.NET Core para se conectar ao Amazon Keyspaces e executar uma consulta.

```
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);
            }
        }
    }
}
```

Observações de uso:

a. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.

b. Certifique-se de que o *ServiceUsernome* e a senha *ServicePassword*correspondam ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. Gerar credenciais específicas do serviço

c. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

## Como usar um driver de cliente Cassandra Go para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente Go. Para fornecer aos usuários e aplicativos credenciais para acesso programático aos recursos do Amazon Keyspaces, você pode executar uma das seguintes ações:

- Criar credenciais específicas do serviço associadas a um usuário específico AWS Identity and Access Management (IAM).
- Para aumentar a segurança, recomendamos criar chaves de acesso do IAM para usuários e funções do IAM que são usadas em todos os AWS serviços. O plug-in de autenticação SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra permite que você autentique chamadas para o Amazon Keyspaces usando chaves de acesso do IAM em vez de nome de usuário e senha. Para ter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Tópicos


- Antes de começar
- Conecte-se ao Amazon Keyspaces usando o driver Gocql para Apache Cassandra e credenciais específicas do serviço
- Conecte-se ao Amazon Keyspaces usando o driver Go para Apache Cassandra e o plug-in de autenticação SigV4


## Antes de começar

Você precisa concluir a tarefa seguinte antes de iniciar.

O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes. Para se conectar ao Amazon Keyspaces usando o TLS, você precisa baixar um certificado digital da Amazon e configurar o driver do Python para usar o TLS.

Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.

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

Conecte-se ao Amazon Keyspaces usando o driver Gocql para Apache Cassandra e credenciais específicas do serviço

1. Crie um diretório para seu aplicativo.

```
mkdir ./gocqlexample
```

2. Navegue até o novo diretório.

```
cd gocqlexample
```

3. Crie um arquivo para o aplicativo.

```
touch cqlapp.go
```

4. Baixe o driver Go.

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

5. Adicione o código de exemplo a seguir ao arquivo 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()
}
```

Observações de uso:

a. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.

b. Certifique-se de que o *ServiceUsernome* e a senha *ServicePassword*correspondam ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. Gerar credenciais específicas do serviço

c. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

6. Crie o programa.

```
go build cqlapp.go
```

7. Execute o programa.

```
./cqlapp
```

Conecte-se ao Amazon Keyspaces usando o driver Go para Apache Cassandra e o plug-in de autenticação SigV4

A seção a seguir descreve como usar o plug-in de autenticação SigV4 do driver Go de código aberto para acessar o Amazon Keyspaces (para Apache Cassandra).

Se ainda não tiver feito isso, crie credenciais para o usuário do IAM ou o perfil do IAM seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”.

Adicione o plug-in de autenticação Go SigV4 ao seu aplicativo a partir do GitHub repositório. O plug-in é compatível com a versão 1.2.x do driver Go de código aberto para Cassandra e depende do SDK for Go. AWS

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

Neste exemplo de código, o endpoint do Amazon Keyspaces é representado pela classe `Cluster`. Ele usa a propriedade `AwsAuthenticator` para o autenticador do cluster para obter credenciais.

```
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)
    }
}
```

Observações de uso:

1. Substitua "*path_to_file*/sf-class2-root.crt" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que o *AccessKeyID* e a *SecretAccesschave* correspondam à chave de acesso e à chave de acesso secreta que você obteve usando`AwsAuthenticator`. Para obter mais informações, consulte Configuração do AWS SDK para Go em AWS SDK for Go.
3. Para armazenar chaves de acesso fora do código, consulte as melhores práticas em the section called “Como gerenciar chaves de acesso”.
4. Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

## Como usar um driver de cliente Cassandra Perl para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de cliente Perl. Para esse exemplo de código, usamos o Perl 5. O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes.

> **Important**
>
> Para criar uma conexão segura, nossos exemplos de código usam o certificado digital Starfield para autenticar o servidor antes de estabelecer a conexão TLS. O driver Perl não valida o certificado Amazon SSL do servidor, o que significa que você não pode confirmar que está se conectando ao Amazon Keyspaces. A segunda etapa, configurar o driver para usar o TLS ao se conectar ao Amazon Keyspaces, ainda é necessária e garante que os dados transferidos entre o cliente e o servidor sejam criptografados.

1. Baixe o driver Cassandra DBI em https://metacpan.org/pod/DBD::Cassandra e instale o driver em seu ambiente Perl. As etapas exatas dependem do ambiente. A seguir, um exemplo comum.

```
cpanm DBD::Cassandra
```

2. Crie um arquivo para o aplicativo.

```
touch cqlapp.pl
```

3. Adicione o código de exemplo a seguir ao arquivo 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**
>
> Certifique-se de que o *ServiceUsernome* e a senha *ServicePassword*correspondam ao nome de usuário e senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. Gerar credenciais específicas do serviço

> **Note**
>
> Para obter uma lista de endpoints disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

4. Execute o aplicativo.

```
perl cqlapp.pl
```

## Tutorial: Conectando-se ao Amazon Keyspaces a partir do Amazon Elastic Kubernetes Service

Este tutorial mostra as etapas necessárias para configurar um cluster do Amazon Elastic Kubernetes Service (Amazon EKS) para hospedar um aplicativo em contêiner que se conecta ao Amazon Keyspaces usando a autenticação SigV4.

O Amazon EKS é um serviço gerenciado que elimina a necessidade de instalar, operar e manter seu próprio plano de controle do Kubernetes. O Kubernetes é um sistema de código aberto que automatiza o gerenciamento, a escalabilidade e a implantação de aplicações em contêineres.

O tutorial fornece step-by-step orientação para configurar, criar e implantar um aplicativo Java em contêiner no Amazon EKS. Na última etapa, você executa o aplicativo para gravar dados em uma tabela do Amazon Keyspaces.

Tópicos

- Pré-requisitos do tutorial
- Etapa 1: configurar o cluster Amazon EKS e configurar as permissões do IAM

- Etapa 2: configurar o aplicativo
- Etapa 3: Crie a imagem do aplicativo e faça o upload do arquivo Docker no seu repositório Amazon ECR
- Etapa 4: Implantar o aplicativo no Amazon EKS e gravar dados na sua tabela do Amazon Keyspaces
- Etapa 5: Limpeza (opcional)

## Pré-requisitos do tutorial

Crie os seguintes AWS recursos antes de começar com o tutorial

1. Antes de começar este tutorial, siga as instruções AWS de configuração emComo acessar o Amazon Keyspaces (para Apache Cassandra). Essas etapas incluem a inscrição AWS e a criação de um diretor AWS Identity and Access Management (IAM) com acesso ao Amazon Keyspaces.
2. Crie um keyspace do Amazon Keyspaces com o nome `aws` e uma tabela com o nome `user` no qual você pode gravar a partir do aplicativo em contêiner executado no Amazon EKS posteriormente neste tutorial. Você pode fazer isso com o AWS CLI ou usando`cqlsh`.

   AWS CLI

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

   Para confirmar que o keyspace foi criado, você pode usar o comando a seguir.

   ```
   aws keyspaces list-keyspaces
   ```

   Para criar a tabela, você pode usar o comando a seguir.

   ```
   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}]'
   ```

   Para confirmar que sua tabela foi criada, você pode usar o comando a seguir.

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

Para obter mais informações, consulte criar espaço de teclas e criar tabela na Referência de AWS CLI comandos.

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
);
```

Para verificar se sua tabela foi criada, você pode usar a instrução a seguir.

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

Sua tabela deve estar listada na saída dessa declaração. Observe que pode haver um atraso até que a tabela seja criada. Para ter mais informações, consulte the section called “CRIAR TABELA”.

3. Crie um cluster Amazon EKS com um tipo de nó Fargate - Linux. O Fargate é um mecanismo de computação sem servidor que permite implantar Kubernetes Pods sem gerenciar instâncias do Amazon Amazon EC2. Para seguir este tutorial sem precisar atualizar o nome do cluster em todos os comandos de exemplo, crie um cluster com o nome `my-eks-cluster` seguindo as instruções em Introdução ao Amazon EKS — `eksctl` no Guia do usuário do Amazon EKS. Quando seu cluster for criado, verifique se seus nós e os dois pods padrão estão funcionando e saudáveis. Você pode fazer isso com o comando a seguir.

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

Você deve ver algo semelhante a essa saída.

```
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. Instalar o Docker. Para obter instruções sobre como instalar o Docker em uma instância do Amazon EC2, consulte Instalar o Docker no Guia do usuário do Amazon Elastic Container Registry.

   O Docker está disponível em muitos sistemas operacionais diferentes, incluindo a maioria das distribuições modernas do Linux, como o Ubuntu e até no MacOS e no Windows. Para obter mais informações sobre como instalar o Docker no seu sistema operacional, consulte o Guia de instalação do Docker.

5. Crie um repositório do Amazon ECR. O Amazon ECR é um serviço AWS gerenciado de registro de imagens de contêineres que você pode usar com sua CLI preferida para enviar, extrair e gerenciar imagens do Docker. Para obter mais informações sobre os repositórios do Amazon ECR, consulte o Guia do usuário do Amazon Elastic Container Registry. Você pode usar o comando a seguir para criar um repositório com o nome`my-ecr-repository`.

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

Depois de concluir as etapas de pré-requisito, vá para the section called “Etapa 1: Configurar o cluster Amazon EKS”.

## Etapa 1: configurar o cluster Amazon EKS e configurar as permissões do IAM

Configure o cluster do Amazon EKS e crie os recursos do IAM necessários para permitir que uma conta de serviço do Amazon EKS se conecte à sua tabela do Amazon Keyspaces

1. Crie um provedor Open ID Connect (OIDC) para o cluster Amazon EKS. Isso é necessário para usar funções do IAM para contas de serviço. Para obter mais informações sobre provedores OIDC e como criá-los, consulte Criação de um provedor IAM OIDC para seu cluster no Guia do usuário do Amazon EKS.

   a. Crie o provedor de identidade OIDC do IAM para o cluster com o comando a seguir. Este exemplo pressupõe que o nome do seu cluster sejamy`-eks-cluster`. Se você tiver um

cluster com um nome diferente, lembre-se de atualizar o nome em todos os comandos future.

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

b. Confirme se o provedor de identidade do OIDC foi registrado no IAM com o comando a seguir.

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

A saída deve ser semelhante a esta. Anote o Amazon Resource Name (ARN) do OIDC. Você precisará dele na próxima etapa ao criar uma política de confiança para a conta de serviço.

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

2. Crie uma conta de serviço para o cluster Amazon EKS. As contas de serviço fornecem uma identidade para processos que são executados em um pod. Um pod é o menor e mais simples objeto Kubernetes que você pode usar para implantar um aplicativo em contêiner. Em seguida, crie uma função do IAM que a conta de serviço possa assumir para obter permissões para os recursos. Você pode acessar qualquer AWS serviço de um pod que tenha sido configurado para usar uma conta de serviço que possa assumir uma função do IAM com permissões de acesso a esse serviço.

   a. Crie um novo namespace para a conta de serviço. Um namespace ajuda a isolar os recursos do cluster criados para este tutorial. Você pode criar um novo namespace usando o comando a seguir.

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

   b. Para usar um namespace personalizado, você precisa associá-lo a um perfil do Fargate. O código a seguir é um exemplo disso.

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

c. Crie uma conta de serviço com o nome `my-eks-serviceaccount` no namespace `my-eks-namespace` do seu cluster Amazon EKS usando o comando a seguir.

```
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. Execute o comando a seguir para criar um arquivo de política de confiança que instrua a função do IAM a confiar em sua conta de serviço. Essa relação de confiança é necessária antes que um diretor possa assumir uma função. Você precisa fazer as seguintes edições no arquivo:

- Para o`Principal`, insira o ARN que o IAM retornou ao `list-open-id-connect-providers` comando. O ARN contém o número da sua conta e a região.
- Na `condition` declaração, substitua o Região da AWS e o ID do OIDC.
- Confirme se o nome e o namespace da conta de serviço estão corretos.

Você precisa anexar o arquivo de política de confiança na próxima etapa ao criar a função do 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
```

Opcional: você também pode adicionar várias entradas nas `StringLike` condições `StringEquals` ou para permitir que várias contas de serviço ou namespaces assumam a função. Para permitir que sua conta de serviço assuma uma função do IAM em uma AWS conta diferente, consulte Permissões do IAM entre contas no Guia do usuário do Amazon EKS.

3. Crie uma função do IAM com o nome a ser assumido `my-iam-role` pela conta de serviço do Amazon EKS. Anexe o arquivo de política de confiança criado na última etapa à função. A política de confiança especifica a conta de serviço e o provedor do OIDC em que a função do IAM pode confiar.

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

4. Atribua as permissões da função do IAM ao Amazon Keyspaces anexando uma política de acesso.

   a. Anexe uma política de acesso para definir as ações que a função do IAM pode realizar em recursos específicos do Amazon Keyspaces. Para este tutorial, usamos a política AWS gerenciada`AmazonKeyspacesFullAccess`, porque nosso aplicativo gravará dados na sua tabela do Amazon Keyspaces. No entanto, como prática recomendada, é recomendável criar políticas de acesso personalizadas que implementem o princípio

de privilégios mínimos. Para ter mais informações, consulte the section called “Como o Amazon Keyspaces funciona com o IAM”.

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

Confirme se a política foi anexada com sucesso à função do IAM com a seguinte declaração.

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

A saída deve ser algo parecido com isso.

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

b. Anote a conta de serviço com o Amazon Resource Name (ARN) da função do IAM que ela pode assumir. Certifique-se de atualizar o ARN da função com o ID da sua conta.

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

5. Confirme se a função do IAM e a conta de serviço estão configuradas corretamente.

a. Confirme se a política de confiança da função do IAM está configurada corretamente com a declaração a seguir.

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

A saída deve ser semelhante a esta.

```
{
    "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. Confirme se a conta de serviço do Amazon EKS está anotada com a função do IAM.

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

A saída deve ser semelhante a esta.

```
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>
[...]
```

Depois de criar a conta de serviço do Amazon EKS, a função do IAM e configurar os relacionamentos e permissões necessários, prossiga parathe section called “Etapa 2: configurar o aplicativo”.

## Etapa 2: configurar o aplicativo

Nesta etapa, você cria seu aplicativo que se conecta ao Amazon Keyspaces usando o plug-in SigV4. Você pode visualizar e baixar o exemplo de aplicativo Java no repositório de código de exemplo do Amazon Keyspaces no Github. Ou você pode acompanhar usando seu próprio aplicativo, certificando-se de concluir todas as etapas de configuração.

Configure seu aplicativo e adicione as dependências necessárias.

1. Você pode baixar o aplicativo Java de exemplo clonando o repositório Github usando o comando a seguir.

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

2. Depois de baixar o repositório do Github, descompacte o arquivo baixado e navegue até o `resources` diretório do arquivo. `application.conf`

   a. Configuração do aplicativo

   Nesta etapa, você configura o plug-in de autenticação SigV4. Você pode usar o exemplo a seguir em seu aplicativo. Se ainda não tiver feito isso, você precisa gerar suas chaves de acesso do IAM (um ID de chave de acesso e uma chave de acesso secreta) e salvá-las em seu arquivo de AWS configuração ou como variáveis de ambiente. Para obter instruções detalhadas, consulte the section called “Credenciais necessárias para autenticação AWS ”. Atualize a AWS região e o endpoint de serviço do Amazon Keyspaces conforme necessário. Para obter mais endpoints de serviço, consultethe section called “Service endpoints (Endpoints de serviço)”. Substitua a localização da loja confiável, o nome da loja confiável e a senha da loja confiável pelos seus.

```
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. Adicione a dependência do módulo STS.

Isso adiciona a capacidade de usar um `WebIdentityTokenCredentialsProvider` que retorna AWS as credenciais que o aplicativo precisa fornecer para que a conta de serviço possa assumir a função do IAM. Você pode fazer isso com base no exemplo a seguir.

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

c. Adicione a dependência SigV4.

Este pacote implementa o plug-in de autenticação SigV4 que é necessário para se autenticar no Amazon Keyspaces

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

3. Adicione uma dependência de registro.

Sem registros, é impossível solucionar problemas de conexão. Neste tutorial, usamos `slf4j` como estrutura de registro e armazenamos `logback.xml` a saída do registro. Definimos o nível de registro `debug` para estabelecer a conexão. Você pode usar o exemplo a seguir para adicionar a dependência.

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

Você pode usar o seguinte trecho de código para configurar o registro em log.

```
<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**
>
> O `debug` nível é necessário para investigar falhas de conexão. Depois de se conectar com sucesso ao Amazon Keyspaces a partir do seu aplicativo, você pode alterar o nível de registro para `info` ou `warning` conforme necessário.

## Etapa 3: Crie a imagem do aplicativo e faça o upload do arquivo Docker no seu repositório Amazon ECR

Nesta etapa, você compila o aplicativo de exemplo, cria uma imagem do Docker e envia a imagem para o seu repositório Amazon ECR.

Crie seu aplicativo, crie uma imagem do Docker e envie-a para o Amazon Elastic Container Registry

1. Defina variáveis de ambiente para a compilação que definem sua Região da AWS. Substitua as regiões nos exemplos pelas suas.

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

2. Compile seu aplicativo com o Apache Maven versão 3.6.3 ou superior usando o comando a seguir.

```
mvn clean install
```

Isso cria um JAR arquivo com todas as dependências incluídas no `target` diretório.

3. Recupere o URI do repositório ECR necessário para a próxima etapa com o comando a seguir. Certifique-se de atualizar a região para a que você está usando.

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

A saída deve ser semelhante ao exemplo a seguir.

```
"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. No diretório raiz do aplicativo, crie a imagem do Docker usando o URI do repositório da última etapa. Modifique o arquivo Docker conforme necessário. No comando build, certifique-se de substituir o ID da sua conta e configurá-lo Região da AWS para a região em que o repositório do Amazon ECR `my-ecr-repository` está localizado.

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

5. Recupere um token de autenticação para enviar a imagem do Docker para o Amazon ECR. Você pode fazer isso com o comando a seguir.

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

6. Primeiro, verifique se há imagens existentes no seu repositório Amazon ECR. É possível usar o seguinte comando.

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

Em seguida, envie a imagem do Docker para o repositório. É possível usar o seguinte comando.

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

## Etapa 4: Implantar o aplicativo no Amazon EKS e gravar dados na sua tabela do Amazon Keyspaces

Nesta etapa do tutorial, você configura a implantação do Amazon EKS para seu aplicativo e confirma se o aplicativo está em execução e pode se conectar ao Amazon Keyspaces.

Para implantar um aplicativo no Amazon EKS, você precisa definir todas as configurações relevantes em um arquivo chamado`deployment.yaml`. Esse arquivo é então usado pelo Amazon EKS para implantar o aplicativo. Os metadados no arquivo devem conter as seguintes informações:

- Nome do aplicativo: o nome do aplicativo. Para este tutorial, usamos`my-keyspaces-app`.
- Namespace Kubernetes: o namespace do cluster Amazon EKS. Para este tutorial, usamos`my-eks-namespace`.
- Nome da conta de serviço Amazon EKS o nome da conta de serviço Amazon EKS. Para este tutorial, usamos`my-eks-serviceaccount`.
- nome da imagem o nome da imagem do aplicativo. Para este tutorial, usamos`my-keyspaces-app`.
- URI da imagem: o URI da imagem Docker do Amazon ECR.
- AWS ID da conta sua ID AWS da conta.
- ARN da função do IAM: o ARN da função do IAM criada para a conta de serviço assumir. Para este tutorial, usamos`my-iam-role`.
- Região da AWS do cluster Amazon EKS em Região da AWS que você criou seu cluster Amazon EKS.

Nesta etapa, você implanta e executa o aplicativo que se conecta ao Amazon Keyspaces e grava dados na tabela.

1. Configure o arquivo `deployment.yaml`. Você precisa substituir os seguintes valores:

   - `name`
   - `namespace`
   - `serviceAccountName`
   - `image`
   - `AWS_ROLE_ARN value`
   - A Região da AWS entrada `CASSANDRA_HOST`
   - `AWS_REGION`

   Você pode usar o arquivo a seguir como exemplo.

```
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. Implante o `deployment.yaml`.

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

A saída deve ser algo parecido com isso.

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

3. Verifique o status do pod no seu namespace do cluster Amazon EKS.

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

A saída deve ser semelhante a este exemplo:

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

Para obter mais detalhes, você pode usar o comando a seguir.

```
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. Verifique os registros do pod para confirmar que seu aplicativo está em execução e pode se conectar à sua tabela do Amazon Keyspaces. Você pode fazer isso com o comando a seguir. Certifique-se de substituir o nome da sua implantação.

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

Você deve conseguir ver as entradas de registro do aplicativo confirmando a conexão com o Amazon Keyspaces, como no exemplo abaixo.

```
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. Execute a seguinte consulta CQL em sua tabela do Amazon Keyspaces para confirmar que uma linha de dados foi gravada em sua tabela:

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

A seguinte saída deverá ser mostrada:

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

## Etapa 5: Limpeza (opcional)

Siga estas etapas para remover todos os recursos criados neste tutorial.

Remova os recursos criados neste tutorial

1. Exclua sua implantação. Você pode usar o comando a seguir para fazer isso.

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

2. Exclua o cluster Amazon EKS e todos os pods contidos nele. Isso também exclui recursos relacionados, como a conta de serviço e o provedor de identidade OIDC. Você pode usar o comando a seguir para fazer isso.

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

3. Exclua a função do IAM usada para a conta de serviço Amazon EKS com permissões de acesso ao Amazon Keyspaces. Primeiro, você precisa remover a política gerenciada que está anexada à função.

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

   Em seguida, você pode excluir a função usando o comando a seguir.

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

   Para obter mais informações, consulte Excluir uma função do IAM (AWS CLI) no Guia do usuário do IAM.

4. Exclua o repositório Amazon ECR, incluindo todas as imagens armazenadas nele. Você pode fazer isso usando o comando a seguir.

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

   Observe que a `force` sinalização é necessária para excluir um repositório que contém imagens. Para excluir sua imagem primeiro, você pode fazer isso usando o comando a seguir.

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

   Para obter mais informações, consulte Excluir uma imagem no Guia do usuário do Amazon Elastic Container Registry.

5. Exclua o keyspace e a tabela do Amazon Keyspaces. A exclusão do espaço de teclas exclui automaticamente todas as tabelas desse espaço de teclas. Você pode usar uma das seguintes opções para fazer isso.

AWS CLI

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

Para confirmar que o keyspace foi excluído, você pode usar o comando a seguir.

```
aws keyspaces list-keyspaces
```

Para excluir a tabela primeiro, você pode usar o comando a seguir.

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

Para confirmar que sua tabela foi excluída, você pode usar o comando a seguir.

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

Para obter mais informações, consulte excluir espaço de teclas e excluir tabela na Referência de AWS CLI comandos.

cqlsh

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

Para verificar se seus espaços de chave foram excluídos, você pode usar a seguinte declaração.

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

Seu keyspace não deve ser listado na saída desta declaração. Observe que pode haver um atraso até que os espaços de teclas sejam excluídos. Para ter mais informações, consulte the section called “DESCARTAR ESPAÇO DE CHAVES”.

Para excluir a tabela primeiro, você pode usar o comando a seguir.

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

Para confirmar que sua tabela foi excluída, você pode usar o comando a seguir.

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

Sua tabela não deve ser listada na saída dessa declaração. Observe que pode haver um atraso até que a tabela seja excluída. Para ter mais informações, consulte the section called “DESCARTAR TABELA”.

# Tutorial: Como conectar-se ao Amazon Keyspaces usando um endpoint da VPC de interface

Este tutorial orienta você ao longo da configuração e do uso de um endpoint da VPC de interface para o Amazon Keyspaces.

Os endpoints da VPC de interface permitem a comunicação privada entre a nuvem privada virtual (VPC) em execução no Amazon VPC e no Amazon Keyspaces. Os endpoints VPC de interface são alimentados por AWS PrivateLink, que é um AWS serviço que permite a comunicação privada entre VPCs e serviços. AWS Para ter mais informações, consulte the section called “Usar VPC endpoints da interface do ”.

Tópicos


- Pré-requisitos e considerações do tutorial
- Etapa 1: iniciar uma instância do Amazon EC2
- Etapa 2: configurar a instância do Amazon EC2
- Etapa 3: criar um endpoint da VPC para o Amazon Keyspaces
- Etapa 4: configurar permissões para a conexão do endpoint da VPC
- Etapa 5: Configurar o monitoramento com CloudWatch
- Etapa 6: (opcional) práticas recomendadas para configurar o tamanho do pool de conexões para seu aplicativo
- Etapa 7: (opcional) limpeza


## Pré-requisitos e considerações do tutorial

Antes de começar este tutorial, siga as instruções AWS de configuração emComo acessar o Amazon Keyspaces (para Apache Cassandra). Essas etapas incluem a inscrição AWS e a criação de um diretor AWS Identity and Access Management (IAM) com acesso ao Amazon Keyspaces. Anote

o nome do usuário do IAM e as chaves de acesso, pois você precisará delas mais adiante neste tutorial.

Crie um espaço de chave com o nome `myKeyspace` e pelo menos uma tabela para testar a conexão usando o endpoint da VPC mais adiante neste tutorial. Você pode encontrar instruções detalhadas em Conceitos básicos.

Depois de concluir as etapas de pré-requisito, vá para Etapa 1: iniciar uma instância do Amazon EC2.

## Etapa 1: iniciar uma instância do Amazon EC2

Nesta etapa, você inicia uma instância do Amazon EC2 em sua Amazon VPC padrão. Você então pode criar e usar um endpoint da VPC para Amazon Keyspaces.

Para executar uma instância do Amazon EC2

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Iniciar instância e faça o seguinte:

   No painel do console do EC2, na caixa Iniciar instância, escolha Iniciar instância e escolha Iniciar instância entre as opções exibidas.

   Em Nome e etiquetas, no Nome, insira um nome descritivo para a instância.

   Em Imagens de aplicações e SO (imagem de máquina da Amazon):

   - Escolha Início rápido e escolha Ubuntu. Este é o sistema operacional (SO) de sua instância.
   - Em imagem de máquina da Amazon (AMI), você pode usar a imagem padrão marcada como Qualificado para o nível gratuito. Uma imagem de máquina da Amazon (AMI) é uma configuração básica que serve de modelo para sua instância.

   Em Tipo de instância:

   - Na lista Tipo de instância, escolha o tipo de instância t2.micro, que está selecionado por padrão.

   Em Par de chaves (login), em Nome do par de chaves, escolha uma das seguintes opções para este tutorial:

- Se você não tiver um par de chaves do Amazon EC2, escolha Create a new key pair (Criar um novo par de chaves) e siga as instruções. Será solicitado que você faça download de um arquivo de chave privada (arquivo .pem). Você precisará deste arquivo mais tarde quando fizer login na instância do Amazon EC2, então anote o caminho do arquivo.
- Se você já tiver um par de chaves existente do Amazon EC2, vá para Select a key pair (Selecionar um par de chaves) e escolha o seu par de chaves na lista. Você já deverá ter o arquivo de chave privada (arquivo .pem) disponível para fazer login na instância do Amazon EC2.

Em Configurações de rede:

- Selecione a opção Editar.
- Escolha Select an existing security group (Selecionar um grupo de segurança existente).
- Na lista de grupos de segurança, escolha default (padrão). Este é o grupo de segurança padrão para sua VPC.

Continue até o Resumo.

- Analise o resumo da configuração da instância no painel Resumo. Quando estiver tudo pronto, escolha Iniciar instância.

3. Na tela de conclusão da nova instância do Amazon EC2, escolha o quadro Conectar-se à instância. A próxima tela mostra as informações necessárias e as etapas necessárias para se conectar à sua nova instância. Anote as seguintes informações:

   - O exemplo de comando para proteger o arquivo de chave
   - String de conexão
   - O nome DNS público IPv4

   Depois de anotar as informações desta página, você poderá prosseguir para a próxima etapa deste tutorial (Etapa 2: configurar a instância do Amazon EC2).

> **Note**
>
> Serão necessários alguns minutos para que a sua instância do Amazon EC2 se torne disponível. Antes de ir para a próxima etapa, verifique se o Estado da instância é `running` e se todas as suas Verificações de status foram aprovadas.

## Etapa 2: configurar a instância do Amazon EC2

Quando sua instância do Amazon EC2 estiver disponível, você poderá fazer login e prepará-la para ser usada pela primeira vez.

> **Note**
>
> As etapas a seguir assumem que você está se conectando à sua instância do Amazon EC2 de um computador que executa o Linux. Para outras formas de se conectar, consulte Conecte-se à sua instância Linux no Guia do usuário do Amazon EC2.

Para configurar sua instância do Amazon EC2

1. É necessário autorizar o tráfego SSH de entrada para a sua instância do Amazon EC2. Para fazer isso, crie um novo grupo de segurança EC2 e atribua o grupo de segurança à sua instância do EC2.

   a. No painel de navegação, escolha Grupos de segurança.

   b. Escolha Create Security Group. Na janela Security group, faça o seguinte:

      - Nome do grupo de segurança: insira um nome para o grupo de segurança. Por exemplo: `my-ssh-access`
      - Descrição: digite uma breve descrição para o grupo de segurança.
      - VPC: escolha sua VPC padrão.
      - Na seção Regras de entrada, escolha Adicionar regra e faça o seguinte:
        - Tipo: escolha SSH.
        - Origem: escolha Meu IP.
        - Escolha Adicionar regra.

Na parte inferior da página, confirme as configurações e escolha Criar grupo de segurança.

c. No painel de navegação, escolha Instâncias.

d. Escolha a instância do Amazon EC2 que você iniciou em Etapa 1: iniciar uma instância do Amazon EC2.

e. Escolha Ações, escolha Segurança e, em seguida, escolha Alterar grupos de segurança.

f. Em Alterar grupos de segurança, selecione o grupo de segurança que você criou anteriormente neste procedimento (por exemplo, `my-ssh-access`). O grupo de segurança `default` existente também deve ser selecionado. Confirme as configurações e escolha Atribuir grupos de segurança.

2. Use o comando a seguir para proteger seu arquivo de chave privada do acesso. Se você pular essa etapa, a conexão falhará.

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

3. Use o comando `ssh` para fazer login na sua instância do Amazon EC2 conforme o exemplo a seguir.

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

Você precisa especificar seu arquivo de chave privada (arquivo .pem) e o nome DNS público da sua instância. (Consulte Etapa 1: iniciar uma instância do Amazon EC2.)

O ID de login é `ubuntu`. Nenhuma senha é necessária.

Para obter mais informações sobre como permitir conexões com sua instância do Amazon EC2 e para AWS CLI obter instruções, consulte Autorizar tráfego de entrada para suas instâncias Linux no Guia do usuário do Amazon EC2.

4. Faça download e instale a versão mais recente da AWS Command Line Interface.

a. Instalar o `unzip`.

```
sudo apt install unzip
```

b. Baixe o arquivo `zip` com a AWS CLI.

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

c. Descompacte o arquivo.

```
unzip awscliv2.zip
```

d. Instale AWS CLI o.

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

e. Confirme a versão da AWS CLI instalação.

```
aws --version
```

O resultado deve ser semelhante ao seguinte:

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

5. Configure suas AWS credenciais, conforme mostrado no exemplo a seguir. Insira o ID da chave de AWS acesso, a chave secreta e o nome da região padrão quando solicitado.

```
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. Você precisa usar uma conexão `cqlsh` com o Amazon Keyspaces para confirmar que seu endpoint da VPC foi configurado corretamente. Se você usa seu ambiente local ou o editor Amazon Keyspaces CQL no AWS Management Console, a conexão passa automaticamente pelo endpoint público em vez do seu endpoint VPC. Para usar `cqlsh` para testar sua conexão de endpoint da VPC neste tutorial, conclua as instruções de configuração em Usar `cqlsh` para se conectar ao Amazon Keyspaces.

Agora, você está pronto para criar um endpoint da VPC para o Amazon Keyspaces.

# Etapa 3: criar um endpoint da VPC para o Amazon Keyspaces

Nesta etapa, você cria um endpoint da VPC para o Amazon Keyspaces usando o AWS CLI. Para criar o endpoint da VPC usando o console da VPC, você pode seguir as instruções Criar um endpoint da VPC no Guia do AWS PrivateLink . Ao filtrar pelo Nome do serviço, insira **`Cassandra`**.

Para criar um VPC endpoint usando o AWS CLI

1. Antes de começar, verifique se você pode se comunicar com o Amazon Keyspaces usando seu endpoint público.

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

   A saída mostra uma lista de tabelas do Amazon Keyspaces que estão contidas no espaço de chave especificado. Se você não tiver tabelas, a lista estará vazia.

   ```
   {
       "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. Verifique se o Amazon Keyspaces é um serviço disponível para criar VPC endpoints na região atual. AWS (O comando é mostrada em negrito, seguido pelo exemplo de saída).

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

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

```
        "com.amazonaws.us-east-1.cassandra-fips"
    ]
}
```

No exemplo de saída, o Amazon Keyspaces é um dos serviços disponíveis, portanto, você pode prosseguir com a criação de um endpoint da VPC para ele.

3. Determine o identificador da VPC.

```
aws ec2 describe-vpcs

{
    "Vpcs": [
        {
            "VpcId": "vpc-a1234bcd",
            "InstanceTenancy": "default",
            "State": "available",
            "DhcpOptionsId": "dopt-8454b7e1",
            "CidrBlock": "111.31.0.0/16",
            "IsDefault": true
        }
    ]
}
```

No exemplo de saída, o ID da VPC é `vpc-a1234bcd`.

4. Use um filtro para coletar detalhes sobre as sub-redes da 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,


        }
    ]
}
```

No exemplo de saída, há duas IDs de sub-rede disponíveis: `subnet-920aacf9` e. `subnet-4c713600`

5. Crie o endpoint da VPC. Para o parâmetro `--vpc-id`, especifique o ID da VPC da etapa anterior. Para o parâmetro `--subnet-id`, especifique as IDs da sub-rede da etapa anterior. Use o parâmetro `--vpc-endpoint-type` para definir o endpoint como uma interface. Para obter mais informações sobre o comando , consulte `create-vpc-endpoint` na Referência de comandos da AWS CLI .

```
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"
    }
}
```

```
}
```

## Etapa 4: configurar permissões para a conexão do endpoint da VPC

Os procedimentos nesta etapa demonstram como configurar regras e permissões para usar o endpoint da VPC com o Amazon Keyspaces.

Para configurar uma regra de entrada para o novo endpoint para permitir tráfego de entrada TCP

1. No console da VPC da Amazon, no painel esquerdo, escolha Endpoints e escolha o endpoint que você criou na etapa anterior.
2. Escolha Grupos de segurança e escolha o grupo de segurança associado a esse endpoint.
3. Escolha Regras de entrada e Editar regras de entrada.
4. Adicione uma regra de entrada com Tipo como CQLSH/CASSANDRA. Isso define o intervalo de portas automaticamente para 9142.
5. Escolha Salvar regras para salvar sua nova regra de entrada.

Para configurar permissões do usuário do IAM

1. Confirme se o usuário do IAM usado para se conectar ao Amazon Keyspaces tem as permissões apropriadas. Em AWS Identity and Access Management (IAM), você pode usar a política AWS gerenciada `AmazonKeyspacesReadOnlyAccess` para conceder ao usuário do IAM acesso de leitura ao Amazon Keyspaces.
   a. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
   b. No painel do console do IAM, escolha Users (Usuários) e, em seguida, escolha seu usuário do IAM na lista.
   c. Na página Summary (Resumo), escolha Add permissions (Adicionar permissões).
   d. Escolha Attach existing policies directly (Anexar políticas existentes diretamente).
   e. Na lista de políticas, escolha AmazonKeyspacesReadOnlyAccess e, em seguida, escolha Next: Review.
   f. Escolha Add permissions (Adicionar permissões).
2. Verifique se você consegue acessar o Amazon Keyspaces por meio do endpoint da VPC.

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

Se quiser, você pode tentar alguns outros AWS CLI comandos para o Amazon Keyspaces. Para obter mais informações, consulte Referência de comandos da AWS CLI.

> **Note**
>
> As permissões mínimas exigidas para que um usuário ou perfil do IAM acesse o Amazon Keyspaces são permissões de leitura para a tabela do sistema, conforme mostrado na política a seguir. Para obter informações sobre permissões baseadas em políticas, consulte the section called “Exemplos de políticas baseadas em identidade”.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:555555555555:/keyspace/system*"
         ]
      }
   ]
}
```

3. Conceda ao usuário do IAM acesso de leitura à instância do Amazon EC2 com a VPC.

   Ao usar o Amazon Keyspaces com endpoints da VPC, você precisa conceder ao usuário ou perfil do IAM que acessa o Amazon Keyspaces permissões somente de leitura para sua instância do Amazon EC2 e para a VPC, para coletar dados de endpoints e interfaces de rede. O Amazon Keyspaces armazena essas informações na tabela `system.peers` e as usa para gerenciar conexões.

> **Note**
>
> As políticas gerenciadas `AmazonKeyspacesReadOnlyAccess_v2` e `AmazonKeyspacesFullAccess` incluem as permissões necessárias para permitir que o Amazon Keyspaces acesse a instância do Amazon EC2 para ler informações sobre os endpoints da VPC de interface disponíveis.

a. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.

b. No painel do console do IAM, escolha Políticas.

c. Escolha Criar política e escolha a guia JSON.

d. Copie a política a seguir e escolha Próximo: Tags.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource": "*"
      }
   ]
}
```

e. Escolha Próximo: Revisar, insira um nome `keyspacesVPCendpoint` para a política e escolha Criar política.

f. No painel do console do IAM, escolha Users (Usuários) e, em seguida, escolha seu usuário do IAM na lista.

g. Na página Summary (Resumo), escolha Add permissions (Adicionar permissões).

h. Escolha Attach existing policies directly (Anexar políticas existentes diretamente).

i. Na lista de políticas, escolha keyspacesVPCendpoint, e Próximo: Revisar.

j. Escolha Add permissions (Adicionar permissões).

4. Para verificar se a tabela `system.peers` do Amazon Keyspaces está sendo atualizada com as informações da VPC, execute a seguinte consulta na sua instância do Amazon EC2 usando `cqlsh`. Se você ainda não tiver instalado `cqlsh` em sua instância do Amazon EC2 na etapa 2, siga as instruções em the section called “Usar a `cqlsh-expansion`”.

```
SELECT peer FROM system.peers;
```

A saída retorna nós com endereços IP privados, dependendo da configuração da VPC e da sub-rede na sua região. AWS

```
peer
---------------
112.11.22.123
112.11.22.124
112.11.22.125
```

> **Note**
>
> Você precisa usar uma conexão `cqlsh` com o Amazon Keyspaces para confirmar que seu endpoint da VPC foi configurado corretamente. Se você usa seu ambiente local ou o editor Amazon Keyspaces CQL no AWS Management Console, a conexão passa automaticamente pelo endpoint público em vez do seu endpoint da VPC. Se houver nove endereços IP, essas são as entradas que o Amazon Keyspaces grava automaticamente na tabela `system.peers` em conexões públicas de endpoints.

## Etapa 5: Configurar o monitoramento com CloudWatch

Esta etapa mostra como usar a Amazon para monitorar CloudWatch a conexão do VPC endpoint com o Amazon Keyspaces.

AWS PrivateLink publica pontos de dados CloudWatch sobre seus endpoints de interface. Você pode usar métricas para verificar se o sistema está executando conforme o esperado. O `AWS/PrivateLinkEndpoints` namespace em CloudWatch inclui as métricas dos endpoints da interface. Para obter mais informações, consulte CloudWatch as AWS PrivateLink métricas do AWS PrivateLink Guia.

Para criar um CloudWatch painel com métricas de VPC endpoint

1. Abra o CloudWatch console emhttps://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Painéis. Em seguida, escolha Criar painel. Insira um nome para o painel e escolha Criar.
3. Em Adicionar widget, escolha Número.
4. Em Métricas, escolha AWS/Endpoints PrivateLink.
5. Escolha Tipo de endpoint, Nome do serviço, ID do endpoint da VPC, ID da VPC.
6. Selecione as métricas `ActiveConnections` e `NewConnections`, e escolha Criar widget.
7. Salve o painel.

A métrica `ActiveConnections` é definida como o número de conexões ativas simultâneas que o endpoint recebeu durante o último período de um minuto. A métrica `NewConnections` é definida como o número de novas conexões estabelecidas pelo endpoint durante o último período de um minuto.

Para obter mais informações sobre a criação de painéis, consulte Criar painel no Guia do CloudWatch usuário.

## Etapa 6: (opcional) práticas recomendadas para configurar o tamanho do pool de conexões para seu aplicativo

Nesta seção, descrevemos como determinar o tamanho ideal do pool de conexões com base nos requisitos de throughput de consultas do seu aplicativo.

O Amazon Keyspaces permite um máximo de 3.000 consultas CQL por segundo por conexão TCP. Portanto, praticamente não há limite no número de conexões que um driver pode estabelecer com o Amazon Keyspaces. No entanto, recomendamos que você combine o tamanho do pool de conexões com os requisitos do seu aplicativo e considere os endpoints disponíveis ao usar o Amazon Keyspaces com conexões de endpoints da VPC.

Você configura o tamanho do pool de conexões no driver do cliente. Por exemplo, com base em um pool local de 2 e em um endpoint da VPC de interface criado em 3 zonas de disponibilidade, o driver estabelece 6 conexões para consulta (7 no total, o que inclui uma conexão de controle). Usando essas 6 conexões, você pode suportar no máximo 18.000 consultas CQL por segundo.

Se seu aplicativo precisar oferecer suporte a 40.000 consultas CQL por segundo, trabalhe retroativamente a partir do número de consultas necessárias para determinar o tamanho do pool de conexões necessário. Para oferecer suporte a 40.000 consultas CQL por segundo, você precisa configurar o tamanho do pool local para pelo menos 5, o que dá suporte a um mínimo de 45.000 consultas CQL por segundo.

Você pode monitorar se excede a cota do número máximo de operações por segundo, por conexão, usando a `PerConnectionRequestRateExceeded` CloudWatch métrica no `AWS/Cassandra` namespace. A métrica `PerConnectionRequestRateExceeded` mostra o número de solicitações para o Amazon Keyspaces que excedem a cota da taxa de solicitações por conexão.

Os exemplos de código nesta etapa mostram como estimar e configurar o pool de conexões quando você usa endpoints da VPC de interface.

Java

O número de conexões por pool é configurável no driver Java. Para obter um exemplo completo de uma conexão de driver de cliente Java, consulte the section called “Uso de um driver de cliente Java Cassandra”.

Quando o driver do cliente é iniciado, primeiro a conexão de controle é estabelecida para tarefas administrativas, como alterações de esquema e topologia. Em seguida, as conexões adicionais são criadas.

No exemplo a seguir, a configuração do driver do tamanho do pool local é especificada como 2. Se o endpoint da VPC for criado em 3 sub-redes dentro da VPC, isso resultará em `NewConnections` 7 entradas para o endpoint da interface, CloudWatch conforme mostrado na fórmula a seguir.

```
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
          }
}
```

Se o número de conexões ativas não corresponder ao tamanho do pool configurado (agregação entre sub-redes) + 1 conexão de controle, algo está impedindo que as conexões sejam criadas.

Node.js

Você pode configurar o número de conexões por pool no driver Node.js. Para obter um exemplo completo de uma conexão de driver de cliente Node.js, consulte the section called “Uso de um driver de cliente Cassandra Node.js”.

No exemplo de código a seguir, a configuração do driver do tamanho do pool local é especificada como 1. Se o endpoint da VPC for criado em 4 sub-redes dentro da VPC, isso resultará em `NewConnections` 5 entradas para o endpoint da interface, CloudWatch conforme mostrado na fórmula a seguir.

```
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 }
        });
```

## Etapa 7: (opcional) limpeza

Para excluir os recursos que você criou neste tutorial, siga estes procedimentos.

Para remover seu endpoint da VPC para o Amazon Keyspaces

1. Faça login em sua instância do Amazon EC2.
2. Determine o ID do endpoint da VPC que é usado para o Amazon Keyspaces. Se você omitir os parâmetros `grep`, as informações do endpoint da VPC serão mostradas para todos os serviços.

```
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"
    }
}
```

   No exemplo de saída, o ID do VPC endpoint é `vpce-9b15e2f2`.
3. Exclua o endpoint da VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-9b15e2f2

{
    "Unsuccessful": []
}
```

A matriz vazia `[]` indica o sucesso (não há solicitações malsucedidas).

Para terminar sua instância do Amazon EC2

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. No painel de navegação, escolha Instâncias.
3. Escolha a instância do Amazon EC2.
4. Em Ações, escolha Estado da instância e Encerrar.
5. Na janela de confirmação, clique em Sim, encerrar.

# Como configurar o acesso entre contas para o Amazon Keyspaces

Você pode criar e usar separadamente Contas da AWS para isolar recursos e para uso em ambientes diferentes, por exemplo, desenvolvimento e produção. Este tópico explica o acesso entre contas ao Amazon Keyspaces usando endpoints da VPC de interface em um Amazon Virtual Private Cloud. Para obter mais informações sobre a configuração de acesso entre contas do IAM, consulte Exemplo de cenário usando contas de desenvolvimento e produção separadas no Guia do usuário do IAM.

Para obter mais informações sobre Amazon Keyspaces e endpoints privados da VPC, consulte the section called “Usar VPC endpoints da interface do ”.

Tópicos

- Como configurar o acesso entre contas para o Amazon Keyspaces em uma VPC compartilhada
- Como configurar o acesso entre contas para o Amazon Keyspaces sem uma VPC compartilhada

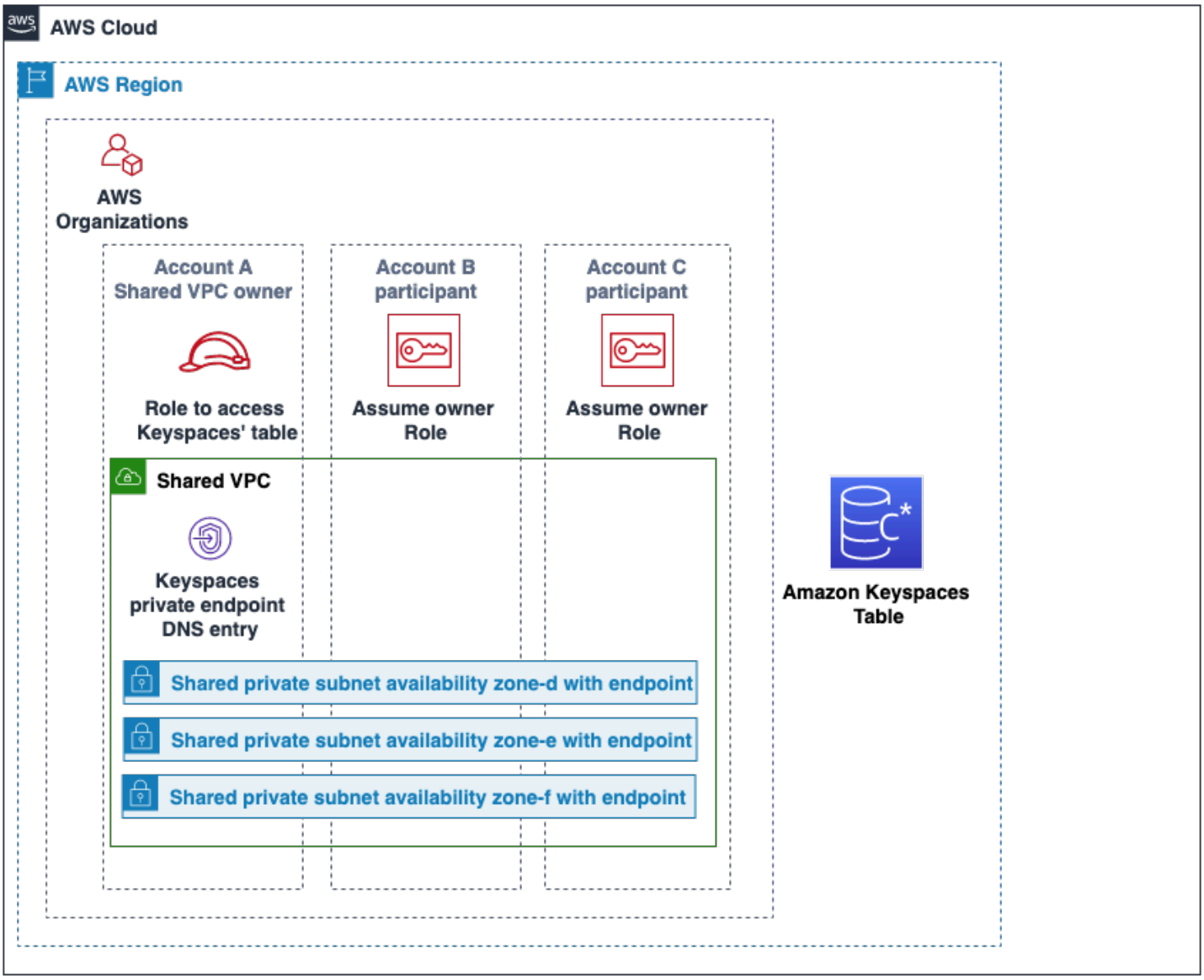
# Como configurar o acesso entre contas para o Amazon Keyspaces em uma VPC compartilhada

Você pode criar Contas da AWS diferentes para separar os recursos dos aplicativos. Por exemplo, você pode criar uma conta para suas tabelas do Amazon Keyspaces, uma conta diferente para aplicativos em um ambiente de desenvolvimento e outra conta para aplicativos em um ambiente de produção. Este tópico mostra as etapas de configuração necessárias para configurar o acesso entre contas para o Amazon Keyspaces usando endpoints de VPC de interface em uma VPC compartilhada.

Para obter etapas detalhadas sobre como configurar um endpoint da VPC para o Amazon Keyspaces, consulte the section called “Etapa 3: criar um endpoint da VPC para o Amazon Keyspaces”.

Neste exemplo, usamos as três contas a seguir em uma VPC compartilhada:

- `Account A`: essa conta contém infraestrutura, incluindo os endpoints da VPC, as sub-redes da VPC e as tabelas do Amazon Keyspaces.
- `Account B`: essa conta contém um aplicativo em um ambiente de desenvolvimento que precisa se conectar à tabela do Amazon Keyspaces em `Account A`.
- `Account C`: essa conta contém um aplicativo em um ambiente de produção que precisa se conectar à tabela do Amazon Keyspaces em `Account A`.




`Account A` é a conta que contém os recursos que `Account B` e `Account C` precisam para o acesso, assim `Account A` é a conta confiável. `Account B` e `Account C` são as contas com as entidades principais que precisam acessar os recursos em `Account A`, portanto, `Account B` e `Account C` são as contas confiáveis. A conta confiável concede as permissões às contas confiáveis compartilhando um perfil do IAM. O procedimento a seguir descreve as etapas de configuração necessárias em `Account A`.

Configuração para a **`Account A`**

1. Use AWS Resource Access Manager para criar um compartilhamento de recursos para a sub-rede e compartilhar a sub-rede privada com `Account B` e `Account C`.

`Account B` e `Account C` agora podem ver e criar recursos na sub-rede que foi compartilhada com eles.

2. Crie um endpoint da VPC privado do Amazon Keyspaces desenvolvido por AWS PrivateLink. Isso cria vários endpoints em sub-redes compartilhadas e entradas de DNS para o endpoint do serviço do Amazon Keyspaces.
3. Crie um espaço de chaves e uma tabela do Amazon Keyspaces.
4. Crie um perfil do IAM que tenha acesso total à tabela do Amazon Keyspaces, leia as tabelas do sistema Amazon Keyspaces e seja capaz de descrever os recursos da VPC do Amazon EC2, conforme mostrado no exemplo de política a seguir.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountAccess",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcEndpoints",
                "cassandra:*"
            ],
            "Resource": "*"
        }
    ]
}
```

5. Configure a política de confiança do perfil do IAM que `Account B` e `Account C` pode assumir como contas confiáveis, conforme mostrado no exemplo a seguir.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111111111111:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {}
        }
```

```
    ]
}
```

Para ver mais informações sobre as políticas entre contas do IAM, consulte Políticas entre contas no Guia do usuário do IAM.

Configuração em **`Account B`** e **`Account C`**

1. Em `Account B` e `Account C`, crie novos perfis e anexe a seguinte política que permite que a entidade principal assuma o perfil compartilhado criado em `Account A`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

   Permitir que a entidade principal assuma o perfil compartilhado é implementado usando a `AssumeRole` API do AWS Security Token Service (AWS STS). Para obter mais informações, consulte Fornecer acesso a um usuário do IAM em outra Conta da AWS de sua propriedade.

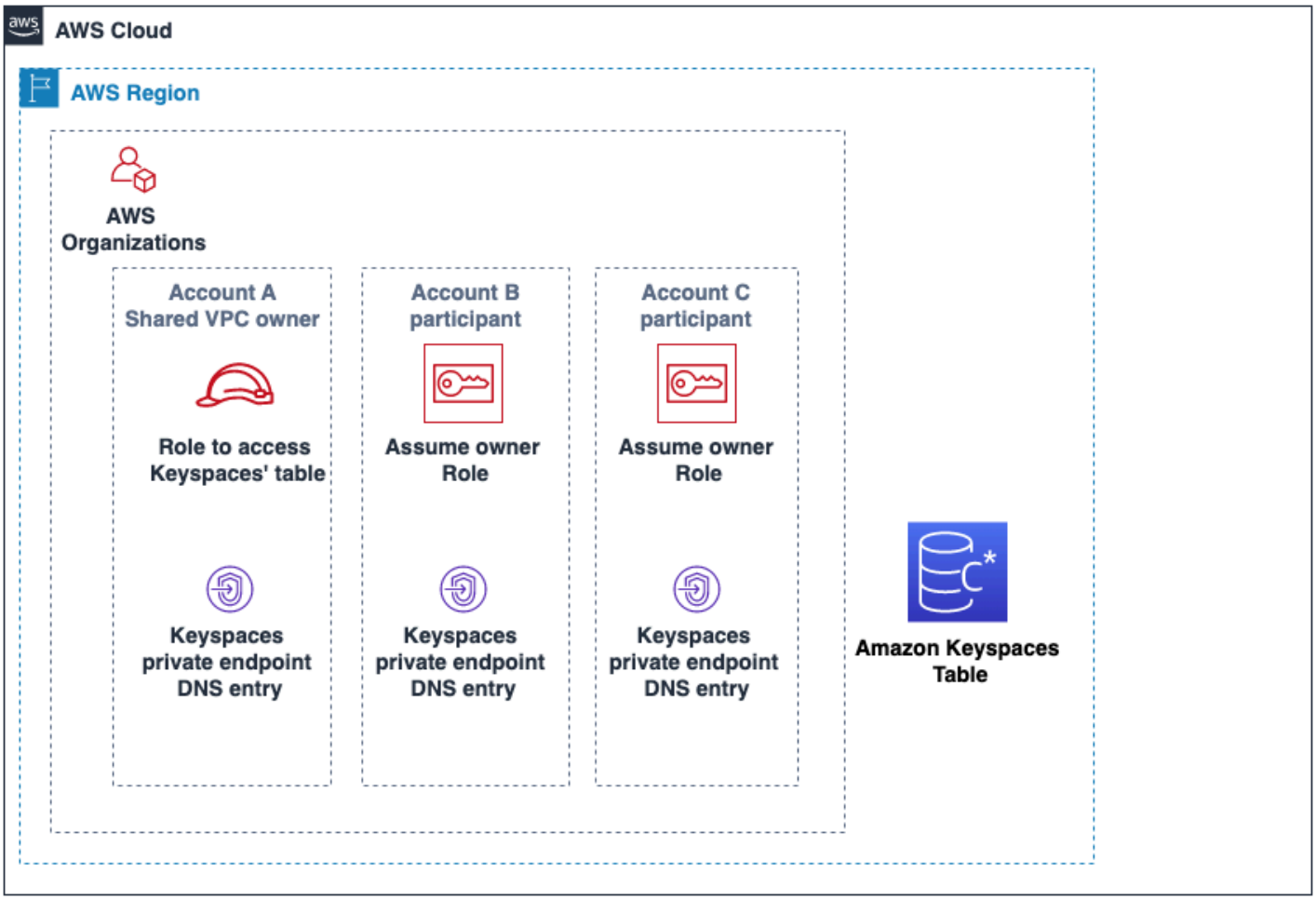
2. Em `Account B` e `Account C`, você pode criar aplicativos que utilizam o plug-in de autenticação SIGV4, que permite que um aplicativo assuma o perfil compartilhado de se conectar à tabela do Amazon Keyspaces localizada em `Account A` por meio do endpoint da VPC na VPC compartilhada. Para obter mais informações sobre o plug-in de autenticação SIGV4, consulte the section called “Criação de credenciais”.

## Como configurar o acesso entre contas para o Amazon Keyspaces sem uma VPC compartilhada

Se a tabela do Amazon Keyspaces e o endpoint privado da VPC pertencerem a contas diferentes, mas não estiverem compartilhando uma VPC, os aplicativos ainda poderão se conectar entre

contas usando endpoints da VPC. Como as contas não estão compartilhando os endpoints da VPC, `Account A`, `Account B` e `Account C` exigem seus próprios endpoints da VPC. Para o driver do cliente Cassandra, o Amazon Keyspaces aparece como um único nó em vez de um cluster de vários nós. Após a conexão, o driver do cliente chega ao servidor DNS, que retorna um dos endpoints disponíveis na VPC da conta.

Você também pode acessar tabelas do Amazon Keyspaces em contas diferentes sem um endpoint da VPC compartilhado usando o endpoint público ou implantando um endpoint da VPC privado em cada conta. Quando não está usando uma VPC compartilhada, cada conta exige seu próprio endpoint da VPC. Neste exemplo, `Account A`, `Account B` e `Account C` exigem seus próprios endpoints da VPC para acessar a tabela em `Account A`. Ao usar endpoints da VPC nessa configuração, o Amazon Keyspaces aparece como um cluster de nó único para o driver do cliente Cassandra em vez de um cluster de vários nós. Após a conexão, o driver do cliente chega ao servidor DNS, que retorna um dos endpoints disponíveis na VPC da conta. Mas o driver do cliente não consegue acessar a tabela `system.peers` para descobrir endpoints adicionais. Como há menos hosts disponíveis, o driver faz menos conexões. Para ajustar isso, aumente a configuração do pool de conexões do driver em um fator de três.


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

# Conceitos básicos do Amazon Keyspaces (para Apache Cassandra)

Este tutorial é para você que não conhece o Apache Cassandra e o Amazon Keyspaces (para Apache Cassandra). Neste tutorial, você instala todos os programas e drivers necessários para usar com sucesso o Amazon Keyspaces.

Para obter tutoriais sobre como se conectar programaticamente ao Amazon Keyspaces usando diferentes drivers de cliente do Cassandra, consulte the section called “Uso de um driver de cliente Cassandra”.

Tópicos


- Pré-requisitos e considerações do tutorial
- Etapa 1 do tutorial: criar um espaço de chave e uma tabela no Amazon Keyspaces
- Etapa 2 do tutorial: criar, ler, atualizar e excluir dados (CRUD)
- Etapa 3 do tutorial: excluir uma tabela e um espaço de chave no Amazon Keyspaces


## Pré-requisitos e considerações do tutorial

Antes de começar este tutorial, siga as instruções AWS de configuração emComo acessar o Amazon Keyspaces (para Apache Cassandra). Essas etapas incluem inscrever-se AWS e criar um usuário AWS Identity and Access Management (IAM) com acesso ao Amazon Keyspaces.

Além disso, se você estiver realizando o tutorial usando um `cqlsh` ou um driver de cliente Cassandra Apache 2.0 licenciado, conclua as instruções de configuração em Usar `cqlsh` para se conectar ao Amazon Keyspaces.

Depois de concluir as etapas de pré-requisito, vá para Etapa 1 do tutorial: criar um espaço de chave e uma tabela no Amazon Keyspaces.

## Etapa 1 do tutorial: criar um espaço de chave e uma tabela no Amazon Keyspaces

Nesta seção, você cria um espaço de chave e adiciona uma tabela a ele usando o console.

> **Note**
>
> Antes de começar, certifique-se de que você configurou todos os pré-requisitos do tutorial.

Tópicos


- Como criar um espaço de chaves
- Criar uma tabela


## Como criar um espaço de chaves

Um espaço de chave agrupa tabelas relacionadas que são relevantes para um ou mais aplicativos. Um espaço de chave contém uma ou mais tabelas e define a estratégia de replicação para todas as tabelas que ele contém. Para obter mais informações sobre espaços de chaves, consulte os seguintes tópicos:

- Como trabalhar com espaços de chaves: the section called “Como criar espaços de chaves”
- Instruções Data Definition Language (DDL): Keyspaces
- Cotas para Amazon Keyspaces (para Apache Cassandra)

Ao criar um espaço de chave, você deve especificar o nome do espaço de chave.

> **Note**
>
> A estratégia de replicação do espaço de chave deve ser `SingleRegionStrategy`. O `SingleRegionStrategy` replica dados em três zonas de disponibilidade na sua Região da AWS.

Usar o console

Para criar um espaço de chave usando o console

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, escolha Keyspaces.

3. Selecione Criar espaço de chaves.
4. Na caixa Nome do keyspace, insira **`myGSGKeyspace`** como o nome do espaço de chaves.

   Restrições de nome:

   - Não pode estar vazio.
   - Caracteres permitidos: caracteres alfanuméricos e sublinhado ( _ ).
   - O tamanho máximo é de 48 caracteres.
5. Para criar o espaço de chaves, escolha Criar espaço de chaves.
6. Verifique se o espaço de chave `myGSGKeyspace` foi criado fazendo o seguinte:

   a. No painel de navegação, escolha Keyspaces.

   b. Localize seu espaço de chave `myGSGKeyspace` na lista de espaços de chave.

Usar SSL

O procedimento a seguir cria um espaço de chave usando CQL.

Para criar um espaço de chave usando CQL

1. Abra um shell de comando e digite o seguinte:

   **`cqlsh`**
2. Crie seu espaço de chave usando o seguinte comando CQL.

   ```
   CREATE KEYSPACE IF NOT EXISTS "myGSGKeyspace"
      WITH REPLICATION = {'class': 'SingleRegionStrategy'};
   ```

   `SingleRegionStrategy`usa um fator de replicação de três e replica dados em três zonas de AWS disponibilidade em sua região.

   > **Note**
   >
   > O Amazon Keyspaces usa como padrão todas as entradas em minúsculas, a menos que você as coloque entre aspas. Nesse caso, observe `"myGSGKeyspace"`.
3. Verifique se seu espaço de chave foi criado.

```
SELECT * from system_schema.keyspaces ;
```

O espaço de chave deve ser listado.

## Criar uma tabela

Uma tabela é onde seus dados são organizados e armazenados. A chave primária da tabela determina como os dados serão particionados na tabela. A chave primária é composta por uma chave de partição necessária e uma ou mais colunas de agrupamento opcionais. Os valores combinados que compõem a chave primária devem ser exclusivos em todos os dados da tabela. Para obter mais informações sobre tabelas, consulte os tópicos a seguir:

- Trabalhar com tabelas: the section called “Criar tabelas”
- Instruções DDL: Tabelas
- Gerenciamento de recursos da tabela: Gerenciamento de recursos de tecnologia sem servidor
- Monitoramento da utilização dos recursos da tabela: the section called “Monitoramento com CloudWatch”
- Cotas para Amazon Keyspaces (para Apache Cassandra)

Ao criar uma tabela, especifique o seguinte:

- O nome da tabela.
- O nome e o tipo de dados de cada coluna em uma tabela.
- A chave primária da tabela.
  - Chave de partição: obrigatória
  - Colunas de clustering: opcional

Use o procedimento a seguir para criar uma tabela com as colunas, os tipos de dados, a chave de partição e a coluna de clustering especificados.

Como usar o console

O procedimento a seguir cria a tabela `employees_tbl` com essas colunas e tipos de dados.

```
ID           text
```

```
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Para criar uma tabela usando o console

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, escolha Keyspaces.
3. Escolha `myGSGKeyspace` como o espaço de chave em que você deseja criar essa tabela.
4. Escolha Create table.
5. Na caixa Nome da tabela, insira **`employees_tbl`** como nome da tabela.

   Restrições de nome:

   - Não pode estar vazio.
   - Caracteres permitidos: caracteres alfanuméricos e sublinhado ( _ ).
   - O tamanho máximo é de 48 caracteres.
6. Na seção Colunas, repita as etapas a seguir para cada coluna que você deseja adicionar a essa tabela.

   Adicione as colunas e os tipos de dados a seguir.

   ```
   id            text
   name          text
   region        text
   division      text
   project       text
   role          text
   pay_scale     int
   vacation_hrs  float
   manager_id    text
   ```

   a. Nome: insira o nome da coluna.

Restrições de nome:

- Não pode estar vazio.
- Caracteres permitidos: caracteres alfanuméricos e sublinhado ( _ ).
- O tamanho máximo é de 48 caracteres.

b. Tipo: na lista de tipos de dados, escolha o tipo de dados para essa coluna.

c. Se quiser adicionar outra coluna, escolha Adicionar coluna.

7. Escolha `id` como chave de partição em Chave de partição. É necessária uma chave de partição para cada tabela. A chave de partição pode ser composta por uma ou mais colunas.
8. Adicione `division` como uma coluna de clustering. As colunas de clustering são opcionais e determinam a ordem de classificação em cada partição.

   a. Para adicionar uma coluna de clustering, escolha Adicionar coluna de clustering.

   b. Na lista Colunas, escolha divisão. Na lista Ordem, escolha ASC para classificar em ordem crescente nos valores dessa coluna. (Escolha DESC para ordem decrescente.)
9. Na seção Configurações da tabela, escolha Configurações padrão.
10. Escolha Create table.
11. Verifique se sua tabela foi criada.

    a. No painel de navegação, selecione Tabelas.

    b. Confirme se sua tabela está na lista de tabelas.

    c. Escolha o nome da sua tabela.

    d. Confirme se todas as suas colunas e tipos de dados estão corretos.

    > **Note**
    >
    > As colunas podem não estar listadas na mesma ordem em que você as adicionou à tabela.

    e. Na coluna de clustering, confirme se a divisão está identificada como verdadeira. Todas as outras colunas da tabela devem ser falsas.

Usar SSL

O procedimento a seguir cria uma tabela com as seguintes colunas e tipos de dados usando CQL. A coluna `id` é a chave de partição.

```
id            text
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Para criar uma tabela usando CQL

1. Abra um shell de comando e digite o seguinte:

   **`cqlsh`**

2. No `cqlsh` prompt (`cqlsh>`), especifique um espaço de chave para criar sua tabela.

   ```
   USE "myGSGKeyspace" ;
   ```

3. No prompt do espaço de chave (`cqlsh:keyspace_name>`), crie sua tabela inserindo o código a seguir na janela de comando.

   ```
   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` é a ordem de clustering padrão. Você também pode especificar `DESC` para ordem decrescente.

Observe que a coluna `id` deve ser a chave de partição. Em seguida, a coluna de clustering `division` é ordenada em ordem crescente (`ASC`).

4. Verifique se sua tabela foi criada.

```
SELECT * from system_schema.tables WHERE keyspace_name='myGSGKeyspace' ;
```

A tabela deve estar listada.

5. Verifique a estrutura da sua tabela.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'myGSGKeyspace' AND
 table_name = 'employees_tbl' ;
```

Confirme se todas as colunas e tipos de dados estão conforme o esperado. A ordem das colunas pode ser diferente da instrução `CREATE`.

Para realizar operações CRUD (criar, ler, atualizar e excluir) nos dados da tabela, vá para the section called “Etapa 2: Operações de CRUD”.

# Etapa 2 do tutorial: criar, ler, atualizar e excluir dados (CRUD)

Nesta seção, você usa o editor CQL no console para executar operações CRUD (criar, ler, atualizar e excluir) nos dados em sua tabela. Você também pode executar os comandos usando `cqlsh`.

Tópicos

- Tutorial: inserção e carregamento de dados em uma tabela do Amazon Keyspaces
- Tutorial: Leia de uma tabela do Amazon Keyspaces
- Tutorial: atualizar dados em uma tabela do Amazon Keyspaces
- Tutorial: Excluir dados em uma tabela do Amazon Keyspaces

# Tutorial: inserção e carregamento de dados em uma tabela do Amazon Keyspaces

Para criar dados em sua tabela `employees_tbl`, use a instrução `INSERT` para adicionar uma única linha.

1. Antes de poder gravar dados em sua tabela do Amazon Keyspaces usando cqlsh, você deve definir a consistência de gravação da sessão atual de cqlsh como `LOCAL_QUORUM`. Para obter mais informações sobre os níveis de consistência suportados, consulte the section called “Níveis de consistência de gravação”. Observe que essa etapa não é necessária se você estiver usando o editor CQL no AWS Management Console.

   ```
   CONSISTENCY LOCAL_QUORUM;
   ```

2. Para inserir um único registro, execute o comando a seguir no editor 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. Verifique se os dados foram adicionados corretamente à tabela executando o comando a seguir.

   ```
   SELECT * FROM "myGSGKeyspace".employees_tbl ;
   ```

Para inserir vários registros de um arquivo usando cqlsh

1. Faça o download do arquivo de dados de amostra (employees.csv) contido no arquivo sampledata.zip. Esse arquivo CSV (valores separados por vírgula) contém os seguintes dados. Lembre-se do caminho em que você salvou o arquivo.

| 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. Abra um shell de comando e digite o seguinte:

**cqlsh**

3. No prompt de `cqlsh` (`cqlsh>`), especifique um espaço de chave.

```
USE "myGSGKeyspace" ;
```

4. Defina a consistência de gravação como `LOCAL_QUORUM`. Para obter mais informações sobre os níveis de consistência suportados, consulte the section called “Níveis de consistência de gravação”.

```
CONSISTENCY LOCAL_QUORUM;
```

5. No prompt do espaço de chave (`cqlsh:keyspace_name>`), execute a consulta a seguir.

```
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. Verifique se os dados foram adicionados corretamente à tabela executando a consulta a seguir.

```
SELECT * FROM employees_tbl ;
```

## Tutorial: Leia de uma tabela do Amazon Keyspaces

Na seção Tutorial: inserção e carregamento de dados em uma tabela do Amazon Keyspaces, você usou a instrução `SELECT` para verificar se os dados foram adicionados com sucesso à sua tabela. Nesta seção, você refina o uso de `SELECT` para exibir colunas específicas e somente linhas que atendam a critérios específicos.

A forma geral da instrução `SELECT` é a seguinte.

```
SELECT column_list FROM table_name [WHERE condition [ALLOW FILTERING]] ;
```

Tópicos

- Como selecionar todos os dados da tabela
- Como selecionar um subconjunto de colunas
- Como selecionar um subconjunto de linhas

## Como selecionar todos os dados da tabela

A forma mais simples da instrução `SELECT` retorna todos os dados da tabela.

> **Important**
>
> Em um ambiente de produção, normalmente não é uma prática recomendada executar esse comando, que retorna todos os dados da tabela.

Para selecionar todos os dados da tabela

- Execute a seguinte consulta .

  ```
  SELECT * FROM "myGSGKeyspace".employees_tbl ;
  ```

  Usar o caractere curinga (`*`) para o `column_list` selecionar todas as colunas.

## Como selecionar um subconjunto de colunas

Para consultar um subconjunto de colunas

- Para recuperar apenas as colunas `id`, `name` e `manager_id`, execute a seguinte consulta.

  ```
  SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
  ```

  A saída conterá somente as colunas especificadas na ordem listada na instrução `SELECT`.

## Como selecionar um subconjunto de linhas

Ao consultar um grande conjunto de dados, talvez você queira apenas registros que atendam a determinados critérios. Para fazer isso, você pode acrescentar uma cláusula `WHERE` ao final da instrução `SELECT`.

Para consultar um subconjunto de linhas

- Para recuperar somente o registro do funcionário com o id `'234-56-7890'`, execute a consulta a seguir.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='234-56-7890' ;
```

A instrução SELECT anterior retorna somente as linhas em que o id é 234-56-7890.

### Entendendo a cláusula **WHERE**

A cláusula WHERE é usada para filtrar os dados e retornar somente os dados que atendem aos critérios especificados. Os critérios especificados podem ser uma condição simples ou uma condição composta.

#### Como usar as condições em uma cláusula **WHERE**

- Uma condição simples, uma única coluna.

  ```
  WHERE column_name=value
  ```

  Você poderá usar uma condição simples em uma cláusula WHERE se uma das seguintes condições for atendida:

  - A coluna é a única coluna na chave primária da tabela.
  - Você adiciona ALLOW FILTERING após a condição na cláusula WHERE.

    Esteja ciente de que usar ALLOW FILTERING pode resultar em desempenho inconsistente, especialmente em tabelas grandes e multiparticionadas.
- Uma condição composta, várias condições simples conectadas por AND.

  ```
  WHERE column_name1=value1 AND column_name2=value2 AND column_name3=value3...
  ```

  É possível usar condições compostas em uma cláusula WHERE se uma das condições a seguir for atendida:

  - As colunas na cláusula WHERE correspondem exatamente às colunas na chave primária da tabela, nem mais nem menos.
  - Você adiciona ALLOW FILTERING após a condição composta na cláusula WHERE, como no exemplo a seguir.

    ```
    SELECT * FROM my_table WHERE col1=5 AND col2='Bob' ALLOW FILTERING ;
    ```

Esteja ciente de que usar `ALLOW FILTERING` pode resultar em desempenho inconsistente, especialmente em tabelas grandes e multiparticionadas.

Experimente

Crie suas próprias consultas em CQL para encontrar o seguinte em sua tabela `employees_tbl`:

- Encontre o `name`, `project` e `id` de todos os funcionários.
- Descubra em qual projeto `Bob` o estagiário está trabalhando (inclua pelo menos seu nome, projeto e perfil no resultado).
- Avançado: crie um aplicativo para encontrar todos os funcionários que têm o mesmo gerente como `Bob` o estagiário. DICA: Isso pode exigir mais de uma consulta.
- Avançado: crie um aplicativo para encontrar colunas selecionadas de todos os funcionários que trabalham no projeto `NightFlight`. DICA: Resolver isso pode exigir várias instruções.

## Tutorial: atualizar dados em uma tabela do Amazon Keyspaces

Para atualizar os dados em sua tabela `employees_tbl`, use a instrução `UPDATE`.

A forma geral da instrução `UPDATE` é a seguinte.

```
UPDATE table_name SET column_name=new_value WHERE primary_key=value ;
```

**Tip**

- É possível atualizar várias colunas usando uma lista separada por vírgula de `column_names` e valores, como no seguinte exemplo.

  ```
  UPDATE my_table SET col1='new_value_1', col2='new_value2' WHERE id='12345' ;
  ```

- Se a chave primária for composta por várias colunas, todas as colunas da chave primária e seus valores deverão ser incluídos na cláusula `WHERE`.
- Você não pode atualizar nenhuma coluna na chave primária porque isso alteraria a chave primária do registro.

Para atualizar uma única célula

Usando sua tabela `employees_tbl`, dê um aumento ao funcionário do id `567-89-0123`.

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale=5 WHERE id='567-89-0123' AND
 division='Marketing' ;
```

Verifique se a escala salarial do funcionário agora é `5`.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='567-89-0123' ;
```

### Experimente

Avançado: sua empresa contratou Bob, o estagiário. Altere seu registro para que sua função seja `'IC'` e sua escala salarial seja `2`.

## Tutorial: Excluir dados em uma tabela do Amazon Keyspaces

Para excluir dados na tabela `employees_tbl`, use a instrução `DELETE`.

Você pode excluir dados de uma linha ou de uma partição. Tenha cuidado ao excluir dados, pois as exclusões são irreversíveis.

Excluir uma ou todas as linhas de uma tabela não exclui a tabela. Assim, você pode preenchê-la novamente com dados. A exclusão de uma tabela exclui a tabela e todos os dados nela contidos. Para usar a tabela novamente, você deverá recriá-la e adicionar dados a ela. A exclusão de um espaço de chave exclui o espaço de chave e todas as tabelas dentro dele. Para usar o espaço de chave e as tabelas, você deve recriá-los e, em seguida, preenchê-los com dados.

### Como excluir células

A exclusão de uma coluna de uma linha remove os dados da célula especificada. Quando você exibe essa coluna usando uma instrução `SELECT`, os dados são exibidos como *`nulos`*, embora um valor nulo não seja armazenado nesse local.

A sintaxe geral para excluir uma ou mais colunas específicas é a seguinte.

```
DELETE column_name1[, column_name2...] FROM table_name WHERE condition ;
```

Na sua tabela `employees_tbl`, você pode ver que o CEO tem `"None"` como gerente. Primeiro, exclua essa célula para que você não carregue nenhum dado nela.

Como excluir uma célula específica

1. Execute a seguinte consulta DELETE.

```
DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive';
```

2. Verifique se a exclusão foi feita conforme o esperado.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive';
```

### Com excluir linhas

Pode haver um momento em que você precise excluir uma linha inteira, como quando um funcionário se aposenta. A sintaxe geral para excluir uma linha é a seguinte.

```
DELETE FROM table_name WHERE condition ;
```

Para excluir uma linha

1. Execute a seguinte consulta DELETE.

```
DELETE FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
 division='Engineering';
```

2. Verifique se a exclusão foi feita conforme o esperado.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
 division='Engineering';
```

## Etapa 3 do tutorial: excluir uma tabela e um espaço de chave no Amazon Keyspaces

Para evitar a cobrança por tabelas e dados desnecessários, exclua todas as tabelas e espaços de chave que não estiver usando. Quando você exclui uma tabela, a tabela e seus dados são excluídos e você deixa de acumular cobranças por eles. No entanto, o espaço chave permanece. Quando você

exclui um espaço de chave, o espaço de chave e todas as suas tabelas são excluídos e você deixa de acumular cobranças por eles.

## Excluir uma tabela

É possível excluir uma tabela usando o console ou CQL. Quando você exclui uma tabela, a tabela e todos os seus dados são excluídos.

Como usar o console

O procedimento a seguir exclui uma tabela e todos os seus dados que usam o AWS Management Console.

Para excluir uma tabela usando o console

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tabelas.
3. Escolha a caixa à esquerda do nome de cada tabela que deseja excluir.
4. Escolha Excluir.
5. Na tela Excluir tabela, insira **`Delete`** na caixa. Em seguida, escolha Excluir tabela.
6. Para verificar se a tabela foi excluída, escolha Tabelas no painel de navegação e confirme se a tabela `employees_tbl` não está mais listada.

Usar SSL

O procedimento a seguir exclui uma tabela e todos os seus dados usando CQL.

Para excluir uma tabela usando CQL

1. Abra um shell de comando e digite o seguinte:

   **`cqlsh`**

2. Exclua sua tabela inserindo o comando a seguir no prompt do espaço de chave (`cqlsh:`*`keyspace_name`*`>`).

```
DROP TABLE IF EXISTS "myGSGKeyspace".employees_tbl ;
```

3. Verifique se sua tabela foi excluída.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = 'myGSGKeyspace' ;
```

   Sua tabela não deve estar listada.

## Como excluir um espaço de chave

Você pode excluir um keyspace usando o AWS Management Console ou o CQL. Quando você exclui um espaço de chave, o espaço de chave e todas as suas tabelas e dados são excluídos.

Como usar o AWS Management Console

O procedimento a seguir exclui um espaço de chave e todas as suas tabelas e dados usando o AWS Management Console.

Para excluir um espaço de chave usando o console

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, escolha Keyspaces.
3. Escolha a caixa à esquerda do nome de cada espaço de chave que deseja excluir.
4. Escolha Excluir.
5. Na tela Excluir espaço de chave, insira **`Delete`** na caixa. Em seguida, escolha Excluir espaço de chave.
6. Para verificar se o espaço de chave `myGSGKeyspace` foi excluído, escolha Espaços de chave no painel de navegação e confirme se ele não está mais listado. Como você excluiu seu espaço de chave, a tabela `employees_tbl` em Tabelas também não deve estar listada.

Usar SSL

O procedimento a seguir exclui um espaço de chave e todas as suas tabelas e dados usando CQL.

Para excluir um espaço de chave usando CQL

1. Abra um shell de comando e digite o seguinte:

   **`cqlsh`**

2. Exclua o espaço de chave inserindo o comando a seguir no prompt do espaço de chave (`cqlsh:keyspace_name>`).

```
DROP KEYSPACE IF EXISTS "myGSGKeyspace" ;
```

3. Verifique se seu espaço de chave foi excluído.

```
SELECT * from system_schema.keyspaces ;
```

Seu espaço de chave não deve ser listado. Observe que, como essa é uma operação assíncrona, pode haver um atraso até que o espaço de teclas seja excluído.

# Migração para o Amazon Keyspaces

A migração para o Amazon Keyspaces (para Apache Cassandra) apresenta uma série de benefícios convincentes para empresas e organizações. Aqui estão algumas das principais vantagens que tornam o Amazon Keyspaces uma opção atraente para migração.

- Escalabilidade — O Amazon Keyspaces foi projetado para lidar com cargas de trabalho massivas e escalar perfeitamente para acomodar volumes e tráfego de dados crescentes. Com o Cassandra tradicional, o dimensionamento não é realizado sob demanda e requer planejamento para futuros picos. Com o Amazon Keyspaces, você pode escalar facilmente suas tabelas para cima ou para baixo com base na demanda, garantindo que seus aplicativos possam lidar com picos repentinos de tráfego sem comprometer o desempenho.
- Desempenho — O Amazon Keyspaces oferece acesso a dados de baixa latência, permitindo que os aplicativos recuperem e processem dados com velocidade excepcional. Sua arquitetura distribuída garante que as operações de leitura e gravação sejam distribuídas em vários nós, oferecendo tempos de resposta consistentes de um dígito de milissegundo, mesmo com altas taxas de solicitação.
- Totalmente gerenciado — o Amazon Keyspaces é um serviço totalmente gerenciado fornecido pela. AWS Isso significa que AWS lida com os aspectos operacionais do gerenciamento de banco de dados, incluindo provisionamento, configuração, aplicação de patches, backups e escalabilidade. Isso permite que você se concentre mais no desenvolvimento de aplicações e menos nas tarefas de administração do banco de dados.
- Arquitetura sem servidor — o Amazon Keyspaces não tem servidor. Você paga somente pela capacidade consumida, sem necessidade de provisionamento antecipado de capacidade. Você não tem servidores para gerenciar nem instâncias para escolher. Esse pay-per-request modelo oferece eficiência de custos e sobrecarga operacional mínima, pois você paga apenas pelos recursos que consome, sem a necessidade de provisionar e monitorar a capacidade.
- Flexibilidade do NoSQL com esquema — O Amazon Keyspaces segue um modelo de dados NoSQL, oferecendo flexibilidade no design do esquema. Com o Amazon Keyspaces, você pode armazenar dados estruturados, semiestruturados e não estruturados, tornando-os adequados para lidar com tipos de dados diversos e em evolução. Além disso, o Amazon Keyspaces realiza a validação do esquema na gravação, permitindo uma evolução centralizada do modelo de dados. Essa flexibilidade permite ciclos de desenvolvimento mais rápidos e uma adaptação mais fácil às mudanças nos requisitos de negócios.

- Alta disponibilidade e durabilidade — o Amazon Keyspaces replica dados em várias zonas de disponibilidade dentro de um Região da AWS, garantindo alta disponibilidade e durabilidade dos dados. Ele gerencia automaticamente a replicação, o failover e a recuperação, minimizando o risco de perda de dados ou interrupções no serviço. O Amazon Keyspaces fornece um SLA de disponibilidade de até 99,999%. Para obter ainda mais resiliência e leituras locais de baixa latência, o Amazon Keyspaces oferece replicação em várias regiões.
- Segurança e conformidade — o Amazon Keyspaces se integra AWS Identity and Access Management para um controle de acesso refinado. Ele fornece criptografia em repouso e em trânsito, ajudando a melhorar a segurança de seus dados. O Amazon Keyspaces também está em conformidade com padrões de segurança e leis de privacidade, incluindo HIPAA, PCI DSS e GDPR, permitindo que você atenda aos requisitos regulatórios.
- Integração com o AWS ecossistema — Como parte do AWS ecossistema, o Amazon Keyspaces se integra perfeitamente a outros Serviços da AWS, por exemplo, AWS CloudFormation Amazon e. CloudWatch AWS CloudTrail Essa integração permite criar arquiteturas sem servidor, aproveitar a infraestrutura como código e criar aplicações orientadas por dados em tempo real.

Considerações gerais sobre migrações para o Amazon Keyspaces

- Divida a migração em componentes menores.

  Considere as seguintes unidades de migração e sua presença potencial em termos de tamanho de dados brutos. A migração de quantidades menores de dados em uma ou mais fases pode ajudar a simplificar sua migração.

  - Por cluster: migre todos os seus dados do Cassandra de uma só vez. Essa abordagem pode ser adequada para clusters menores.
  - Por espaço de chaves ou tabela: divida sua migração em grupos de espaços de chaves ou tabelas. Essa abordagem pode ajudá-lo a migrar dados em fases com base nos requisitos de cada workload.
  - Por dados: considere migrar dados para um grupo específico de usuários ou produtos, para reduzir ainda mais o tamanho dos dados.
- Priorize quais dados migrar primeiro com base na simplicidade.

  Considere se você tem dados que poderiam ser migrados primeiro com mais facilidade – por exemplo, dados que não mudam em horários específicos, dados de trabalhos em lotes noturnos, dados não usados em horários off-line ou dados de aplicativos internos.

Tópicos


- Orientação para migrar dados do Apache Cassandra
- Ferramentas para migrar dados para o Amazon Keyspaces


# Orientação para migrar dados do Apache Cassandra

Para uma migração bem-sucedida do Apache Cassandra para o Amazon Keyspaces, recomendamos um planejamento cuidadoso e uma comparação das opções disponíveis. Este tópico descreve como o processo de migração funciona, quais ferramentas estão disponíveis e como você pode avaliar diferentes estratégias de migração para selecionar a que melhor atenda às suas necessidades.

Tópicos


- Compatibilidade funcional
- Estime os preços do Amazon Keyspaces
- Escolher uma estratégia de migração
- Migração off-line para o Amazon Keyspaces


## Compatibilidade funcional

Considere cuidadosamente as diferenças funcionais entre o Apache Cassandra e o Amazon Keyspaces antes da migração. O Amazon Keyspaces oferece suporte a todas as operações de plano de dados do Cassandra comumente usadas, como criar espaços de chaves e tabelas, ler dados e gravar dados. No entanto, existem algumas APIs do Cassandra que o Amazon Keyspaces não suporta. Para obter mais informações sobre as APIs compatíveis, consultethe section called “APIs, operações, funções e tipos de dados compatíveis do Cassandra”. Para uma visão geral de todas as diferenças funcionais entre o Amazon Keyspaces e o Apache Cassandra, consulte. the section called “Diferenças funcionais com o Apache Cassandra”

Para comparar as APIs e o esquema do Cassandra que você está usando com a funcionalidade compatível no Amazon Keyspaces, você pode executar um script de compatibilidade disponível no kit de ferramentas do Amazon Keyspaces em. GitHub

Como usar o script de compatibilidade

1. Baixe o script de compatibilidade do Python GitHube mova-o para um local que tenha acesso ao seu cluster Apache Cassandra existente.
2. O script de compatibilidade usa parâmetros semelhantes aosCQLSH. `--port`Insira `--host` e insira o endereço IP e a porta que você usa para se conectar e executar consultas em um dos nós do Cassandra em seu cluster. Se seu cluster Cassandra usa autenticação, você também precisa fornecer e. `-username` `-password` Para executar o script de compatibilidade, você pode usar o comando a seguir.

```
python toolkit-compat-tool.py --host hostname or IP -u "username" -p "password" --
port native transport port
```

## Estime os preços do Amazon Keyspaces

Esta seção fornece uma visão geral das informações que você precisa coletar das tabelas do Apache Cassandra para calcular o custo estimado do Amazon Keyspaces. Cada uma de suas tabelas exige tipos de dados diferentes, precisa oferecer suporte a diferentes consultas CQL e manter um tráfego de leitura/gravação distinto. Pensar em seus requisitos com base em tabelas se alinha aos modos de isolamento de recursos em nível de tabela e capacidade de taxa de transferência de leitura/gravação do Amazon Keyspaces. Com o Amazon Keyspaces, você pode definir a capacidade de leitura/gravação e as políticas de escalabilidade automática para tabelas de forma independente. Compreender os requisitos das tabelas ajuda você a priorizar tabelas para migração com base na funcionalidade, no custo e no esforço de migração.

Colete as seguintes métricas da tabela do Cassandra antes de uma migração. Essas informações ajudam a estimar o custo da sua carga de trabalho no Amazon Keyspaces.

- Nome da tabela — O nome do espaço de teclas totalmente qualificado e o nome da tabela.
- Descrição — Uma descrição da tabela, por exemplo, como ela é usada ou que tipo de dados são armazenados nela.
- Média de leituras por segundo — O número médio de leituras em nível de coordenadas na tabela em um grande intervalo de tempo.
- Média de gravações por segundo — O número médio de gravações em nível de coordenadas na tabela em um grande intervalo de tempo.
- Tamanho médio da linha em bytes — O tamanho médio da linha em bytes.

- Tamanho do armazenamento em GBs — O tamanho bruto do armazenamento de uma tabela.
- Detalhamento da consistência de leitura — A porcentagem de leituras que usam consistência eventual (`LOCAL_ONE`ou`ONE`) versus consistência forte (`LOCAL_QUORUM`).

Esta tabela mostra um exemplo das informações sobre suas tabelas que você precisa reunir ao planejar uma migração.

| Nome da tabela | Descrição | Média de leituras por segundo | Média de gravações por segundo | Tamanho médio da linha em bytes | Tamanho de armazenamento em GBs | Detalhamento da consistência de leitura |
|---|---|---|---|---|---|---|
| mykeyspace.mytable | Usado para armazenar o histórico do carrinho de compras | 10.000 | 5.000 | 2.200 | 2.000 | 100% `LOCAL_ONE` |
| mykeyspace.mytable2 | Usado para armazenar as informações mais recentes do perfil | 20.000 | 1.000 | 850 | 1.000 | 25% `LOCAL_QUORUM` 75% `LOCAL_ONE` |

## Como coletar métricas de tabela

Esta seção fornece instruções passo a passo sobre como coletar as métricas de tabela necessárias do seu cluster Cassandra existente. Essas métricas incluem tamanho da linha, tamanho da tabela e solicitações de leitura/gravação por segundo (RPS). Eles permitem que você avalie os requisitos de capacidade de processamento de uma tabela do Amazon Keyspaces e estime os preços.

Como coletar métricas de tabela na tabela de origem do Cassandra

1. Determine o tamanho da linha

   O tamanho da linha é importante para determinar a capacidade de leitura e a utilização da capacidade de gravação no Amazon Keyspaces. O diagrama a seguir mostra a distribuição típica de dados em um intervalo de tokens do Cassandra.


   Você pode usar um script de amostragem de tamanho de linha disponível GitHubpara coletar métricas de tamanho de linha para cada tabela em seu cluster do Cassandra. O script exporta dados da tabela do Apache Cassandra usando `cqlsh` e calculando o mínimo, o máximo, `awk` a média e o desvio padrão do tamanho da linha em um conjunto de amostra configurável de dados da tabela. O amostrador de tamanho de linha passa os argumentos para`cqlsh`, portanto, os mesmos parâmetros podem ser usados para se conectar e ler do seu cluster do Cassandra.

   A instrução a seguir é um exemplo disso.

```
./row-size-sampler.sh 10.22.33.44 9142 \\
   -u "username" -p "password" --ssl
```

   Para obter mais informações sobre como o tamanho da linha é calculado no Amazon Keyspaces, consulte. the section called “Como calcular o tamanho da linha”

2. Determine o tamanho da tabela

   Com o Amazon Keyspaces, você não precisa provisionar o armazenamento com antecedência. O Amazon Keyspaces monitora continuamente o tamanho faturável de suas tabelas para determinar suas cobranças de armazenamento. O armazenamento é cobrado por GB por mês. O tamanho da tabela do Amazon Keyspaces é baseado no tamanho bruto (não compactado)

de uma única réplica. Para monitorar o tamanho da tabela no Amazon Keyspaces, você pode usar a métrica`BillableTableSizeInBytes`, que é exibida para cada tabela no. AWS Management Console

Para estimar o tamanho faturável da sua tabela do Amazon Keyspaces, você pode usar um desses dois métodos:

- Use o tamanho médio da linha e multiplique pelo número ou linhas.

  Você pode estimar o tamanho da tabela Amazon Keyspaces multiplicando o tamanho médio da linha pelo número de linhas da tabela de origem do Cassandra. Use o script de amostra de tamanho de linha da seção anterior para capturar o tamanho médio da linha. Para capturar a contagem de linhas, você pode usar ferramentas como `dsbulk count` determinar o número total de linhas na tabela de origem.

- Use o `nodetool` para coletar os metadados da tabela.

  `Nodetool`é uma ferramenta administrativa fornecida na distribuição Apache Cassandra que fornece uma visão sobre o estado do processo do Cassandra e retorna os metadados da tabela. Você pode usar `nodetool` para amostrar metadados sobre o tamanho da tabela e, com isso, extrapolar o tamanho da tabela no Amazon Keyspaces. O comando a ser usado é`nodetool tablestats`. Tablestats retorna o tamanho e a taxa de compactação da tabela. O tamanho da tabela é armazenado como `tablelivespace` o da tabela e você pode dividi-lo pelo`compression ratio`. Em seguida, multiplique esse valor de tamanho pelo número de nós. Por fim, divida pelo fator de replicação (normalmente três). Essa é a fórmula completa do cálculo que você pode usar para avaliar o tamanho da tabela.

  ```
  ((tablelivespace / compression ratio) * (total number of nodes))/ (replication
   factor)
  ```

  Vamos supor que seu cluster do Cassandra tenha 12 nós. A execução do `nodetool tablestats` comando retorna um `tablelivespace` de 200 GB e um `compression ratio` de 0,5. O keyspace tem um fator de replicação de três. É assim que o cálculo desse exemplo se parece.

  ```
  (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
  ```

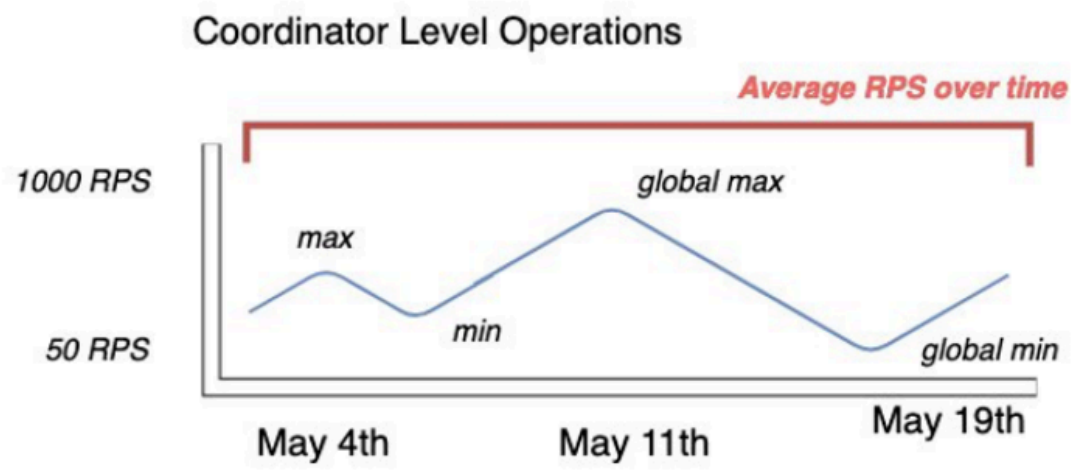
3. Capture o número de leituras e gravações

   Para determinar os requisitos de capacidade e escalabilidade para suas tabelas do Amazon Keyspaces, capture a taxa de solicitação de leitura e gravação de suas tabelas do Cassandra antes da migração.

   O Amazon Keyspaces não tem servidor e você paga somente pelo que usa. Em geral, o preço da taxa de transferência de leitura/gravação no Amazon Keyspaces é baseado no número e no tamanho das solicitações. Há dois modos de capacidade no Amazon Keyspaces: o modo de capacidade sob demanda e o modo de capacidade provisionada. O modo de capacidade sob demanda é uma opção de cobrança flexível capaz de atender a milhares de solicitações por segundo sem a necessidade de planejamento de capacidade. Essa opção oferece pay-per-request preços para solicitações de leitura e gravação, de forma que você pague somente pelo que usar. Se você selecionar o modo de capacidade de throughput provisionada, especifique o número de leituras e gravações por segundo necessárias para seu aplicativo. Isso o ajuda a gerenciar seu uso do Amazon Keyspaces para permanecer em ou abaixo de uma taxa de solicitação definida para otimizar o preço e manter a previsibilidade. O modo provisionado oferece escalonamento automático para ajustar automaticamente sua taxa provisionada para aumentar ou diminuir a escala para melhorar a eficiência operacional. Para obter mais informações sobre o gerenciamento de recursos sem servidor, consulte. Gerenciamento de recursos de tecnologia sem servidor

   Como você provisiona a capacidade de taxa de transferência de leitura e gravação no Amazon Keyspaces separadamente, você precisa medir a taxa de solicitação de leituras e gravações em suas tabelas existentes de forma independente.

   Para reunir as métricas de utilização mais precisas do seu cluster Cassandra existente, capture a média de solicitações por segundo (RPS) para operações de leitura e gravação em nível de coordenador durante um longo período de tempo para uma tabela agregada em todos os nós em um único data center. Capturar o RPS médio em um período de pelo menos várias semanas capturará picos e baixos em seus padrões de tráfego, conforme mostrado no diagrama a seguir.




Você tem duas opções para determinar a taxa de solicitações de leitura e gravação da sua tabela do Cassandra.

- Use o monitoramento existente do Cassandra

  Você pode usar as métricas mostradas na tabela a seguir para observar as solicitações de leitura e gravação. Observe que os nomes das métricas podem mudar com base na ferramenta de monitoramento que você está usando.

| Dimensão | Métrica Cassandra JMX |
| --- | --- |
| Escreve | `org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count` |
| Lê | `org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count` |

- Usar a `nodetool`

  Use `nodetool tablestats` e `nodetool info` para capturar a média de operações de leitura e gravação da tabela. `tablestats`retorna a contagem total de leituras e gravações a partir do momento em que o nó foi iniciado. `nodetool info`fornece o tempo de atividade de um nó em segundos. Para receber a média de leituras e gravações

por segundo, divida a contagem de leitura e gravação pelo tempo de atividade do nó em segundos. Em seguida, para leituras, você divide pelo nível de consistência e, para gravações, divide pelo fator de replicação. Esses cálculos são expressos nas fórmulas a seguir.

Fórmula para leituras médias por segundo:

```
((number of reads * number of nodes in cluster) / read consistency quorum
 (2)) / uptime
```

Fórmula para a média de gravações por segundo:

```
((number of writes * number of nodes in cluster) / replication factor of 3) /
 uptime
```

Vamos supor que temos um cluster de 12 nós que está ativo há 4 semanas. `nodetool info`retorna 2.419.200 segundos de tempo de atividade e `nodetool tablestats` retorna 1 bilhão de gravações e 2 bilhões de leituras. Esse exemplo resultaria no cálculo a seguir.

```
((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. Determine a utilização da capacidade da tabela

   Para estimar a utilização média da capacidade, comece com as taxas médias de solicitação e o tamanho médio da linha da tabela de origem do Cassandra.

   O Amazon Keyspaces usa unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs) para medir a capacidade de transferência provisionada para leituras e gravações em tabelas. Para essa estimativa, usamos essas unidades para calcular as necessidades de capacidade de leitura e gravação da nova tabela Amazon Keyspaces após a migração. Posteriormente neste tópico, discutiremos como a escolha entre o modo de capacidade provisionada e sob demanda afeta o faturamento. Mas, para a estimativa da utilização da capacidade, presumimos que a tabela esteja no modo provisionado.

Uma RCU representa uma solicitação de `LOCAL_QUORUM` leitura, ou duas solicitações de `LOCAL_ONE` leitura, para uma linha de até 4 KB de tamanho. Se você precisar ler uma linha maior que 4 KB, a operação de leitura usará RCUs adicionais. O número total de RCUs necessárias depende do tamanho da linha e se você deseja usar `LOCAL_QUORUM` ou `LOCAL_ONE` ler a consistência. Por exemplo, a leitura de uma linha de 8 KB requer 2 RCUs usando consistência de `LOCAL_QUORUM` leitura e 1 RCU se você escolher `LOCAL_ONE` consistência de leitura.

Uma WCU representa uma gravação para uma linha de até 1 KB de tamanho. Todas as gravações estão usando consistência `LOCAL_QUORUM` e não há cobrança adicional pelo uso de transações leves (LWTs). Se você precisar gravar uma linha maior que 1 KB, a operação de gravação usará WCUs adicionais. O número total de WCUs necessárias depende do tamanho da linha. Por exemplo, se o tamanho da linha for de 2 KB, você precisará de 2 WCUs para realizar uma solicitação de gravação.

A fórmula a seguir pode ser usada para estimar as RCUs e WCUs necessárias. A capacidade de leitura em RCUs pode ser determinada multiplicando as leituras por segundo pelo número de linhas lidas por leitura multiplicado pelo tamanho médio da linha dividido por 4 KB e arredondado para o número inteiro mais próximo.

A capacidade de gravação nas WCUs pode ser determinada multiplicando o número de solicitações pelo tamanho médio da linha dividido por 1 KB e arredondado para o número inteiro mais próximo. Isso é expresso nas fórmulas a seguir.

```
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
```

Por exemplo, se você estiver realizando 4.960 solicitações de leitura com um tamanho de linha de 2,5 KB em sua tabela do Cassandra, precisará de 4.960 RCUs no Amazon Keyspaces. Se você está realizando atualmente 1.653 solicitações de gravação por segundo com um tamanho de linha de 2,5 KB em sua tabela do Cassandra, você precisa de 4.959 WCUs por segundo no Amazon Keyspaces. Esse exemplo é expresso nas fórmulas a seguir.

```
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
```

`eventual consistency`O uso permite que você economize até metade da capacidade de taxa de transferência em cada solicitação de leitura. Cada leitura eventualmente consistente pode consumir até 8 KB. Você pode calcular eventuais leituras consistentes multiplicando o cálculo anterior por 0,5, conforme mostrado na fórmula a seguir.

```
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. Calcule a estimativa de preço mensal para o Amazon Keyspaces

   Para estimar o faturamento mensal da tabela com base na taxa de transferência da capacidade de leitura/gravação, você pode calcular os preços para o modo sob demanda e para o modo provisionado usando fórmulas diferentes e comparar as opções da sua tabela.

   Modo provisionado — o consumo de capacidade de leitura e gravação é cobrado em uma taxa horária com base nas unidades de capacidade por segundo. Primeiro, divida essa taxa por 0,7 para representar a meta de utilização padrão de escalonamento automático de 70%. Em seguida, multiplique por 30 dias corridos, 24 horas por dia e preços tarifários regionais. Esse cálculo está resumido nas fórmulas a seguir.

   ```
   (read capacity per second / .7) * 24 hours * 30 days * regional rate
                   (write capacity per second / .7) * 24 hours * 30 days * regional
    rate
   ```

   Modo sob demanda — a capacidade de leitura e gravação é cobrada de acordo com uma taxa por solicitação. Primeiro, multiplique a taxa de solicitações por 30 dias corridos e 24 horas por dia. Em seguida, divida por um milhão de unidades de solicitação. Por fim, multiplique pela taxa regional. Esse cálculo está resumido nas fórmulas a seguir.

   ```
   ((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
```

## Escolher uma estratégia de migração

Em geral, você pode escolher entre três estratégias de migração diferentes ao migrar do Apache Cassandra para o Amazon Keyspaces:

- Off-line — Essa migração envolve a cópia de um conjunto de dados do Cassandra para o Amazon Keyspaces com uma implantação de migração de aplicativos no estilo azul/verde. Se seu aplicativo puder tolerar algum tempo de inatividade durante a migração, essa opção poderá simplificar o processo de migração. Para obter mais informações sobre migração off-line, consulte the section called “Migração offline”.
- On-line — Essa é uma implantação no estilo canário que normalmente inclui gravações duplas gravadas diretamente na lógica do aplicativo. Os aplicativos que não exigem tempo de inatividade durante a migração exigem que os dados sejam copiados enquanto as leituras e gravações em tempo real são trocadas de uma fonte de dados para outra.
- Híbrido — Essa abordagem permite que as alterações sejam replicadas quase em tempo real, mas o aplicativo é responsável por alternar entre leituras e gravações.

Depois de analisar as estratégias de migração disponíveis com mais detalhes, você pode colocar as opções em uma árvore decisória para simplificar o processo de acordo com seus requisitos e recursos disponíveis.

## Migração off-line para o Amazon Keyspaces

As migrações off-line são adequadas quando você pode arcar com o tempo de inatividade para realizar a migração. É comum entre as empresas ter janelas de manutenção para patches, grandes lançamentos ou períodos de inatividade para atualizações de hardware ou atualizações importantes. A migração off-line pode usar essa janela para copiar dados e transferir o tráfego do aplicativo do Apache Cassandra para o Amazon Keyspaces. A migração off-line reduz as modificações no aplicativo porque não exige comunicação simultânea com o Cassandra e o Amazon Keyspaces. Além disso, com o fluxo de dados pausado, o estado exato pode ser copiado sem manter as mutações.

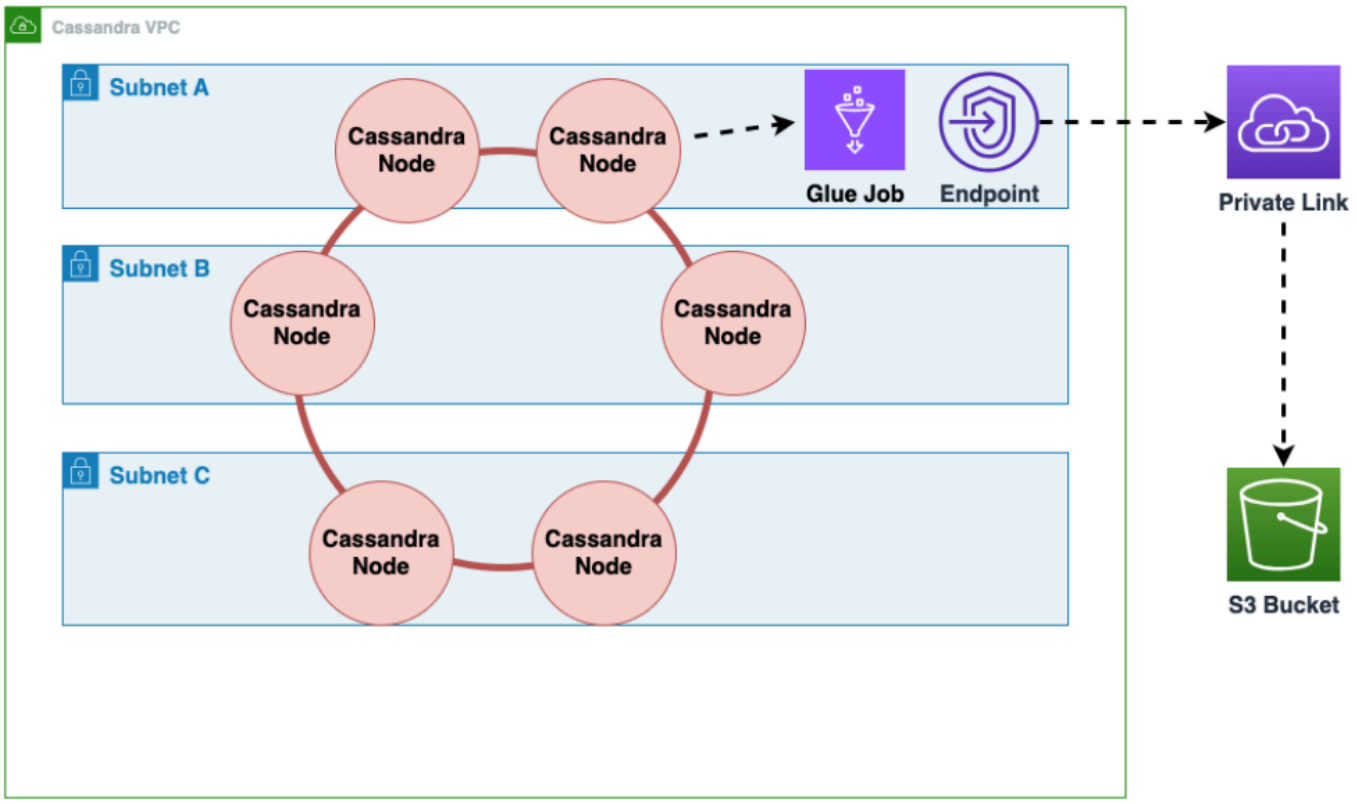
Neste exemplo, você usa o Amazon Simple Storage Service (Amazon S3) como uma área de armazenamento de dados durante a migração off-line para minimizar o tempo de inatividade. Você pode importar automaticamente os dados armazenados no formato Parquet no Amazon S3 em uma tabela do Amazon Keyspaces usando o conector Spark Cassandra e. AWS Glue A seção a seguir mostrará uma visão geral de alto nível do processo. Você pode encontrar exemplos de código para esse processo no Github.

O processo de migração offline do Apache Cassandra para o Amazon Keyspaces usando o Amazon S3 requer os seguintes trabalhos. AWS Glue AWS Glue

1. Um trabalho de ETL que extrai e transforma dados CQL e os armazena em um bucket do Amazon S3.
2. Um segundo trabalho que importa os dados do bucket para o Amazon Keyspaces.
3. Um terceiro trabalho para importar dados incrementais.

Como realizar uma migração off-line do Cassandra para o Amazon Keyspaces em execução no Amazon EC2 em uma Amazon Virtual Private Cloud

1. Primeiro, você pode AWS Glue exportar dados da tabela do Cassandra no formato Parquet e salvá-los em um bucket do Amazon S3. Você precisa executar um AWS Glue trabalho usando um AWS Glue conector para uma VPC onde reside a instância do Amazon EC2 que executa o Cassandra. Em seguida, usando o endpoint privado do Amazon S3, você pode salvar dados no bucket do Amazon S3. O diagrama a seguir ilustra essas etapas.




2. Embaralhe os dados no bucket do Amazon S3 para melhorar a randomização dos dados. Dados importados uniformemente permitem um tráfego mais distribuído na tabela de destino. Essa etapa é necessária ao exportar dados do Cassandra com partições grandes (partições com mais de 1000 linhas) para evitar padrões de teclas de atalho ao inserir os dados no Amazon Keyspaces. Problemas com teclas de atalho causam problemas `WriteThrottleEvents` no Amazon Keyspaces e resultam em maior tempo de carregamento.


3. Use outro AWS Glue trabalho para importar dados do bucket do Amazon S3 para o Amazon Keyspaces. Os dados embaralhados no bucket do Amazon S3 são armazenados no formato Parquet.


# Ferramentas para migrar dados para o Amazon Keyspaces

Diferentes ferramentas estão disponíveis para migrar dados para o Amazon Keyspaces

- Ferramentas de migração
  - Para grandes migrações, considere usar uma ferramenta de extração, transformação e carregamento (ETL). Você pode usar AWS Glue para realizar migrações de transformação de dados de forma rápida e eficaz.
  - Para saber como usar o conector Apache Cassandra do Spark para gravar dados no Amazon Keyspaces, consulte Integração com Apache Spark.
  - Comece rapidamente a carregar dados no Amazon Keyspaces usando o comando cqlsh `COPY FROM`. O cqlsh está incluído no Apache Cassandra e é mais adequado para carregar pequenos conjuntos de dados ou dados de teste. Para step-by-step obter instruções, consultethe section called “Como carregar dados usando cqlsh”.
  - Você também pode usar o DataStax Bulk Loader for Apache Cassandra para carregar dados no Amazon Keyspaces usando o comando. `dsbulk` O DSBulk fornece recursos de importação mais robustos do que o cqlsh e está disponível no repositório. GitHub Para step-by-step obter instruções, consultethe section called “Carregamento de dados usando o DSBulk”.

Tópicos

- Tutorial: Como carregar dados no Amazon Keyspaces usando cqlsh
- Tutorial: Carregamento de dados no Amazon Keyspaces usando o DSBulk

## Tutorial: Como carregar dados no Amazon Keyspaces usando cqlsh

Este step-by-step tutorial orienta você na migração de dados do Apache Cassandra para o Amazon Keyspaces usando o comando. `cqlsh COPY` Neste tutorial, você faz o seguinte:

Tópicos


- Pré-requisitos
- Etapa 1: Criar o arquivo CSV de origem e a tabela de destino
- Etapa 2: Preparar os dados
- Etapa 3: definir a capacidade de throughput da tabela
- Etapa 4: Definir configurações de cqlsh COPY FROM
- Etapa 5: executar o comando cqlsh COPY FROM
- Solução de problemas


## Pré-requisitos

É necessário concluir as tarefas a seguir antes de iniciar este tutorial.

1. Se você ainda não fez isso, inscreva-se em um Conta da AWS seguindo as etapas emthe section called “Conf AWS Identity and Access Management iguração”.
2. Crie credenciais específicas do serviço seguindo as etapas em the section called “Gerar credenciais específicas do serviço usando o console”.
3. Configure a conexão shell do Cassandra Query Language (cqlsh) e confirme se você pode se conectar ao Amazon Keyspaces seguindo as etapas em the section called “Utilizar o `cqlsh`”.

## Etapa 1: Criar o arquivo CSV de origem e a tabela de destino

Neste tutorial, usamos um arquivo de valores separados por vírgula (CSV) com o nome `keyspaces_sample_table.csv` como arquivo de origem para a migração de dados. O arquivo de amostra fornecido contém algumas linhas de dados de uma tabela com o nome `book_awards`.

1. Criar o arquivo de origem. Você pode escolher uma das seguintes opções:
   - Baixe o arquivo CSV de amostra (`keyspaces_sample_table.csv`) contido no seguinte arquivo samplemigration.zip. Descompacte o arquivo e anote o caminho até `keyspaces_sample_table.csv`.
   - Para preencher um arquivo CSV com seus próprios dados armazenados em um banco de dados do Apache Cassandra, você pode preencher o arquivo CSV de origem usando a instrução `cqlsh COPY TO`, conforme mostrado no exemplo a seguir.

```
cqlsh localhost 9042 -u "username" -p "password" --execute
 "COPY mykeyspace.mytable TO 'keyspaces_sample_table.csv' WITH HEADER=true"
```

Certifique-se de que o arquivo CSV criado atenda aos seguintes requisitos:

- A primeira linha contém os nomes das colunas.
- Os nomes das colunas no arquivo CSV de origem correspondem aos nomes das colunas na tabela de destino.
- Os dados são delimitados por uma vírgula.
- Todos os valores de dados são tipos de dados válidos do Amazon Keyspaces. Consulte the section called “Tipos de dados”.

2. Criar o espaço de chaves e a tabela de destino no Amazon Keyspaces.

   a. Conecte-se ao Amazon Keyspaces usando `cqlsh` e substituindo o endpoint do serviço, o nome de usuário e a senha no exemplo a seguir por seus próprios valores.

   ```
   cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
   p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
   ```

   b. Crie um novo espaço de chave com o nome `catalog` mostrado no exemplo a seguir.

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

   c. Quando o novo espaço de chave estiver disponível, use o código a seguir para criar a tabela `book_awards` de destino.

   ```
   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)
      );
   ```

Se o Apache Cassandra for sua fonte de dados original, uma maneira simples de criar a tabela de destino do Amazon Keyspaces com cabeçalhos correspondentes é gerar a instrução `CREATE TABLE` a partir da tabela de origem, conforme mostrado na instrução a seguir.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Em seguida, crie a tabela de destino no Amazon Keyspaces com os nomes das colunas e os tipos de dados correspondentes à descrição da tabela de origem do Cassandra.

## Etapa 2: Preparar os dados

Preparar os dados de origem para uma transferência eficiente é um processo de duas etapas. Primeiro, você randomiza os dados. Na segunda etapa, você analisa os dados para determinar os valores de parâmetros `cqlsh` apropriados e as configurações de tabela necessárias.

Randomizar os dados

O comando `cqlsh COPY FROM` lê e grava dados na mesma ordem em que aparecem no arquivo CSV. Se você usar o comando `cqlsh COPY TO` para criar o arquivo de origem, os dados serão gravados em ordem de classificação por chave no CSV. Internamente, o Amazon Keyspaces particiona os dados usando chaves de partição. Embora o Amazon Keyspaces tenha uma lógica integrada para ajudar a balancear a carga de solicitações para a mesma chave de partição, carregar os dados é mais rápido e eficiente se você randomizar o pedido. Isso ocorre porque você pode aproveitar o balanceamento de carga incorporado que ocorre quando o Amazon Keyspaces está gravando em partições diferentes.

Para distribuir uniformemente as gravações pelas partições, você deve randomizar os dados no arquivo de origem. Você pode escrever um aplicativo para fazer isso ou usar uma ferramenta de código aberto, como o Shuf. O Shuf está disponível gratuitamente em distribuições do Linux, no macOS (instalando coreutils no homebrew) e no Windows (usando o Subssistema Windows para Linux (WSL)). É necessária uma etapa extra para evitar que a linha do cabeçalho com os nomes das colunas sejam embaralhados nessa etapa.

Para randomizar o arquivo de origem enquanto preserva o cabeçalho, insira o código a seguir.

```
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
```

O Shuf reescreve os dados em um novo arquivo CSV chamado `keyspace.table.csv`. Agora você pode excluir o arquivo `keyspaces_sample_table.csv` — ele não é mais necessário.

Analisar os dados

Determine o tamanho médio e máximo da linha analisando os dados.

Você pode fazer isso pelas seguintes razões:

- O tamanho médio da linha ajuda a estimar a quantidade total de dados a serem transferidos.
- Você precisa do tamanho médio da linha para provisionar a capacidade de gravação necessária para o upload dos dados.
- Você pode garantir que cada linha tenha menos de 1 MB, que é o tamanho máximo da linha no Amazon Keyspaces.

> **Note**
>
> Essa cota se refere ao tamanho da linha, não ao tamanho da partição. Diferentemente das partições do Apache Cassandra, as partições do Amazon Keyspaces podem ser virtualmente desvinculadas em tamanho. As chaves de partição e as colunas de clustering exigem armazenamento adicional para metadados, que você deve adicionar ao tamanho bruto das linhas. Para ter mais informações, consulte the section called “Como calcular o tamanho da linha”.

O código a seguir usa AWK para analisar um arquivo CSV e imprimir o tamanho médio e máximo da linha.

```
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
```

A execução desse código resulta na saída a seguir.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Você usa o tamanho médio da linha na próxima etapa deste tutorial para provisionar a capacidade de gravação da tabela.

## Etapa 3: definir a capacidade de throughput da tabela

Este tutorial mostra como ajustar o cqlsh para carregar dados em um intervalo de tempo definido. Como você sabe com antecedência quantas leituras e gravações você executa, use o modo de capacidade provisionada. Depois de concluir a transferência de dados, você deve definir o modo de capacidade da tabela de acordo com os padrões de tráfego do seu aplicativo. Para saber mais sobre gerenciamento de capacidade, consulte Gerenciamento de recursos de tecnologia sem servidor.

Com o modo de capacidade provisionada, você especifica com antecedência quanta capacidade de leitura e gravação deseja provisionar para sua tabela. A capacidade de gravação é cobrada por hora e medida em unidades de capacidade de gravação (WCUs). Cada WCU tem capacidade de gravação suficiente para suportar a gravação de 1 KB de dados por segundo. Quando você carrega os dados, a taxa de gravação deve estar abaixo do máximo de WCUs (parâmetro: `write_capacity_units`) definidas na tabela de destino.

Por padrão, você pode provisionar até 40.000 WCUs em uma tabela e 80.000 WCUs em todas as tabelas da sua conta. Se precisar aumentar a capacidade, solicite um aumento de cota no console Service Quotas. Para obter mais informações sobre cotas, consulte Cotas.

Calcule o número médio de WCUs necessárias para uma inserção

A inserção de 1 KB de dados por segundo exige 1 WCU. Se o arquivo CSV tiver 360.000 linhas e você quiser carregar todos os dados em 1 hora, deverá gravar 100 linhas por segundo (360.000 linhas / 60 minutos / 60 segundos = 100 linhas por segundo). Se cada linha tiver até 1 KB de dados, para inserir 100 linhas por segundo, você deverá provisionar 100 WCUs em sua tabela. Se cada linha tiver 1,5 KB de dados, você precisará de duas WCUs para inserir uma linha por segundo. Portanto, para inserir 100 linhas por segundo, você deve provisionar 200 WCUs.

Para determinar quantas WCUs você precisa para inserir uma linha por segundo, divida o tamanho médio da linha em bytes por 1024 e arredonde para o número inteiro mais próximo.

Por exemplo, se o tamanho médio da linha for 3000 bytes, você precisará de três WCUs para inserir uma linha por segundo.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Calcule o tempo e a capacidade de carregamento de dados

Agora que você sabe o tamanho médio e o número de linhas em seu arquivo CSV, você pode calcular quantas WCUs são necessárias para carregar os dados em um determinado período de tempo e o tempo aproximado necessário para carregar todos os dados em seu arquivo CSV usando diferentes configurações de WCU.

Por exemplo, se cada linha em seu arquivo tiver 1 KB e você tiver 1.000.000 de linhas em seu arquivo CSV, para carregar os dados em 1 hora, você precisará provisionar pelo menos 278 WCUs em sua tabela durante essa hora.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Defina as configurações de capacidade provisionada

Você pode definir as configurações de capacidade de gravação de uma tabela ao criar a tabela ou usando o comando CQL `ALTER TABLE`. A seguir está a sintaxe para alterar as configurações de capacidade provisionada de uma tabela com o comando CQL `ALTER TABLE`.

```
ALTER TABLE mykeyspace.mytable WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100,
 'write_capacity_units': 278}} ;
```

Para obter a referência completa da linguagem, consulte the section called “ALTER TABLE”.

## Etapa 4: Definir configurações de **`cqlsh COPY FROM`**

Esta seção descreve como determinar os valores dos parâmetros de `cqlsh COPY FROM`. O comando `cqlsh COPY FROM` lê o arquivo CSV que você preparou anteriormente e insere os dados no Amazon Keyspaces usando CQL. O comando divide as linhas e distribui as operações `INSERT` entre um conjunto de operadores. Cada operador estabelece uma conexão com o Amazon Keyspaces e envia solicitações `INSERT` por esse canal.

O comando `cqlsh COPY` não tem lógica interna para distribuir o trabalho uniformemente entre seus operadores. No entanto, você pode configurá-lo manualmente para garantir que o trabalho seja distribuído uniformemente. Comece revisando esses principais parâmetros do cqlsh:

- DELIMITADOR: se você usou um delimitador diferente de uma vírgula, você pode definir esse parâmetro, cujo padrão é vírgula.
- INGESTRATE: o número alvo de linhas que `cqlsh COPY FROM` tenta processar por segundo. Se não especificado, o padrão será 100.000.
- NUMPROCESSES: o número de processos do operador secundário que o cqlsh cria para tarefas `COPY FROM`. O máximo para essa configuração é 16, o padrão é `num_cores - 1`, onde `num_cores` é o número de núcleos de processamento no host executando cqlsh.
- MAXBATCHSIZE: o tamanho do lote determina o número máximo de linhas inseridas na tabela de destino em um único lote. Se não for definido, o cqlsh usa lotes de 20 linhas inseridas.
- CHUNKSIZE: o tamanho da unidade de trabalho que passa para o operador secundário. Por padrão, ele é definido como 5.000.
- MAXATTEMPTS: o número máximo de novas tentativas de um bloco de operadores com falha. Depois que a tentativa máxima é atingida, os registros com falha são gravados em um novo arquivo CSV que você pode executar novamente depois de investigar a falha.

Defina `INGESTRATE` com base no número de WCUs que você provisionou na tabela de destino pretendida. O `INGESTRATE` do comando `cqlsh COPY FROM` não é um limite, é uma média de destino. Isso significa que ele pode (e geralmente acontece) ultrapassar o número que você definiu. Para permitir picos e garantir que haja capacidade suficiente para lidar com as solicitações de carregamento de dados, defina `INGESTRATE` como 90% da capacidade de gravação da tabela.

```
INGESTRATE = WCUs * .90
```

Em seguida, defina o parâmetro `NUMPROCESSES` como igual a um a menos que o número de núcleos em seu sistema. Para descobrir qual é o número de núcleos do seu sistema, você pode executar o código a seguir.

```
python -c "import multiprocessing; print(multiprocessing.cpu_count())"
```

Neste tutorial, use o valor a seguir.

```
NUMPROCESSES = 4
```

Cada processo cria um operador e cada operador estabelece uma conexão com o Amazon Keyspaces. O Amazon Keyspaces pode suportar até 3.000 solicitações de CQL por segundo em

cada conexão. Isso significa que você precisa garantir que cada operador esteja processando menos de 3.000 solicitações por segundo.

Assim como INGESTRATE, os operadores geralmente ultrapassam o número definido e não são limitados pelos segundos do relógio. Portanto, para contabilizar as intermitências, defina seus parâmetros de cqlsh para que cada operador processe 2.500 solicitações por segundo. Para calcular a quantidade de trabalho distribuída a um operador, use a seguinte diretriz.

- Divida INGESTRATE por NUMPROCESSES.
- Se INGESTRATE / NUMPROCESSES > 2.500, diminua INGESTRATE para tornar essa fórmula verdadeira.

```
INGESTRATE / NUMPROCESSES <= 2,500
```

Antes de definir as configurações para otimizar o upload de nossos dados de amostra, vamos analisar as configurações padrão cqlsh e ver como o uso delas afeta o processo de carregamento de dados. Como cqlsh COPY FROM usa o CHUNKSIZE para criar blocos do trabalho (instruções INSERT) para distribuir aos operadores, o trabalho não é automaticamente distribuído uniformemente. Alguns operadores podem ficar ociosos, dependendo da configuração de INGESTRATE.

Para distribuir o trabalho uniformemente entre os operadores e manter cada operador na taxa ideal de 2.500 solicitações por segundo, você deve definir CHUNKSIZE, MAXBATCHSIZE e INGESTRATE alterando os parâmetros de entrada. Para otimizar a utilização do tráfego de rede durante o carregamento de dados, escolha um valor para MAXBATCHSIZE próximo ao valor máximo de 30. Ao mudar CHUNKSIZE para 100 e MAXBATCHSIZE para 25, as 10.000 linhas são distribuídas uniformemente entre os quatro operadores (10.000/2.500 = 4).

O exemplo de código a seguir ilustra isso.

```
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
```

Para resumir, use as seguintes fórmulas ao definir os parâmetros de `cqlsh COPY FROM`:

- INGESTRATE = write_capacity_units * .90
- NUMPROCESSES = num_cores -1 (padrão)
- INGESTRATE/NUMPROCESSES = 2.500 (Essa deve ser uma afirmação verdadeira.)
- MAXBATCHSIZE = 30 (O padrão é 20. O Amazon Keyspaces aceita lotes de até 30.)
- CHUNKSIZE = (INGESTRATE / NUMPROCESSES) / MAXBATCHSIZE

Agora que você calculou `NUMPROCESSES`, `INGESTRATE` e `CHUNKSIZE`, você está pronto para carregar seus dados.

### Etapa 5: executar o comando **`cqlsh COPY FROM`**

Conclua as etapas a seguir para executar o comando `cqlsh COPY FROM`.

1. Conecte-se ao Amazon Keyspaces usando cqlsh.
2. Escolha seu espaço de chave com o código a seguir.

   ```
   USE catalog;
   ```

3. Defina a consistência de gravação como `LOCAL_QUORUM`. Para garantir a durabilidade dos dados, o Amazon Keyspaces não permite outras configurações de consistência de gravação. Consulte o código a seguir.

   ```
   CONSISTENCY LOCAL_QUORUM;
   ```

4. Prepare sua sintaxe `cqlsh COPY FROM` usando o exemplo de código a seguir.

   ```
   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. Execute a instrução preparada na etapa anterior. O cqlsh reproduz todas as configurações que você definiu.

a. Verifique se as configurações correspondem à entrada. Veja o exemplo a seguir.

```
Reading options from the command line: {'chunksize': '120', 'header': 'true',
 'ingestrate': '36000', 'numprocesses': '15', 'maxbatchsize': '20'}
Using 15 child processes
```

b. Verifique o número de linhas transferidas e a taxa média atual, conforme mostrado no exemplo a seguir.

```
Processed: 57834 rows; Rate: 6561 rows/s; Avg. rate: 31751 rows/s
```

c. Quando o cqlsh terminar de carregar os dados, revise o resumo das estatísticas de carregamento de dados (o número de arquivos lidos, o runtime e as linhas ignoradas), conforme mostrado no exemplo a seguir.

```
15556824 rows imported from 1 files in 8 minutes and 8.321 seconds (0 skipped).
```

Nesta etapa final do tutorial, você fez o upload dos dados para o Amazon Keyspaces.

> **Important**
>
> Agora que você transferiu seus dados, ajuste as configurações do modo de capacidade da tabela de destino para corresponder aos padrões de tráfego regulares do seu aplicativo. Você incorre em cobranças de acordo com a taxa horária de sua capacidade provisionada até alterá-la.

## Solução de problemas

Depois que o upload dos dados for concluído, verifique se as linhas foram ignoradas. Para fazer isso, navegue até o diretório de origem do arquivo CSV de origem e pesquise um arquivo com o nome a seguir.

```
import_yourcsvfilename.err.timestamp.csv
```

O cqlsh grava todas as linhas de dados ignoradas em um arquivo com esse nome. Se o arquivo existir em seu diretório de origem e tiver dados nele, essas linhas não foram carregadas no Amazon Keyspaces. Para repetir essas linhas, primeiro verifique se há erros encontrados durante o upload

e ajuste os dados adequadamente. Para repetir essas linhas, você pode executar o processo novamente.

### Erros comuns

Os motivos mais comuns pelos quais as linhas não são carregadas são erros de capacidade e erros de análise.

Erros de solicitação inválidos ao fazer o upload de dados para o Amazon Keyspaces

No exemplo a seguir, a tabela de origem contém uma coluna de contador, que resulta em chamadas em lote registradas do comando cqlsh `COPY`. As chamadas em lote registradas não são suportadas pelo 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
```

Para corrigir esse erro, use o DSBulk para migrar dados. Para ter mais informações, consulte the section called "Carregamento de dados usando o DSBulk".

Erros do analisador ao fazer o upload de dados para o Amazon Keyspaces

O exemplo a seguir mostra uma linha ignorada devido a `ParseError`.

```
Failed to import 1 rows: ParseError - Invalid ... –
```

Para resolver esse erro, você precisa garantir que os dados a serem importados correspondam ao esquema da tabela no Amazon Keyspaces. Verifique se há erros de análise no arquivo de importação. Você pode tentar usar uma única linha de dados usando uma instrução `INSERT` para isolar o erro.

Erros de capacidade ao fazer o upload de dados para o 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
```

O Amazon Keyspaces usa as exceções `ReadTimeout` e `WriteTimeout` para indicar quando uma solicitação de gravação falha devido à capacidade de throughput insuficiente. Para ajudar a diagnosticar exceções de capacidade insuficiente, o Amazon Keyspaces publica `WriteThrottleEvents` uma métrica na Amazon. `ReadThrottledEvents` CloudWatch Para ter mais informações, consulte the section called “Monitoramento com CloudWatch”.

Erros de cqlsh ao fazer upload de dados para o Amazon Keyspaces

Para ajudar a solucionar erros de cqlsh, execute novamente o comando com falha com o sinalizador `--debug`.

Ao usar uma versão incompatível do cqlsh, você vê o seguinte erro.

```
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
```

Confirme se a versão correta do cqlsh foi instalada executando o comando a seguir.

```
cqlsh --version
```

Você deve ver algo parecido com a saída a seguir.

```
cqlsh 5.0.1
```

Se estiver usando o Windows, substitua todas as instâncias de `cqlsh` por `cqlsh.bat`. Por exemplo, para verificar a versão do cqlsh no Windows, execute o comando a seguir.

```
cqlsh.bat --version
```

A conexão com o Amazon Keyspaces falha depois que o cliente cqlsh recebe três erros consecutivos de qualquer tipo do servidor. O cliente cqlsh falha com a seguinte mensagem.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Para resolver esse erro, você precisa garantir que os dados a serem importados correspondam ao esquema da tabela no Amazon Keyspaces. Verifique se há erros de análise no arquivo de importação. Você pode tentar usar uma única linha de dados usando uma instrução INSERT para isolar o erro.

O cliente tenta automaticamente restabelecer a conexão.

# Tutorial: Carregamento de dados no Amazon Keyspaces usando o DSBulk

Este step-by-step tutorial orienta você na migração de dados do Apache Cassandra para o Amazon Keyspaces usando DataStax o Bulk Loader (DSBulk) disponível em. GitHub Neste tutorial, você concluirá as seguintes etapas:

Tópicos


- Pré-requisitos
- Etapa 1: criar o arquivo CSV de origem e a tabela de destino
- Etapa 2: preparar os dados
- Etapa 3: definir a capacidade de throughput da tabela
- Etapa 4: definir as configurações do DSBulk
- Etapa 5: executar o comando load do DSBulk


## Pré-requisitos

É necessário concluir as tarefas a seguir antes de iniciar este tutorial.

1. Se você ainda não fez isso, inscreva-se em uma AWS conta seguindo as etapas emthe section called “Conf AWS Identity and Access Management iguração”.
2. Crie credenciais seguindo as etapas em the section called “Credenciais do IAM para autenticação AWS ”.
3. Crie um arquivo JKS de armazenamento confiável.

    a. Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

    > **Note**
    >
    > Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade

adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.

b. Converta o certificado digital Starfield em um arquivo 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
```

Nesta etapa, você precisa criar uma senha para o repositório de chaves e confiar nesse certificado. O comando interativo tem a aparência a seguir.

```
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. Configure a conexão shell do Cassandra Query Language (cqlsh) e confirme se você pode se conectar ao Amazon Keyspaces seguindo as etapas em the section called “Utilizar o `cqlsh`”.
5. Faça download e instale o DSBulk.

   a. Para baixar o DSBulk, você pode usar o código a seguir.

   ```
   curl -OL https://downloads.datastax.com/dsbulk/dsbulk-1.8.0.tar.gz
   ```

   b. Em seguida, descompacte o arquivo tar e adicione o DSBulk ao seu PATH, conforme mostrado no exemplo a seguir.

   ```
   tar -zxvf dsbulk-1.8.0.tar.gz
   # add the DSBulk directory to the path
   export PATH=$PATH:./dsbulk-1.8.0/bin
   ```

   c. Crie um arquivo `application.conf` para armazenar as configurações a serem usadas pelo DSBulk. Você pode salvar o exemplo a seguir como `./dsbulk_keyspaces.conf`. Substitua `localhost` pelo ponto de contato do seu cluster Cassandra local se você não estiver no nó local, por exemplo, o nome DNS ou o endereço IP. Anote o nome e o caminho do arquivo, pois você precisará especificar isso posteriormente no comando `dsbulk load`.

   ```
   datastax-java-driver {
     basic.contact-points = [ "localhost"]
     advanced.auth-provider {
           class = software.aws.mcs.auth.SigV4AuthProvider
           aws-region = us-east-1
     }
   }
   ```

d. Para ativar o suporte ao SigV4, baixe o `jar` arquivo sombreado GitHube coloque-o na `lib` pasta DSBulk, conforme mostrado no exemplo a seguir.

```
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
```

## Etapa 1: criar o arquivo CSV de origem e a tabela de destino

Neste tutorial, usamos um arquivo de valores separados por vírgula (CSV) com o nome `keyspaces_sample_table.csv` como arquivo de origem para a migração de dados. O arquivo de amostra fornecido contém algumas linhas de dados de uma tabela com o nome `book_awards`.

1. Criar o arquivo de origem. Você pode escolher uma das seguintes opções:
   - Baixe o arquivo CSV de amostra (`keyspaces_sample_table.csv`) contido no seguinte arquivo samplemigration.zip. Descompacte o arquivo e anote o caminho até `keyspaces_sample_table.csv`.
   - Para preencher um arquivo CSV com seus próprios dados armazenados em um banco de dados Apache Cassandra, você pode preencher o arquivo CSV de origem usando `dsbulk unload`, conforme mostrado no exemplo a seguir.

     ```
     dsbulk unload -k mykeyspace -t mytable -f ./my_application.conf
      > keyspaces_sample_table.csv
     ```

     Certifique-se de que o arquivo CSV criado atenda aos seguintes requisitos:

     - A primeira linha contém os nomes das colunas.
     - Os nomes das colunas no arquivo CSV de origem correspondem aos nomes das colunas na tabela de destino.
     - Os dados são delimitados por uma vírgula.
     - Todos os valores de dados são tipos de dados válidos do Amazon Keyspaces. Consulte the section called “Tipos de dados”.
2. Criar o espaço de chaves e a tabela de destino no Amazon Keyspaces.

   a. Conecte-se ao Amazon Keyspaces usando `cqlsh` e substituindo o endpoint do serviço, o nome de usuário e a senha no exemplo a seguir por seus próprios valores.

```
cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Crie um novo espaço de chave com o nome `catalog` mostrado no exemplo a seguir.

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

c. Depois que o novo keyspace tiver o status de disponível, use o código a seguir para criar a tabela `book_awards` de destino. Para saber mais sobre a criação assíncrona de recursos e como verificar se um recurso está disponível, consulte the section called “Como criar espaços de chaves”.

```
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)
   );
```

Se o Apache Cassandra for sua fonte de dados original, uma maneira simples de criar a tabela de destino do Amazon Keyspaces com cabeçalhos correspondentes é gerar a declaração `CREATE TABLE` a partir da tabela de origem, conforme mostrado na declaração a seguir.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Em seguida, crie a tabela de destino no Amazon Keyspaces com os nomes das colunas e os tipos de dados correspondentes à descrição da tabela de origem do Cassandra.

## Etapa 2: preparar os dados

Preparar os dados de origem para uma transferência eficiente é um processo de duas etapas. Primeiro, você randomiza os dados. Na segunda etapa, você analisa os dados para determinar os valores de parâmetros `dsbulk` apropriados e as configurações de tabela necessárias.

### Randomizar os dados

O comando `dsbulk` lê e grava dados na mesma ordem em que aparecem no arquivo CSV. Se você usar o comando `dsbulk` para criar o arquivo de origem, os dados serão gravados em ordem de classificação por chave no CSV. Internamente, o Amazon Keyspaces particiona os dados usando chaves de partição. Embora o Amazon Keyspaces tenha uma lógica integrada para ajudar a balancear a carga de solicitações para a mesma chave de partição, carregar os dados é mais rápido e eficiente se você randomizar o pedido. Isso ocorre porque você pode aproveitar o balanceamento de carga incorporado que ocorre quando o Amazon Keyspaces está gravando em partições diferentes.

Para distribuir uniformemente as gravações pelas partições, você deve randomizar os dados no arquivo de origem. Você pode escrever um aplicativo para fazer isso ou usar uma ferramenta de código aberto, como o Shuf. O Shuf está disponível gratuitamente em distribuições do Linux, no macOS (instalando coreutils no homebrew) e no Windows (usando o Subssistema Windows para Linux (WSL)). É necessária uma etapa extra para evitar que a linha do cabeçalho com os nomes das colunas sejam embaralhados nessa etapa.

Para randomizar o arquivo de origem enquanto preserva o cabeçalho, insira o código a seguir.

```
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
```

O Shuf reescreve os dados em um novo arquivo CSV chamado `keyspace.table.csv`. Agora você pode excluir o arquivo `keyspaces_sample_table.csv` — ele não é mais necessário.

### Analisar os dados

Determine o tamanho médio e máximo da linha analisando os dados.

Você pode fazer isso pelas seguintes razões:

- O tamanho médio da linha ajuda a estimar a quantidade total de dados a serem transferidos.
- Você precisa do tamanho médio da linha para provisionar a capacidade de gravação necessária para o upload dos dados.
- Você pode garantir que cada linha tenha menos de 1 MB, que é o tamanho máximo da linha no Amazon Keyspaces.

> **Note**
>
> Essa cota se refere ao tamanho da linha, não ao tamanho da partição. Diferentemente das partições do Apache Cassandra, as partições do Amazon Keyspaces podem ser virtualmente desvinculadas em tamanho. As chaves de partição e as colunas de clustering exigem armazenamento adicional para metadados, que você deve adicionar ao tamanho bruto das linhas. Para ter mais informações, consulte the section called “Como calcular o tamanho da linha”.

O código a seguir usa AWK para analisar um arquivo CSV e imprimir o tamanho médio e máximo da linha.

```
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
```

A execução desse código resulta na saída a seguir.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Certifique-se de que o tamanho máximo da linha não exceda 1 MB. Se isso acontecer, você precisará dividir a linha ou compactar os dados para reduzir o tamanho da linha para menos de 1 MB. Na próxima etapa deste tutorial, você usa o tamanho médio da linha para provisionar a capacidade de gravação da tabela.

## Etapa 3: definir a capacidade de throughput da tabela

Este tutorial mostra como ajustar o DSBulk para carregar dados em um intervalo de tempo definido. Como você sabe com antecedência quantas leituras e gravações você executa, use o modo de capacidade provisionada. Depois de concluir a transferência de dados, você deve definir o modo de capacidade da tabela de acordo com os padrões de tráfego do seu aplicativo. Para saber mais sobre gerenciamento de capacidade, consulte Gerenciamento de recursos de tecnologia sem servidor.

Com o modo de capacidade provisionada, você especifica com antecedência quanta capacidade de leitura e gravação deseja provisionar para sua tabela. A capacidade de gravação é cobrada por hora e medida em unidades de capacidade de gravação (WCUs). Cada WCU tem capacidade

de gravação suficiente para suportar a gravação de 1 KB de dados por segundo. Quando você carrega os dados, a taxa de gravação deve estar abaixo do máximo de WCUs (parâmetro: `write_capacity_units`) definidas na tabela de destino.

Por padrão, você pode provisionar até 40.000 WCUs em uma tabela e 80.000 WCUs em todas as tabelas da sua conta. Se precisar aumentar a capacidade, solicite um aumento de cota no console Service Quotas. Para obter mais informações sobre cotas, consulte Cotas.

Calcule o número médio de WCUs necessárias para uma inserção

A inserção de 1 KB de dados por segundo exige 1 WCU. Se o arquivo CSV tiver 360.000 linhas e você quiser carregar todos os dados em 1 hora, deverá gravar 100 linhas por segundo (360.000 linhas / 60 minutos / 60 segundos = 100 linhas por segundo). Se cada linha tiver até 1 KB de dados, para inserir 100 linhas por segundo, você deverá provisionar 100 WCUs em sua tabela. Se cada linha tiver 1,5 KB de dados, você precisará de duas WCUs para inserir uma linha por segundo. Portanto, para inserir 100 linhas por segundo, você deve provisionar 200 WCUs.

Para determinar quantas WCUs você precisa para inserir uma linha por segundo, divida o tamanho médio da linha em bytes por 1024 e arredonde para o número inteiro mais próximo.

Por exemplo, se o tamanho médio da linha for 3000 bytes, você precisará de três WCUs para inserir uma linha por segundo.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Calcule o tempo e a capacidade de carregamento de dados

Agora que você sabe o tamanho médio e o número de linhas em seu arquivo CSV, você pode calcular quantas WCUs são necessárias para carregar os dados em um determinado período de tempo e o tempo aproximado necessário para carregar todos os dados em seu arquivo CSV usando diferentes configurações de WCU.

Por exemplo, se cada linha em seu arquivo tiver 1 KB e você tiver 1.000.000 de linhas em seu arquivo CSV, para carregar os dados em 1 hora, você precisará provisionar pelo menos 278 WCUs em sua tabela durante essa hora.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Defina as configurações de capacidade provisionada

Você pode definir as configurações de capacidade de gravação de uma tabela ao criar a tabela ou usando o comando `ALTER TABLE`. A seguir está a sintaxe para alterar as configurações de capacidade provisionada de uma tabela com o comando `ALTER TABLE`.

```
ALTER TABLE catalog.book_awards WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100, 'write_capacity_units':
 278}} ;
```

Para obter a referência completa do idioma, consulte the section called “CRIAR TABELA” e the section called “ALTER TABLE”.

## Etapa 4: definir as configurações do **DSBulk**

Esta seção descreve as etapas necessárias para configurar o DSBulk para upload de dados para o Amazon Keyspaces. Você configura o DSBulk usando um arquivo de configuração. Você especifica o arquivo de configuração diretamente da linha de comando.

1. Crie um arquivo de configuração DSBulk para a migração para o Amazon Keyspaces. Neste exemplo, usamos o nome de arquivo `dsbulk_keyspaces.conf`. Especifique as configurações a seguir no arquivo de configuração DSBulk.

   a. *`PlainTextAuthProvider`* – Crie o provedor de autenticação com a classe `PlainTextAuthProvider`. `ServiceUserName` e `ServicePassword` devem corresponder ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas em the section called “Criação de credenciais”.

   b. *`local-datacenter`*— Defina o valor do `local-datacenter` ao Região da AWS qual você está se conectando. Por exemplo, se o aplicativo estiver se conectando a `cassandra.us-east-2.amazonaws.com`, defina o datacenter local como `us-east-2`. Para todos os disponíveis Regiões da AWS, consultethe section called “Service endpoints (Endpoints de serviço)”. Para evitar réplicas, defina `slow-replica-avoidance` como `false`.

   c. *`SSLEngineFactory`* – Para configurar o SSL/TLS, inicialize o `SSLEngineFactory` adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com `class = DefaultSslEngineFactory`. Forneça o caminho para `cassandra_truststore.jks` e a senha que você criou anteriormente.

   d. *`consistency`* – Defina o nível de consistência como `LOCAL_QUORUM`. Outros níveis de consistência de gravação não são suportados; para obter mais informações, consulte the section called “Níveis de consistência suportados do Cassandra”.

e. O número de conexões por grupo é configurável no driver Java. Para este exemplo, defina `advanced.connection.pool.local.size` como 3.

Veja a seguir o arquivo de configuração de exemplo completo.

```
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. Revise os parâmetros do comando `load` do DSBulk.

a. *`executor.maxPerSecond`* – O número máximo de linhas que o comando load tenta processar simultaneamente por segundo. Se não for definida, essa configuração é desativada com -1.

Defina `executor.maxPerSecond` com base no número de WCUs que você provisionou na tabela de destino pretendida. O `executor.maxPerSecond` do comando `load` não é um limite – é uma média de destino. Isso significa que ele pode (e geralmente acontece) ultrapassar o número que você definiu. Para permitir picos e garantir que haja

capacidade suficiente para lidar com as solicitações de carregamento de dados, defina `executor.maxPerSecond` como 90% da capacidade de gravação da tabela.

```
executor.maxPerSecond = WCUs * .90
```

Neste tutorial, definimos `executor.maxPerSecond` como 5.

> **Note**
>
> Se você estiver usando o DSBulk 1.6.0 ou superior, você pode usar `dsbulk.engine.maxConcurrentQueries` em vez disso.

b. Configure esses parâmetros adicionais para o comando `load` do DSBulk.

- *`batch-mode`* – Esse parâmetro instrui o sistema a agrupar as operações por chave de partição. Recomendamos desativar o modo em lote, pois isso pode resultar em cenários e causas de teclas de atalho`WriteThrottleEvents`.
- *`driver.advanced.retry-policy-max-retries`* – Isso determina quantas vezes tentar novamente uma consulta com falha. Se não estiver definido, o padrão é 10. Você pode ajustar esse valor conforme necessário.
- *`driver.basic.request.timeout`* – O tempo em minutos em que o sistema espera o retorno de uma consulta. Se não estiver definido, o padrão é "5 minutos". Você pode ajustar esse valor conforme necessário.

## Etapa 5: executar o comando **`load`** do DSBulk

Na etapa final deste tutorial, você carrega os dados no Amazon Keyspaces.

Conclua as etapas a seguir para executar o comando `load` do DSBulk.

1. Execute o código a seguir para fazer o upload dos dados do seu arquivo csv para a tabela do Amazon Keyspaces. Certifique-se de atualizar o caminho para o arquivo de configuração do aplicativo que você criou anteriormente.

```
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. A saída inclui a localização de um arquivo de log que detalha as operações bem-sucedidas e malsucedidas. O arquivo é armazenado no diretório a seguir.

```
Operation directory: /home/user_name/logs/UNLOAD_20210308-202317-801911
```

3. As entradas do arquivo de log incluirão métricas, como no exemplo a seguir. Verifique se o número de linhas é consistente com o número de linhas em seu arquivo csv.

```
total | failed | rows/s | p50ms | p99ms | p999ms
   200 |      0 |    200 | 21.63 | 21.89 |  21.89
```

> **Important**
>
> Agora que você transferiu seus dados, ajuste as configurações do modo de capacidade da tabela de destino para corresponder aos padrões de tráfego regulares do seu aplicativo. Você incorre em cobranças de acordo com a taxa horária de sua capacidade provisionada até alterá-la. Para ter mais informações, consulte the section called “Modos de capacidade de leitura/gravação”.

# Exemplos de código para Amazon Keyspaces usando SDKs AWS

Os exemplos de código a seguir mostram como usar o Amazon Keyspaces com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos

## Olá, Amazon Keyspaces

Os exemplos de código a seguir mostram como começar a usar o Amazon Keyspaces.

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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();
    }
}
```

- Para obter detalhes da API, consulte ListKeyspacesa Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte ListKeyspacesa Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
/**
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}")
        }
    }
}
```

- Para obter detalhes da API, consulte a ListKeyspacesreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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"))
```

- Para obter detalhes da API, consulte a ListKeyspacesReferência da API AWS SDK for Python (Boto3).

Exemplos de código

- Ações para Amazon Keyspaces usando SDKs AWS
  - Use CreateKeyspace com um AWS SDK ou CLI
  - Use CreateTable com um AWS SDK ou CLI
  - Use DeleteKeyspace com um AWS SDK ou CLI
  - Use DeleteTable com um AWS SDK ou CLI
  - Use GetKeyspace com um AWS SDK ou CLI
  - Use GetTable com um AWS SDK ou CLI
  - Use ListKeyspaces com um AWS SDK ou CLI
  - Use ListTables com um AWS SDK ou CLI
  - Use RestoreTable com um AWS SDK ou CLI
  - Use UpdateTable com um AWS SDK ou CLI
- Cenários para Amazon Keyspaces usando SDKs AWS
  - Comece a usar tabelas e espaços de chave do Amazon Keyspaces usando um SDK AWS

# Ações para Amazon Keyspaces usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Keyspaces com AWS SDKs. Esses trechos chamam a API do Amazon Keyspaces e são trechos de código de programas maiores que devem ser executados no contexto. Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência da API do Amazon Keyspaces (para Apache Cassandra).

Exemplos

- Use CreateKeyspace com um AWS SDK ou CLI
- Use CreateTable com um AWS SDK ou CLI

- Use DeleteKeyspace com um AWS SDK ou CLI
- Use DeleteTable com um AWS SDK ou CLI
- Use GetKeyspace com um AWS SDK ou CLI
- Use GetTable com um AWS SDK ou CLI
- Use ListKeyspaces com um AWS SDK ou CLI
- Use ListTables com um AWS SDK ou CLI
- Use RestoreTable com um AWS SDK ou CLI
- Use UpdateTable com um AWS SDK ou CLI

## Use **`CreateKeyspace`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `CreateKeyspace`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
/// <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;
    }
```

- Para obter detalhes da API, consulte CreateKeyspacea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte CreateKeyspacea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
    }
}
```

- Para obter detalhes da API, consulte a CreateKeyspacereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a CreateKeyspaceReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use `CreateTable` com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `CreateTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   /// <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;
    }
```

- Para obter detalhes da API, consulte CreateTablea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
    }
```

```
    }
```

- Para obter detalhes da API, consulte CreateTablea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
    }
}
```

- Para obter detalhes da API, consulte a CreateTablereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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"]
```

- Para obter detalhes da API, consulte a CreateTableReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Use **`DeleteKeyspace`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `DeleteKeyspace`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
/// <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;
}
```

- Para obter detalhes da API, consulte DeleteKeyspacea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte DeleteKeyspacea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}
```

- Para obter detalhes da API, consulte a DeleteKeyspacereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a DeleteKeyspaceReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`DeleteTable`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `DeleteTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte DeleteTablea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    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);
        }
    }
```

- Para obter detalhes da API, consulte DeleteTablea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
```

```
}
```

- Para obter detalhes da API, consulte a DeleteTablereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a DeleteTableReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`GetKeyspace`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `GetKeyspace`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte GetKeyspacea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte GetKeyspacea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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")
    }
}
```

- Para obter detalhes da API, consulte a GetKeyspacereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a GetKeyspaceReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`GetTable`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `GetTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte GetTablea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    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);
        }
    }
```

- Para obter detalhes da API, consulte GetTablea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
            }
        }
    }
}
```

- Para obter detalhes da API, consulte a GetTablereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a GetTableReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`ListKeyspaces`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ListKeyspaces`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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}");
        }
    }
```

- Para obter detalhes da API, consulte ListKeyspacesa Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte ListKeyspacesa Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
            }
    }
}
```

- Para obter detalhes da API, consulte a ListKeyspacesreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a ListKeyspacesReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Use `ListTables` com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ListTables`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte ListTablesa Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte ListTablesa Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
            }
    }
}
```

- Para obter detalhes da API, consulte a ListTablesreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a ListTablesReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`RestoreTable`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `RestoreTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte RestoreTablea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
```

```
    }
```

- Para obter detalhes da API, consulte RestoreTablea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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}")
    }
}
```

- Para obter detalhes da API, consulte a RestoreTablereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a RestoreTableReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

## Use **`UpdateTable`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `UpdateTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos de keyspaces e tabelas

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte UpdateTablea Referência AWS SDK for .NET da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
   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);
       }
   }
```

- Para obter detalhes da API, consulte UpdateTablea Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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)
    }
}
```

- Para obter detalhes da API, consulte a UpdateTablereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
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
```

- Para obter detalhes da API, consulte a UpdateTableReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Cenários para Amazon Keyspaces usando SDKs AWS

Os exemplos de código a seguir mostram como implementar cenários comuns no Amazon Keyspaces com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções no Amazon Keyspaces. Cada cenário inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código.

Exemplos

- Comece a usar tabelas e espaços de chave do Amazon Keyspaces usando um SDK AWS

## Comece a usar tabelas e espaços de chave do Amazon Keyspaces usando um SDK AWS

Os exemplos de código a seguir mostram como:

- Crie um keyspace e uma tabela. O esquema da tabela contém dados do filme e tem a point-in-time recuperação ativada.
- Conectar-se ao keyspace usando uma conexão TLS segura com autenticação SigV4.
- Consultar a tabela. Adicionar, recuperar e atualizar dados do filme.

- Atualizar a tabela. Adicionar uma coluna para rastrear os filmes assistidos.
- Restaurar a tabela ao estado anterior e limpar os recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da 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!;
        }
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for .NET .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da 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);
        }
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
/**
 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}")
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Kotlin.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
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.")
```

Defina uma classe que envolva ações de keyspace e tabelas.

```
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
```

Defina uma classe que crie uma conexão TLS com um keyspace, autentique com SigV4 e envie consultas CQL para uma tabela no keyspace.

```
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],
        )
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o Amazon Keyspaces com um SDK AWS. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

# Bibliotecas e ferramentas do Amazon Keyspaces (para Apache Cassandra)

Esta seção fornece informações sobre as bibliotecas, exemplos de código e ferramentas do Amazon Keyspaces (para Apache Cassandra.

Tópicos


- Bibliotecas e exemplos
- Repositórios de amostras e ferramentas para desenvolvedores em destaque


## Bibliotecas e exemplos

Você pode encontrar bibliotecas de código aberto e ferramentas para desenvolvedores do Amazon Keyspaces no GitHub nos repositórios de amostras AWS e AWS.

### Kit de ferramentas para desenvolvedores do Amazon Keyspaces (para Apache Cassandra)

Esse repositório fornece uma imagem do Docker com ferramentas úteis para desenvolvedores do Amazon Keyspaces. Por exemplo, ele inclui um arquivo CQLSHRC com as melhores práticas, uma expansão de autenticação AWS opcional para cqlsh e ferramentas auxiliares para realizar tarefas comuns. O kit de ferramentas é otimizado para o Amazon Keyspaces, mas também funciona com clusters do Apache Cassandra.

https://github.com/aws-samples/amazon-keyspaces-toolkit.

### Exemplos do Amazon Keyspaces (para Apache Cassandra)

Esse repositório é nossa lista oficial de exemplos de códigos do Amazon Keyspaces. O repositório é subdividido em seções por idioma (veja Exemplos). Cada idioma tem sua própria subseção de exemplos. Esses exemplos demonstram implementações e padrões comuns do serviço Amazon Keyspaces que você pode usar ao criar aplicativos.

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

## AWS Plugins de autenticação Signature Version 4 (SigV4)

Os plug-ins permitem que você gerencie o acesso ao Amazon Keyspaces usando usuários e funções 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.

Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

# Repositórios de amostras e ferramentas para desenvolvedores em destaque

Veja a seguir uma seleção de ferramentas comunitárias úteis do Amazon Keyspaces (para Apache Cassandra).

## Buffers de protocolo do Amazon Keyspaces

Você pode usar Buffers de Protocolo (Protobuf) com o Amazon Keyspaces para fornecer uma alternativa aos tipos definidos pelo usuário (UDTs) do Apache Cassandra. O Protobuf é um formato de dados multiplataforma gratuito e de código aberto usado para serializar dados estruturados. Você pode armazenar dados do Protobuf usando o tipo de dados CQL `BLOB` e refatorar UDTs, preservando dados estruturados em aplicativos e linguagens de programação.

Esse repositório fornece um exemplo de código que se conecta ao Amazon Keyspaces, cria uma nova tabela e insere uma linha contendo uma mensagem Protobuf. Em seguida, a linha é lida com forte consistência.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/protobuf-user-defined-types

## Modelo AWS CloudFormation para criar o painel do Amazon CloudWatch para as métricas do Amazon Keyspaces (para Apache Cassandra)

Esse repositório fornece modelos AWS CloudFormation para configurar rapidamente as métricas do CloudWatch para o Amazon Keyspaces. O uso desse modelo permitirá que você comece com mais

facilidade fornecendo painéis do CloudWatch pré-criados e implantáveis com métricas comumente usadas.

https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates.

## Como usar o Amazon Keyspaces (para Apache Cassandra) com AWS Lambda

O repositório contém exemplos que mostram como se conectar ao Amazon Keyspaces a partir do Lambda. Veja a seguir alguns exemplos:

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.

Outro exemplo do Lambda que mostra como implantar e usar o Amazon Keyspaces a partir de um Python Lambda está disponível no repositório a seguir.

https://github.com/aws-samples/aws-keyspaces-lambda-python

## Como usar o Amazon Keyspaces (para Apache Cassandra) com Spring

Este é um exemplo que mostra como usar o Amazon Keyspaces com o Spring Boot.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

## Como usar o Amazon Keyspaces (para Apache Cassandra) com Scala

Este é um exemplo que mostra como se conectar ao Amazon Keyspaces usando o plug-in de autenticação SigV4 com Scala.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/connection-sigv4

## Como usar o Amazon Keyspaces (para Apache Cassandra) com AWS Glue

Este é um exemplo que mostra como usar o Amazon Keyspaces com AWS Glue.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue

## Amazon Keyspaces (para Apache Cassandra) Cassandra Query Language (CQL) para conversor AWS CloudFormation

Esse pacote implementa uma ferramenta de linha de comando para converter scripts do Apache Cassandra Query Language (CQL) em modelos AWS CloudFormation (CloudFormation), o que permite que os esquemas do Amazon Keyspaces sejam facilmente gerenciados em pilhas do CloudFormation.

https://github.com/aws/amazon-keyspaces-cql-to-cfn-converter.

## Auxiliares do Amazon Keyspaces (para Apache Cassandra) para driver do Apache Cassandra para Java

Esse repositório contém políticas de drivers, exemplos e melhores práticas ao usar o driver Java DataStax com o Amazon Keyspaces (para Apache Cassandra).

https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

## Demonstração de compressão rápida do Amazon Keyspaces (para Apache Cassandra

Esse repositório demonstra como compactar, armazenar e ler/gravar objetos grandes para obter um desempenho mais rápido e reduzir os custos de throughput e armazenamento.

https://github.com/aws-samples/amazon-keyspaces-compression-example.

## Demonstração do codec do Amazon Keyspaces (para Apache Cassandra) e Amazon S3

O codec personalizado do Amazon S3 oferece suporte ao mapeamento transparente e configurável pelo usuário de ponteiros UUID para objetos do Amazon S3.

https://github.com/aws-samples/amazon-keyspaces-large-object-s3-demo.

# Integração do Amazon Keyspaces com o Apache Spark

O Apache Spark é um mecanismo de código aberto para análise de dados em grande escala. O Apache Spark permite que você realize análises em dados armazenados no Amazon Keyspaces com mais eficiência. Você também pode usar o Amazon Keyspaces para fornecer aos aplicativos acesso de leitura consistente, de um dígito e milissegundo, aos dados analíticos do Spark. O Spark Cassandra Connector de código aberto simplifica a leitura e gravação de dados entre o Amazon Keyspaces e o Spark.

O suporte do Amazon Keyspaces para o Spark Cassandra Connector simplifica a execução de workloads do Cassandra em pipelines de análise baseados em Spark usando um serviço de banco de dados totalmente gerenciado e de tecnologia sem servidor. Com o Amazon Keyspaces, você não precisa se preocupar com a concorrência do Spark pelos mesmos recursos de infraestrutura subjacente de suas tabelas. As tabelas do Amazon Keyspaces aumentam e diminuem a escala automaticamente com base no tráfego do seu aplicativo.

O tutorial a seguir mostra as etapas e as práticas recomendadas necessárias para ler e gravar dados no Amazon Keyspaces usando o Spark Cassandra Connector. O tutorial demonstra como migrar dados para o Amazon Keyspaces carregando dados de um arquivo com o Spark Cassandra Connector e gravando-os em uma tabela do Amazon Keyspaces. Em seguida, o tutorial mostra como ler os dados do Amazon Keyspaces usando o Spark Cassandra Connector. Você faria isso para executar workloads do Cassandra em pipelines de análise baseados em Spark.

Tópicos


- Pré-requisitos para estabelecer conexões com o Amazon Keyspaces usando o Spark Cassandra Connector
- Etapa 1: Configurar o Amazon Keyspaces para integração com o Apache Cassandra Spark Connector
- Etapa 2: Configurar o Apache Cassandra Spark Connector
- Etapa 3: Criar o arquivo de configuração do aplicativo
- Etapa 4: Preparar os dados de origem e a tabela de destino no Amazon Keyspaces
- Etapa 5: Gravar e ler dados do Amazon Keyspaces usando o Apache Cassandra Spark Connector
- Solução de problemas comuns ao usar o Spark Cassandra Connector com o Amazon Keyspaces

# Pré-requisitos para estabelecer conexões com o Amazon Keyspaces usando o Spark Cassandra Connector

Antes de se conectar ao Amazon Keyspaces com o Spark Cassandra Connector, você precisa ter certeza de que instalou o seguinte. A compatibilidade do Amazon Keyspaces com o Spark Cassandra Connector foi testada com as seguintes versões recomendadas:

- Java versão 8
- Scala 2.12
- Spark 3.4
- Cassandra Connector 2.5 e superior
- Cassandra driver 4.12

1. Para instalar Scala, siga as instruções em https://www.scala-lang.org/download/scala2.html.
2. Para instalar o Spark 3.4.1, siga este exemplo.

```
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
```

# Etapa 1: Configurar o Amazon Keyspaces para integração com o Apache Cassandra Spark Connector

Nesta etapa, você confirma que o particionador da sua conta é compatível com o Apache Spark Connector e configura as permissões necessárias do IAM. As práticas recomendadas a seguir ajudam você a provisionar capacidade suficiente de leitura/gravação para a tabela.

1. Confirme se o particionador `Murmur3Partitioner` é o particionador padrão da conta. Esse particionador é compatível com o Spark Cassandra Connector. Para obter mais informações

sobre particionadores e como alterá-los, consulte the section called “Como trabalhar com particionadores”.

2. Configure suas permissões do IAM para o Amazon Keyspaces, usando endpoints da VPC de interface, com o Apache Spark.

   - Atribua acesso de leitura/gravação à tabela de usuários e acesso de leitura às tabelas do sistema, conforme mostrado no exemplo de política do IAM listado abaixo.
   - É necessário preencher a tabela system.peers com seus endpoints da VPC de interface disponíveis para clientes que acessam o Amazon Keyspaces com o Spark por meio de endpoints da 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. Considere as seguintes práticas recomendadas para configurar capacidade de throughput de leitura/gravação suficiente para que sua tabela do Amazon Keyspaces ofereça suporte ao tráfego do Spark Cassandra Connector.

   - Comece a usar a capacidade sob demanda para ajudá-lo a testar o cenário.
   - Para otimizar o custo do throughput da tabela para ambientes de produção, use um limitador de taxa para o tráfego do conector e configure sua tabela para usar a capacidade provisionada com escalabilidade automática. Para obter mais informações, consulte the section called “Gerencie a capacidade de produção com escalabilidade automática”.
   - Você pode usar um limitador de taxa fixa que vem com o driver Cassandra. Há alguns limitadores de taxa personalizados para Amazon Keyspaces no repositório de amostras da AWS.
   - Para obter mais informações sobre gerenciamento de capacidade, consulte the section called “Modos de capacidade de leitura/gravação”.

# Etapa 2: Configurar o Apache Cassandra Spark Connector

O Apache Spark é uma plataforma de computação de uso geral que você pode configurar de diferentes maneiras. Para configurar o Spark e o Spark Cassandra Connector para integração com o Amazon Keyspaces, recomendamos que você comece com as configurações mínimas descritas na seção a seguir e depois aumente-as posteriormente, conforme apropriado para sua workload.

- Crie tamanhos de partição do Spark menores que 8 MBs.

  No Spark, as partições representam um bloco atômico de dados que pode ser executado paralelamente. Quando você está gravando dados no Amazon Keyspaces com o Spark Cassandra Connector, quanto menor a partição do Spark, menor a quantidade de registros que a tarefa gravará. Se uma tarefa do Spark encontrar vários erros, ela falhará depois que o número designado de novas tentativas for esgotado. Para evitar a repetição de tarefas grandes e o reprocessamento de muitos dados, mantenha o tamanho da partição do Spark pequeno.
- Use um número baixo de gravações simultâneas por executor com um grande número de novas tentativas.

  O Amazon Keyspaces retorna erros de capacidade insuficiente aos drivers do Cassandra como tempos limite de operação. Você não pode resolver os tempos limite causados pela capacidade insuficiente alterando a duração do tempo limite configurado porque o Spark Cassandra Connector tenta repetir as solicitações de forma transparente usando a `MultipleRetryPolicy`. Para

garantir que as novas tentativas não sobrecarreguem o pool de conexões do driver, use um número baixo de gravações simultâneas por executor com um grande número de novas tentativas. O código snippet a seguir é um exemplo disso.

```
spark.cassandra.query.retry.count = 500
spark.cassandra.output.concurrent.writes = 3
```

- Divida o throughput total e distribua-o em várias sessões do Cassandra.
  - O Cassandra Spark Connector cria uma sessão para cada executor do Spark. Pense nessa sessão como a unidade de escala para determinar o throughput necessário e o número de conexões necessárias.
  - Ao definir o número de núcleos por executor e o número de núcleos por tarefa, comece baixo e aumente conforme necessário.
  - Defina as falhas de tarefas do Spark para permitir o processamento em caso de erros transitórios. Depois de se familiarizar com as características e os requisitos de tráfego do seu aplicativo, recomendamos definir um valor limitado para `spark.task.maxFailures`.
  - Por exemplo, a configuração a seguir pode lidar com duas tarefas simultâneas por executor, por sessão:

    ```
    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
    ```

- Desative o processamento em lotes.
  - Recomendamos que você desative o processamento em lotes para melhorar os padrões de acesso aleatório. O código snippet a seguir é um exemplo disso.

    ```
    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)
    ```

- Defina **`SPARK_LOCAL_DIRS`** como um disco local rápido com espaço suficiente.
  - Por padrão, o Spark salva os arquivos de saída do mapa e os conjuntos de dados distribuídos resilientes (RDDs) em uma pasta `/tmp` . Dependendo da configuração do seu host Spark, isso pode resultar em erros do estilo falta de espaço no dispositivo.
  - Para definir a variável de ambiente `SPARK_LOCAL_DIRS` para um diretório chamado `/example/spark-dir`, você pode usar o comando a seguir.

```
export SPARK_LOCAL_DIRS=/example/spark-dir
```

# Etapa 3: Criar o arquivo de configuração do aplicativo

Para usar o Spark Cassandra Connector de código aberto com o Amazon Keyspaces, você precisa fornecer um arquivo de configuração do aplicativo que contenha as configurações necessárias para se conectar ao driver DataStax Java. Você pode usar as credenciais específicas do serviço ou o plug-in SigV4 para se conectar.

Se você ainda não tiver feito isso, você precisará converter o certificado digital Starfield em um arquivo trustStore. Você pode seguir as etapas detalhadas no tutorial de conexão do driver Java, the section called “Antes de começar”. Anote o caminho e a senha do arquivo trustStore porque você precisará dessa informação ao criar o arquivo de configuração do aplicativo.

## Conecte-se com autenticação SigV4

Esta seção mostra um arquivo de exemplo `application.conf` que você pode usar ao se conectar com credenciais da AWS e com o plug-in SigV4. Se ainda não tiver feito isso, você precisa gerar suas chaves de acesso do IAM (um ID de chave de acesso e uma chave de acesso secreta) e salvá-las em seu arquivo de configuração da AWS ou como variáveis de ambiente. Para obter instruções detalhadas, consulte the section called “Credenciais necessárias para autenticação AWS ”.

No exemplo a seguir, substitua o caminho do arquivo trustStore e substitua a senha.

```
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
}
```

Atualize e salve esse arquivo de configuração como `/home/user1/application.conf`. Os exemplos a seguir usam este caminho.

## Conecte-se com credenciais específicas do serviço

Esta seção mostra um arquivo `application.conf` de exemplo que você pode usar ao se conectar com credenciais específicas do serviço. Se você ainda não tiver feito isso, você precisará gerar credenciais específicas do serviço para Amazon Keyspaces. Para obter instruções detalhadas, consulte the section called “Credenciais específicas do serviço”.

No exemplo a seguir, substitua `username` e `password` pelas suas próprias credenciais. Além disso, substitua o caminho do arquivo trustStore e substitua a senha.

```
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
                }
            }
        }
}
```

Atualize e salve esse arquivo de configuração `/home/user1/application.conf` para usar com o exemplo de código.

## Conecte-se com uma taxa fixa

Para forçar uma taxa fixa por executor do Spark, você pode definir um limitador de solicitações. Esse limitador de solicitações limita a taxa de solicitações por segundo. O Spark Cassandra Connector implanta uma sessão do Cassandra por executor. Usar a fórmula a seguir pode ajudá-lo a obter um throughput consistente em relação a uma tabela.

```
max-request-per-second * numberOfExecutors = total throughput against a table
```

Você pode adicionar esse exemplo ao arquivo de configuração do aplicativo que você criou anteriormente.

```
datastax-java-driver {
  advanced.throttler {
    class = RateLimitingRequestThrottler

    max-requests-per-second = 3000
    max-queue-size = 30000
    drain-interval = 1 millisecond
  }
}
```

# Etapa 4: Preparar os dados de origem e a tabela de destino no Amazon Keyspaces

Nesta etapa, você criará um arquivo de origem com dados demonstrativos e uma tabela do Amazon Keyspaces.

1. Criar o arquivo de origem. Você pode escolher uma das seguintes opções:
   - Neste tutorial, você usará um arquivo de valores separados por vírgula (CSV) com o nome `keyspaces_sample_table.csv` como arquivo de origem para a migração de dados. O arquivo de amostra fornecido contém algumas linhas de dados de uma tabela com o nome `book_awards`.
     - Faça o download do arquivo CSV de amostra (`keyspaces_sample_table.csv`) que está contido no seguinte arquivo samplemigration.zip. Descompacte o arquivo e anote o caminho até `keyspaces_sample_table.csv`.
   - Se você quiser acompanhar seu próprio arquivo CSV para gravar dados no Amazon Keyspaces, certifique-se de que os dados sejam randomizados. Os dados lidos diretamente de um banco de dados ou exportados para arquivos simples geralmente são ordenados pela partição e pela chave primária. A importação de dados ordenados para o Amazon Keyspaces pode fazer com que eles sejam gravados em segmentos menores de partições do Amazon Keyspaces, o que resulta em uma distribuição de tráfego desigual. Isso pode causar um desempenho mais lento e a taxas de erro mais altas.

     Por outro lado, a randomização de dados ajuda a aproveitar os recursos integrados de balanceamento de carga do Amazon Keyspaces ao distribuir o tráfego entre partições de forma mais uniforme. Há várias ferramentas que você pode usar para randomizar dados. Para ver um exemplo que usa a ferramenta de código aberto Shuf, consulte o tutorial de migração de dados the section called “Etapa 2: preparar os dados”. Veja a seguir um exemplo que mostra como embaralhar dados como um `DataFrame`.

     ```
     import org.apache.spark.sql.functions.randval
     shuffledDF = dataframe.orderBy(rand())
     ```

2. Criar o espaço de chaves e a tabela de destino no Amazon Keyspaces.

   a. Conecte-se ao Amazon Keyspaces usando `cqlsh` e substitua o endpoint do serviço, o nome de usuário e a senha no exemplo a seguir por seus próprios valores.

```
cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Crie um novo espaço de chave com o nome `catalog` mostrado no exemplo a seguir.

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

c. Depois que o novo keyspace tiver o status de disponível, use o código a seguir para criar a tabela `book_awards` de destino. Para saber mais sobre a criação assíncrona de recursos e como verificar se um recurso está disponível, consulte the section called “Como criar espaços de chaves”.

```
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)
   );
```

## Etapa 5: Gravar e ler dados do Amazon Keyspaces usando o Apache Cassandra Spark Connector

Nesta etapa, você começa carregando os dados do arquivo de amostra em um `DataFrame` com o Spark Cassandra Connector. Em seguida, você grava os dados do `DataFrame` em sua tabela do Amazon Keyspaces. Você também pode usar essa parte de forma independente, por exemplo, para migrar dados para uma tabela do Amazon Keyspaces. Por fim, você lê os dados da tabela em um `DataFrame` usando o Spark Cassandra Connector. Você também pode usar essa parte de forma independente, por exemplo, para ler dados de uma tabela do Amazon Keyspaces para realizar análises de dados com o Apache Spark.

1. Inicie o Spark Shell conforme mostrado no exemplo a seguir. Observe que este exemplo usa a autenticação 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. Importe o Spark Cassandra Connector com o código a seguir.

```
import org.apache.spark.sql.cassandra._
```

3. Para ler dados do arquivo CSV e armazená-los em um `DataFrame`, você pode usar o exemplo de código a seguir.

```
var df =
 spark.read.option("header","true").option("inferSchema","true").csv("keyspaces_sample_tabl
```

É possível exibir o resultado com o seguinte comando.

```
scala> df.show();
```

A saída deve ser semelhante a esta.

```
+----------------+----+-----------+----+------------------+--------------------
+-------------+
|           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|
+----------------+----+-----------+----+------------------+--------------------
+-------------+
```

Você pode confirmar o esquema dos dados no `DataFrame`, conforme mostrado no exemplo a seguir.

```
scala> df.printSchema
```

A saída deve ser algo parecido com isso.

```
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. Use o comando a seguir para gravar os dados no `DataFrame` para a tabela do Amazon Keyspaces.

```
df.write.cassandraFormat("book_awards", "catalog").mode("APPEND").save()
```

5. Para confirmar que os dados foram salvos, você pode lê-los novamente em um quadro de dados, conforme mostrado no exemplo a seguir.

```
var newDf = spark.read.cassandraFormat("book_awards", "catalog").load()
```

Em seguida, você pode mostrar os dados que agora estão contidos no quadro de dados.

```
scala> newDf.show()
```

A saída do comando deve ser algo parecido com isso.

```
+--------------------+-----------------+---------------+----------+-------------
+----+----+
|          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|
+--------------------+-----------------+---------------+----------+-------------
+----+----+
```

# Solução de problemas comuns ao usar o Spark Cassandra Connector com o Amazon Keyspaces

Se você estiver usando o Amazon Virtual Private Cloud e se conectar ao Amazon Keyspaces, os erros mais comuns encontrados ao usar o conector Spark são causados pelos seguintes problemas de configuração.

- O usuário ou o perfil do IAM usado na VPC não tem as permissões necessárias para acessar a tabela `system.peers` no Amazon Keyspaces. Para obter mais informações, consulte the section called “Como preencher entradas da tabela `system.peers` com informações do endpoint da VPC de interface”.

- O usuário ou o perfil do IAM usado não tem as permissões necessárias de leitura/gravação para a tabela do usuário e acesso de leitura às tabelas de sistema no Amazon Keyspaces. Para obter mais informações, consulte the section called “Etapa 1: Configurar o Amazon Keyspaces”.
- A configuração do driver Java não desativa a verificação do nome do host ao criar a conexão SSL/TLS. Para ver exemplos, consulte the section called “Etapa 2: configurar o driver”.

Para ver as etapas detalhadas de solução de problemas de conexão, consulte the section called “Erros de conexão do endpoint da VPC”.

Além disso, você pode usar as métricas do Amazon CloudWatch para ajudá-lo a solucionar problemas na configuração do Spark Cassandra Connector no Amazon Keyspaces. Para saber mais sobre o uso do Amazon Keyspaces com CloudWatch, consulte the section called “Monitoramento com CloudWatch”.

A seção a seguir descreve as métricas mais úteis a serem observadas ao usar o Spark Cassandra Connector.

PerConnectionRequestRateExceeded

O Amazon Keyspaces tem uma cota de 3.000 solicitações por segundo por conexão. Cada executor do Spark estabelece uma conexão com o Amazon Keyspaces. Executar várias tentativas pode esgotar sua cota de taxa de solicitação por conexão. Se você exceder essa cota, o Amazon Keyspaces emite uma métrica `PerConnectionRequestRateExceeded` no CloudWatch.

Se notar eventos PerConnectionRequestRateExceeded junto com outros erros do sistema ou do usuário, é provável que o Spark esteja executando várias tentativas além do número alocado de solicitações por conexão.

Se você vir eventos `PerConnectionRequestRateExceeded` sem outros erros, talvez seja necessário aumentar o número de conexões nas configurações do driver para permitir mais throughput, ou talvez seja necessário aumentar o número de executores em seu trabalho do Spark.

StoragePartitionThroughputCapacityExceeded

O Amazon Keyspaces tem uma cota de 1.000 WCUs ou WRUs por segundo/3.000 RCUs ou RRUs por segundo, por partição. Se você estiver vendo eventos `StoragePartitionThroughputCapacityExceeded` do CloudWatch, isso pode indicar

que os dados não estão randomizados durante o carregamento. Para ver exemplos de como embaralhar dados, consulte the section called “Etapa 4: Preparar os dados de origem e a tabela de destino”.

## Erros e avisos comuns

Se você estiver usando a Amazon Virtual Private Cloud e se conectar ao Amazon Keyspaces, o driver do Cassandra poderá emitir uma mensagem de aviso sobre o próprio nó de controle na tabela `system.peers`. Para obter mais informações, consulte the section called “Erros e avisos comuns”. Você pode ignorar esse aviso com segurança.

# Solução de problemas do Amazon Keyspaces (para Apache Cassandra)

As seções a seguir fornecem informações sobre como solucionar problemas comuns de configuração que você pode encontrar ao usar o Amazon Keyspaces (para Apache Cassandra).

Para obter orientações de solução de problemas específicas para o acesso ao IAM, consulte the section called “Solução de problemas”.

Para obter mais informações sobre as melhores práticas de segurança, consulte the section called “Práticas recomendadas de segurança”.

Tópicos


- Solução de problemas gerais no Amazon Keyspaces
- Solução de problemas de conexões no Amazon Keyspaces
- Solução de problemas de gerenciamento de capacidade no Amazon Keyspaces
- Solução de problemas da linguagem de definição de dados no Amazon Keyspaces


## Solução de problemas gerais no Amazon Keyspaces

Recebendo erros gerais? Aqui estão alguns problemas comuns e como resolvê-los.

### Erros gerais

Você está recebendo uma das seguintes exceções de alto nível que podem ocorrer devido a vários motivos diferentes.

- `NoNodeAvailableException`
- `NoHostAvailableException`
- `AllNodesFailedException`

Essas exceções são geradas pelo driver do cliente e podem ocorrer quando você está estabelecendo a conexão de controle ou quando está realizando solicitações de leitura/gravação/preparação/execução/em lote.

Quando o erro ocorre enquanto você estabelece a conexão de controle, é um sinal de que todos os pontos de contato especificados em seu aplicativo estão inacessíveis. Quando o erro ocorre ao realizar consultas de leitura/gravação/preparação/execução, isso indica que todas as novas tentativas dessa solicitação foram esgotadas. Cada nova tentativa é tentada em um nó diferente quando você está usando a política de repetição padrão.

### Como isolar o erro subjacente das exceções do driver Java de nível superior

Esses erros gerais podem ser causados por problemas de conexão ou ao executar operações de leitura/gravação/preparação/execução. Falhas transitórias devem ser esperadas em sistemas distribuídos e devem ser tratadas repetindo a solicitação. O driver Java não tenta novamente automaticamente quando são encontrados erros de conexão, portanto, é recomendável implementar a política de repetição ao estabelecer a conexão do driver em seu aplicativo. Para obter uma visão geral detalhada das melhores práticas de conexão, consultethe section called “Conexões”.

Por padrão, o driver Java é definido como false `idempotence` para todas as solicitações, o que significa que o driver Java não repete automaticamente a solicitação com falha de leitura/gravação/preparação. `idempotence`Para configurar `true` e solicitar ao driver que repita solicitações com falha, você pode fazer isso de algumas maneiras diferentes. Aqui está um exemplo de como você pode definir a idempotência programaticamente para uma única solicitação em seu aplicativo Java.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");
s.setIdempotent(true);
```

Ou você pode definir a idempotência padrão para todo o seu aplicativo Java programaticamente, conforme mostrado no exemplo a seguir.

```
// 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
```

Outra recomendação é criar uma política de repetição no nível do aplicativo. Nesse caso, o aplicativo precisa capturar `NoNodeAvailableException` e repetir a solicitação. Recomendamos 10 novas tentativas com recuo exponencial começando em 10 ms e trabalhando até 100 ms com um tempo total de 1 segundo para todas as novas tentativas.

Outra opção é aplicar a política de repetição exponencial do Amazon Keyspaces ao estabelecer a conexão do driver Java disponível no Github.

Confirme se você estabeleceu conexões com mais de um nó ao usar a política de repetição padrão. Você pode fazer isso usando a seguinte consulta no Amazon Keyspaces.

```
SELECT * FROM system.peers;
```

Se a resposta para essa consulta estiver vazia, isso indica que você está trabalhando com um único nó para o Amazon Keyspaces. Se você estiver usando a política de repetição padrão, não haverá novas tentativas porque a nova tentativa padrão sempre ocorre em um nó diferente. Para saber mais sobre como estabelecer conexões em VPC endpoints, consulte. the section called “Conexões de endpoint da VPC”

Para ver um step-by-step tutorial que mostra como estabelecer uma conexão com o Amazon Keyspaces usando o driver Datastax 4.x Cassandra, consulte. the section called “Plugin de autenticação para Java 4.x”

# Solução de problemas de conexões no Amazon Keyspaces

Está tendo problemas para se conectar? Aqui estão alguns problemas comuns e como resolvê-los.

## Erros na conexão com um endpoint do Amazon Keyspaces

Falhas e erros de conexão podem resultar em mensagens de erro diferentes. A seção a seguir aborda os cenários mais comuns.

Tópicos


- Não consigo me conectar ao Amazon Keyspaces com o cqlsh
- Não consigo me conectar ao Amazon Keyspaces usando os drivers do cliente Cassandra


### Não consigo me conectar ao Amazon Keyspaces com o cqlsh

Você está tentando se conectar a um endpoint do Amazon Keyspaces usando cqlsh e a conexão falha com um **`Connection error`**.

Se você tentar se conectar a uma tabela do Amazon Keyspaces e o cqlsh não tiver sido configurado corretamente, a conexão falhará. A seção a seguir fornece exemplos dos problemas de configuração mais comuns que resultam em erros de conexão quando você tenta estabelecer uma conexão usando cqlsh.

> **Note**
>
> Se você estiver tentando se conectar ao Amazon Keyspaces a partir de uma VPC, serão necessárias permissões adicionais. Para configurar com êxito uma conexão usando endpoints da VPC, siga as etapas no the section called “Conexão com VPC endpoints”.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro de conexão **`timed out`**.

Esse pode ser o caso se você não tiver fornecido a porta correta, o que resultará no erro a seguir.

```
#  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")})
```

Para resolver esse problema, verifique se você está usando a porta 9142 para a conexão.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`Name or service not known`**.

Esse pode ser o caso se você usou um endpoint com ortografia incorreta ou que não existe. No exemplo a seguir, o nome do endpoint está escrito incorretamente.

```
#  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
```

Para resolver esse problema ao usar endpoints públicos para se conectar, selecione um endpoint disponível a partir de the section called “Service endpoints (Endpoints de serviço)” e verifique se o nome do endpoint não tem erros. Se você estiver usando endpoints da VPC para se conectar, verifique se as informações do endpoint da VPC estão corretas em sua configuração cqlsh.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`OperationTimedOut`**.

O Amazon Keyspaces exige que o SSL esteja habilitado para conexões para garantir uma segurança forte. O parâmetro SSL pode estar ausente se você receber o erro a seguir.

```
# 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',)})
#
```

Para resolver esse problema, adicione o seguinte sinalizador ao comando de conexão cqlsh.

```
--ssl
```

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh e recebe um erro **`SSL transport factory requires a valid certfile to be specified`**.

Nesse caso, o caminho para o certificado SSL/TLS está ausente, o que resulta no erro a seguir.

```
# 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.
#
```

Para resolver esse problema, adicione o caminho para o arquivo de certificado em seu computador.

```
certfile =  path_to_file/sf-class2-root.crt
```

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`No such file or directory`**.

Esse pode ser o caso se o caminho para o arquivo de certificado em seu computador estiver errado, o que resulta no erro a seguir.

```
# 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')})
#
```

Para resolver esse problema, verifique se o caminho para o arquivo de certificado em seu computador está correto.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`[X509] PEM lib`**.

Nesse caso, o caminho para o arquivo de certificado SSL/TLS `sf-class2-root.crt` é inválido, o que resulta no erro a seguir.

```
# 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)")})
#
```

Para resolver esse problema, baixe o certificado digital Starfield usando o comando a seguir. Salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro SSL **unknown**.

Nesse caso, o caminho para o arquivo de certificado SSL/TLS `sf-class2-root.crt` é vazio, o que resulta no erro a seguir.

```
# 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)")})
#
```

Para resolver esse problema, baixe o certificado digital Starfield usando o comando a seguir. Salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`SSL: CERTIFICATE_VERIFY_FAILED`**.

Esse pode ser o caso se o arquivo do certificado SSL/TLS não puder ser verificado, o que resulta no erro a seguir.

```
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)")})
```

Para resolver esse problema, baixe o certificado digital novamente usando o comando a seguir. Salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas você está recebendo um erro**`Last error: timed out`**.

Esse pode ser o caso se você não configurou uma regra de saída para o Amazon Keyspaces em seu grupo de segurança do Amazon EC2, o que resulta no erro a seguir.

```
# 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")})
#
```

Para confirmar se esse problema é causado pela configuração da instância do Amazon EC2 ou não`cqlsh`, você pode tentar se conectar ao seu keyspace usando o AWS CLI, por exemplo, com o comando a seguir.

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

Se esse comando também atingir o tempo limite, a instância do Amazon EC2 não está configurada corretamente.

Para confirmar que você tem permissões suficientes para acessar o Amazon Keyspaces, você pode usar o AWS CloudShell para se conectar. `cqlsh` Se essas conexões forem estabelecidas, você precisará configurar a instância do Amazon EC2.

Para resolver esse problema, confirme se sua instância do Amazon EC2 tem uma regra de saída que permite o tráfego para o Amazon Keyspaces. Se não for esse o caso, você precisa criar um novo grupo de segurança para a instância do EC2 e adicionar uma regra que permita tráfego de saída para os recursos do Amazon Keyspaces. Para atualizar a regra de saída para permitir o tráfego para o Amazon Keyspaces, escolha CQLSH/CASSANDRA no menu suspenso Tipo.

Depois de criar o novo grupo de segurança com a regra de tráfego de saída, você precisa adicioná-lo à instância. Selecione a instância e, em seguida, escolha Ações, depois Segurança e, em seguida, Alterar grupos de segurança. Adicione o novo grupo de segurança com a regra de saída, mas certifique-se de que o grupo padrão também permaneça disponível.

Para obter mais informações sobre como visualizar e editar regras de saída do EC2, consulte Adicionar regras a um grupo de segurança no Guia do usuário do Amazon EC2.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`Unauthorized`**.

Esse pode ser o caso se você não tiver permissões do Amazon Keyspaces na política de usuário do IAM, o que resulta no erro a seguir.

```
# 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."',)})
#
```

Para resolver esse problema, verifique se o usuário do IAM `testuser-at-12345678910` tem permissões para acessar o Amazon Keyspaces. Para obter exemplos de políticas do IAM que concedem acesso ao Amazon Keyspaces, consulte the section called “Exemplos de políticas baseadas em identidade”.

Para obter orientações de solução de problemas específicas para o acesso ao IAM, consulte the section called “Solução de problemas”.

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **`Bad credentials`**.

Esse pode ser o caso se o nome de usuário ou a senha estiverem incorretos, o que resultará no erro a seguir.

```
# 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"',)})
#
```

Para resolver esse problema, verifique se o *`NOME DE USUÁRIO`* e a *`SENHA`* em seu código correspondem ao nome de usuário e à senha que você obteve ao gerar credenciais específicas do serviço.

> **Important**
>
> Se você continuar vendo erros ao tentar se conectar com cqlsh, execute novamente o comando com a opção `--debug` e inclua a saída detalhada ao entrar em contato com AWS Support.

## Não consigo me conectar ao Amazon Keyspaces usando os drivers do cliente Cassandra

As seções a seguir mostram os erros mais comuns ao se conectar a um driver do cliente Cassandra.

Você está tentando se conectar a uma tabela do Amazon Keyspaces usando o driver DataStax Java, mas você recebe um **`NodeUnavailableException`** erro.

Se a conexão na qual a solicitação foi tentada for interrompida, isso resultará no seguinte erro.

```
[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)]
```

Para resolver esse problema, encontre o valor do batimento cardíaco e diminua-o para 30 segundos, se for maior.

```
advanced.heartbeat.interval = 30 seconds
```

Em seguida, procure o tempo limite associado e verifique se o valor está definido para pelo menos 5 segundos.

```
advanced.connection.init-query-timeout = 5 seconds
```

Você está tentando se conectar a uma tabela do Amazon Keyspaces usando um driver e o complemento SigV4, mas recebe um erro **`AttributeError`**.

Se as credenciais não estiverem configuradas corretamente, isso resultará no erro a seguir.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'44.234.22.154:9142': AttributeError("'NoneType' object has no attribute
 'access_key'")})
```

Para resolver esse problema, verifique se você está passando as credenciais associadas ao seu usuário ou perfil do IAM ao usar o complemento SigV4. O complemento SigV4 exige as seguintes credenciais.

- `AWS_ACCESS_KEY_ID`— Especifica uma chave de AWS acesso associada a um usuário ou função do IAM.
- `AWS_SECRET_ACCESS_KEY`: especifica a chave secreta associada à chave de acesso. Essencialmente, essa é a "senha" para a chave de acesso.

Para saber mais sobre as chaves de acesso e o complemento SigV4, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Você está tentando se conectar a uma tabela do Amazon Keyspaces usando um driver, mas recebe um erro **`PartialCredentialsError`**.

Se o `AWS_SECRET_ACCESS_KEY` estiver ausente, isso pode resultar no seguinte erro.

```
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’)})
```

Para resolver esse problema, verifique se você está transmitindo o `AWS_ACCESS_KEY_ID` e o `AWS_SECRET_ACCESS_KEY` ao usar o complemento SigV4. Para saber mais sobre as chaves de acesso e o complemento SigV4, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Você está tentando se conectar a uma tabela do Amazon Keyspaces usando um driver, mas recebe um erro **`Invalid signature`**.

Esse pode ser o caso se você usou credenciais erradas, o que resultará no seguinte erro.

```
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”’)})
```

Para resolver esse problema, verifique se as credenciais que você está passando estão associadas ao usuário ou perfil do IAM configurados para acessar o Amazon Keyspaces. Para saber mais sobre as chaves de acesso e o complemento SigV4, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Minha conexão de endpoint da VPC não funciona corretamente

Você está tentando se conectar ao Amazon Keyspaces usando endpoints da VPC, mas está recebendo erros no mapa de tokens ou está tendo baixo throughput.

Esse pode ser o caso se a conexão do endpoint da VPC não estiver configurada corretamente.

Para resolver esses problemas, verifique os detalhes de configuração a seguir. Para seguir um step-by-step tutorial para aprender como configurar uma conexão por meio de endpoints VPC de interface para Amazon Keyspaces, consulte. the section called “Conexão com VPC endpoints”

1. Confirme se a entidade IAM usada para se conectar ao Amazon Keyspaces tem acesso de leitura/gravação à tabela de usuários e acesso de leitura às tabelas do sistema, conforme mostrado no exemplo a seguir.

```
{
   "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. Confirme se a entidade IAM usada para se conectar ao Amazon Keyspaces tem as permissões de leitura necessárias para acessar as informações do endpoint da VPC na sua instância do Amazon EC2, conforme mostrado no exemplo a seguir.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
```

```
    ]
}
```

> **Note**
>
> As políticas gerenciadas `AmazonKeyspacesReadOnlyAccess_v2` e `AmazonKeyspacesFullAccess` incluem as permissões necessárias para permitir que o Amazon Keyspaces acesse a instância do Amazon EC2 para ler informações sobre os endpoints da VPC de interface disponíveis.

Para saber mais sobre endpoints da VPC, consulte the section called “Como usar o endpoint da VPC para o Amazon Keyspaces”

3. Confirme se a configuração SSL do driver Java define a validação do nome do host como falsa, conforme mostrado neste exemplo.

```
hostname-validation = false
```

Para obter mais informações sobre a configuração do driver, consulte the section called “Etapa 2: configurar o driver”.

4. Para confirmar se o endpoint da VPC foi configurado corretamente, você pode executar a instrução a seguir de dentro da sua VPC.

> **Note**
>
> Você não pode usar seu ambiente de desenvolvedor local ou o editor de CQL do Amazon Keyspaces para confirmar essa configuração, porque eles usam o endpoint público.

```
SELECT peer FROM system.peers;
```

A saída deve ser semelhante a este exemplo e retornar entre 2 a 6 nós com endereços IP privados, dependendo da configuração da VPC e AWS da região.

```
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)
```

Não consigo me conectar usando **`cassandra-stress`**

Você está tentando se conectar ao Amazon Keyspaces usando o comando **`cassandra-stress`**, mas está recebendo um erro **`SSL context`**.

Isso acontece se você tentar se conectar ao Amazon Keyspaces, mas não tiver a configuração correta do trustStore. O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes.

Nesse caso, você verá o erro a seguir.

```
Error creating the initializing the SSL Context
```

Para resolver esse problema, siga as instruções para configurar um trustStore conforme mostrado neste tópico the section called “Antes de começar”.

Depois que o trustStore estiver configurado, você deverá ser capaz de se conectar com o comando a seguir.

```
./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"
```

Não consigo me conectar usando identidades IAM

Você está tentando se conectar ao Amazon Keyspaces usando uma identidade IAM, mas está recebendo um erro **`Unauthorized`**.

Isso acontece se você tentar se conectar a uma tabela do Amazon Keyspaces usando uma identidade do IAM (por exemplo, um usuário do IAM) sem implementar a política e fornecer primeiro ao usuário as permissões necessárias.

Nesse caso, você verá o erro a seguir.

```
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."',)})
```

Para resolver esse problema, verifique as permissões do usuário do IAM. Para se conectar com um driver padrão, um usuário deve ter pelo menos acesso `SELECT` às tabelas do sistema, porque a maioria dos drivers lê os espaços de chaves/as tabelas do sistema quando estabelecem a conexão.

Para obter exemplos de políticas do IAM que concedem acesso ao sistema Amazon Keyspaces e às tabelas de usuários, consulte the section called “Como acessar as tabelas do Amazon Keyspaces”.

Para revisar a seção de solução de problemas específica do IAM, consulte the section called “Solução de problemas”.

Estou tentando importar dados com cqlsh e a conexão com minha tabela Amazon Keyspaces foi perdida

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas está recebendo erros de conexão.

A conexão com o Amazon Keyspaces falha depois que o cliente cqlsh recebe três erros consecutivos de qualquer tipo do servidor. O cliente cqlsh falha com a seguinte mensagem.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Para resolver esse erro, você precisa garantir que os dados a serem importados correspondam ao esquema da tabela no Amazon Keyspaces. Verifique se há erros de análise no arquivo de importação. Você pode tentar usar uma única linha de dados usando uma instrução INSERT para isolar o erro.

O cliente tenta automaticamente restabelecer a conexão.

# Solução de problemas de gerenciamento de capacidade no Amazon Keyspaces

Está tendo problemas com a capacidade de tecnologia sem servidor? Aqui estão alguns problemas comuns e como resolvê-los.

# Erros de capacidade de tecnologia sem servidor

Esta seção descreve como reconhecer erros relacionados ao gerenciamento de capacidade de tecnologia sem servidor e como resolvê-los. Por exemplo, você poderá observar eventos de capacidade insuficiente quando seu aplicativo exceder a capacidade de throughput provisionada.

Como o Apache Cassandra é um software baseado em cluster projetado para ser executado em uma frota de nós, ele não tem mensagens de exceção relacionadas a recursos de tecnologia sem servidor, como capacidade de throughput. A maioria dos drivers só entende os códigos de erro que estão disponíveis no Apache Cassandra, então o Amazon Keyspaces usa esse mesmo conjunto de códigos de erro para manter a compatibilidade.

Para mapear os erros do Cassandra para os eventos de capacidade subjacentes, você pode usar CloudWatch a Amazon para monitorar as métricas relevantes do Amazon Keyspaces. Eventos de capacidade insuficiente que resultam em erros do lado do cliente podem ser categorizados nesses três grupos com base no recurso que está causando o evento:

- Tabela: se você escolher o modo de capacidade Provisionada para uma tabela e seu aplicativo exceder o throughput provisionado, você poderá observar erros de capacidade insuficiente. Para ter mais informações, consulte the section called “Modos de capacidade de leitura/gravação”.
- Partição: você poderá enfrentar eventos de capacidade insuficiente se o tráfego em uma determinada partição exceder 3.000 RCUs ou 1.000 WCUs. Recomendamos distribuir o tráfego uniformemente entre partições como uma prática recomendada. Para ter mais informações, consulte the section called “Modelagem de dados”.
- Conexão: você pode ter um throughput insuficiente se exceder a cota para o número máximo de operações por segundo, por conexão. Para aumentar o throughput, você pode aumentar o número de conexões padrão ao configurar a conexão com o driver.

  Para saber como configurar conexões para o Amazon Keyspaces, consulte. the section called “Como configurar conexões” Para obter mais informações sobre como otimizar conexões em VPC endpoints, consulte. the section called “Conexões de endpoint da VPC”

Para determinar qual recurso está causando o evento de capacidade insuficiente que está retornando o erro do lado do cliente, você pode verificar o painel no console do Amazon Keyspaces. Por padrão, o console fornece uma visão agregada das CloudWatch métricas mais comuns relacionadas à capacidade e ao tráfego na seção Capacidade e métricas relacionadas na guia Capacidade da tabela.

Para criar seu próprio painel usando a Amazon CloudWatch, verifique as seguintes métricas do Amazon Keyspaces.

- `PerConnectionRequestRateExceeded`: solicitações para o Amazon Keyspaces que excedam a cota de taxa de solicitação por conexão. Cada conexão do cliente com o Amazon Keyspaces pode suportar até 3 mil solicitações de CQL por segundo. Você pode realizar mais de 3 mil solicitações por segundo criando várias conexões.
- `ReadThrottleEvents`: solicitações para Amazon Keyspaces que excedem a capacidade de leitura de uma tabela.
- `StoragePartitionThroughputCapacityExceeded`: solicitações para uma partição de armazenamento do Amazon Keyspaces que excedam a capacidade de throughput da partição. As partições de armazenamento do Amazon Keyspaces podem suportar até 1.000 WCU/WRU por segundo e 3000 RCU/RRU por segundo. Para mitigar estas exceções, recomendamos que reveja o seu modelo de dados para distribuir o tráfego de leitura/gravação em mais divisórias.
- `WriteThrottleEvents`: solicitações para Amazon Keyspaces que excedem a capacidade de gravação de uma tabela.

Para saber mais sobre CloudWatch, consultethe section called “Monitoramento com CloudWatch”. Para obter uma lista de todas as CloudWatch métricas disponíveis para o Amazon Keyspaces, consulte. the section called “Métricas e dimensões”

> **Note**
>
> Para começar com um painel personalizado que mostra todas as métricas comumente observadas para o Amazon Keyspaces, você pode usar um CloudWatch modelo pré-criado disponível GitHub no AWS repositório de amostras.

Tópicos

- Estou recebendo erros de capacidade insuficientes NoHostAvailable do driver do meu cliente
- Estou recebendo erros de tempo limite de gravação durante a importação de dados
- Não consigo ver o tamanho real do armazenamento de um espaço de chaves ou uma tabela

## Estou recebendo erros de capacidade insuficientes **`NoHostAvailable`** do driver do meu cliente

Você está vendo exceções **`Read_Timeout`** ou **`Write_Timeout`** para uma tabela.

Tentar repetidamente gravar ou ler uma tabela do Amazon Keyspaces com capacidade insuficiente pode resultar em erros do lado do cliente que são específicos do driver.

Use CloudWatch para monitorar suas métricas de taxa de transferência provisionadas e reais e eventos de capacidade insuficientes para a tabela. Por exemplo, uma solicitação de leitura que não tem capacidade de throughput suficiente falha com uma exceção `Read_Timeout` e é publicada na métrica `ReadThrottleEvents`. Uma solicitação de gravação que não tem capacidade de throughput suficiente falha com uma exceção `Write_Timeout` e é publicada na métrica `WriteThrottleEvents`. Para ter mais informações sobre essas métricas, consulte the section called “Métricas e dimensões”.

Para resolver esses problemas, considere uma das opções a seguir.

- Aumente o throughput provisionado para a tabela, que é a quantidade máxima de capacidade de throughput que um aplicativo pode consumir. Para ter mais informações, consulte the section called “Unidades de capacidade de leitura e unidades de capacidade de gravação”.
- Permita que o serviço gerencie a capacidade de throughput em seu nome com o escalonamento automático. Para ter mais informações, consulte the section called “Gerencie a capacidade de produção com escalabilidade automática”.
- Escolha o modo de capacidade sob demanda para a tabela. Para ter mais informações, consulte the section called “Modo de capacidade sob demanda”.

Se você precisar aumentar a cota de capacidade padrão da sua conta, consulte Cotas.

Você está vendo erros relacionados à capacidade excedida da partição.

Quando você vê o erro, a capacidade `StoragePartitionThroughputCapacityExceeded` da partição é temporariamente excedida. Isso pode ser tratado automaticamente pela capacidade adaptativa ou pela capacidade sob demanda. Recomendamos revisar seu modelo de dados para distribuir o tráfego de leitura/gravação em mais partições para mitigar esses erros. As partições de armazenamento do Amazon Keyspaces podem suportar até 1.000 WCU/WRU por segundo e 3000 RCU/RRU por segundo. Para saber mais sobre como melhorar seu modelo de dados para distribuir o tráfego de leitura/gravação em mais partições, consulte the section called “Modelagem de dados”.

Exceções `Write_Timeout` também podem ser causadas por uma taxa elevada de operações de gravação simultâneas que incluem dados estáticos e não estáticos na mesma partição lógica. Se é esperado que o tráfego execute várias operações de gravação simultâneas que incluam dados estáticos e não estáticos na mesma partição lógica, recomendamos gravar dados estáticos e não estáticos separadamente. Gravar os dados separadamente também ajuda a otimizar os custos de throughput.

Você está vendo erros relacionados à taxa de solicitação de conexão excedida.

Você está vendo `PerConnectionRequestRateExceeded` devido a uma das seguintes causas.

- Talvez você não tenha conexões suficientes configuradas por sessão.
- Talvez você esteja obtendo menos conexões do que os pares disponíveis, porque não tem as permissões do endpoint da VPC configuradas corretamente. Para obter mais informações sobre as políticas de endpoint da VPC, consulte the section called “Como usar o endpoint da VPC para o Amazon Keyspaces”.
- Se você estiver usando um driver 4.x, verifique se a validação do nome do host está ativada. O driver habilita a verificação do nome do host TLS por padrão. Essa configuração faz com que o Amazon Keyspaces apareça como um cluster de nó único para o driver. Recomendamos que você desative a verificação do nome de host.

Recomendamos que você siga estas práticas recomendadas para garantir que suas conexões e seu throughput sejam otimizados:

- Configure o ajuste do throughput de consultas CQL.

  O Amazon Keyspaces suporta até 3.000 consultas CQL por conexão TCP por segundo, mas não há limite no número de conexões que um driver pode estabelecer.

  A maioria dos drivers de código aberto do Cassandra estabelece um pool de conexões com o Cassandra e balanceia a carga das consultas sobre esse pool de conexões. O Amazon Keyspaces expõe 9 endereços IP emparelhados aos drivers. O comportamento padrão da maioria dos drivers é estabelecer uma única conexão com cada endereço IP emparelhado. Portanto, o throughput máximo de consultas CQL de um driver usando as configurações padrão será de 27.000 consultas CQL por segundo.

  Para aumentar esse número, recomendamos aumentar o número de conexões por endereço IP que seu driver mantém no grupo de conexões. Por exemplo, definir o máximo de conexões por

endereço IP como 2 dobrará o throughput máximo do seu driver para 54.000 consultas CQL por segundo.

- Otimize suas conexões de nó único.

  Por padrão, a maioria dos drivers Cassandra de código aberto estabelece uma ou mais conexões com cada endereço IP anunciado na tabela `system.peers` ao estabelecer uma sessão. No entanto, certas configurações podem fazer com que um driver se conecte a um único endereço IP do Amazon Keyspaces. Isso pode acontecer se o driver estiver tentando validar o nome de host SSL dos nós pares (por exemplo, drivers DataStax Java) ou quando estiver se conectando por meio de um VPC endpoint.

  Para obter a mesma disponibilidade e desempenho de um driver com conexões para vários endereços IP, recomendamos fazer o seguinte:

  - Aumente o número de conexões por IP para 9 ou mais, dependendo do throughput desejado do cliente.
  - Crie uma política de repetição personalizada que garanta que as novas tentativas sejam executadas no mesmo nó.
  - Se você usa endpoints da VPC, conceda à entidade IAM que é usada para se conectar ao Amazon Keyspaces permissões de acesso para consultar sua VPC para obter informações sobre o endpoint e a interface de rede. Isso melhora o balanceamento de carga e aumenta o throughput de leitura/gravação. Para ter mais informações, consulte ???.

## Estou recebendo erros de tempo limite de gravação durante a importação de dados

Você está recebendo um erro de tempo limite ao carregar dados usando o comando **`cqlsh COPY`**.

```
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
```

O Amazon Keyspaces usa as exceções `ReadTimeout` e `WriteTimeout` para indicar quando uma solicitação de gravação falha devido à capacidade de throughput insuficiente. Para ajudar a diagnosticar exceções de capacidade insuficiente, o Amazon Keyspaces publica as seguintes métricas na Amazon. CloudWatch

- `WriteThrottleEvents`

- `ReadThrottledEvents`
- `StoragePartitionThroughputCapacityExceeded`

Para resolver erros de capacidade insuficiente durante um carregamento de dados, diminua a taxa de gravação por trabalhador ou a taxa total de ingestão e, em seguida, tente carregar as linhas novamente. Para ter mais informações, consulte the section called “Etapa 4: Definir configurações de `cqlsh COPY FROM`”. Para uma opção de upload de dados mais robusta, considere usar o DSBulk, que está disponível no GitHub repositório. Para step-by-step obter instruções, consultethe section called “Carregamento de dados usando o DSBulk”.

### Não consigo ver o tamanho real do armazenamento de um espaço de chaves ou uma tabela

Você não consegue ver o tamanho real de armazenamento de um espaço de chaves ou uma tabela.

Para saber mais sobre o tamanho de armazenamento da sua mesa, consultethe section called “Avaliar seus custos no nível da tabela”. Você também pode estimar o tamanho do armazenamento começando a calcular o tamanho da linha em uma tabela. Instruções detalhadas para calcular o tamanho da linha estão disponíveis emthe section called “Como calcular o tamanho da linha”.

## Solução de problemas da linguagem de definição de dados no Amazon Keyspaces

Está tendo problemas para criar recursos? Aqui estão alguns problemas comuns e como resolvê-los.

### Erros de linguagem de definição de dados

O Amazon Keyspaces executa operações de linguagem de definição de dados (DDL) de forma assíncrona, por exemplo, criando e excluindo espaços de chaves e tabelas. Se um aplicativo estiver tentando usar o recurso antes de estar pronto, a operação falhará.

Você pode monitorar o status de criação de novos espaços de chave e tabelas no AWS Management Console, o que indica quando um espaço de teclas ou tabela está pendente ou ativo. Você também pode monitorar programaticamente o status de criação de um novo espaço de chaves ou tabela consultando a tabela do esquema do sistema. Um espaço de chaves ou tabela fica visível no esquema do sistema quando está pronto para uso.

> **Note**
>
> Para otimizar a criação de espaços de chave usando AWS CloudFormation, você pode usar esse utilitário para converter scripts CQL em CloudFormation modelos. A ferramenta está disponível no GitHub repositório.

Tópicos


- Eu criei um espaço de chaves, mas não consigo visualizá-lo nem o acessar
- Eu criei uma nova tabela, mas não consigo visualizá-la nem a acessar
- Estou tentando restaurar uma tabela usando a point-in-time recuperação do Amazon Keyspaces (PITR), mas a restauração falha
- Estou tentando usar INSERIR/ATUALIZAR para editar configurações personalizadas de vida útil (TTL), mas a operação falha
- Estou tentando fazer o carregamento de dados para minha tabela do Amazon Keyspaces e recebo um erro sobre exceder o número de colunas
- Estou tentando excluir dados na minha tabela do Amazon Keyspaces e a exclusão falha no intervalo


## Eu criei um espaço de chaves, mas não consigo visualizá-lo nem o acessar

Você está recebendo erros do seu aplicativo que está tentando acessar um novo espaço de chaves.

Se você tentar acessar um espaço de chaves recém-criado do Amazon Keyspaces que ainda está sendo criado de forma assíncrona, você receberá um erro. Veja a seguir um exemplo de log de erros.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 keyspace mykeyspace"
```

O padrão de design recomendado para verificar quando um novo espaço de chaves está pronto para uso é pesquisar as tabelas de esquema do sistema do Amazon Keyspaces (system_schema_mcs.*).

Para ter mais informações, consulte the section called “Como criar espaços de chaves”.

## Eu criei uma nova tabela, mas não consigo visualizá-la nem a acessar

Você está recebendo erros do seu aplicativo que está tentando acessar uma nova tabela.

Se você tentar acessar uma tabela recém-criada do Amazon Keyspaces que ainda está sendo criado de forma assíncrona, você receberá um erro. Por exemplo, tentar consultar uma tabela que ainda não está disponível falha com um erro `unconfigured table`.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 table mykeyspace.mytable"
```

Tentando ver a tabela com falhas `sync_table()` com um `KeyError`.

```
KeyError: 'mytable'
```

O padrão de design recomendado para verificar quando uma nova tabela está pronta para uso é pesquisar as tabelas de esquema do sistema do Amazon Keyspaces (system_schema_mcs.*).

Esse é o exemplo de saída de uma tabela que está sendo criada.

```
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)
```

Esse é o exemplo de saída de uma tabela que está ativa.

```
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)
```

Para ter mais informações, consulte the section called “Criar tabelas”.

## Estou tentando restaurar uma tabela usando a point-in-time recuperação do Amazon Keyspaces (PITR), mas a restauração falha

Se você está tentando restaurar uma tabela do Amazon Keyspaces com point-in-time recuperação (PITR) e vê o processo de restauração começar, mas não ser concluído com sucesso, talvez você não tenha configurado todas as permissões necessárias para o processo de restauração dessa tabela específica.

Além das permissões de usuário, o Amazon Keyspaces pode exigir permissões para realizar ações durante o processo de restauração em nome da sua entidade principal. Esse é o caso se a tabela for criptografada com uma chave gerenciada pelo cliente ou se você estiver usando políticas do IAM que restringem o tráfego de entrada.

Por exemplo, se você estiver usando chaves de condição em sua política do IAM para restringir o tráfego de origem a endpoints ou intervalos de IP específicos, a operação de restauração falhará. Para permitir que o Amazon Keyspaces execute a operação de restauração da tabela em nome da sua entidade principal, você deve adicionar uma chave de condição `aws:ViaAWSService` global na política do IAM.

Para obter mais informações sobre as permissões para restaurar tabelas, consulte the section called “Restaurar permissões”.

## Estou tentando usar INSERIR/ATUALIZAR para editar configurações personalizadas de vida útil (TTL), mas a operação falha

Se você estiver tentando inserir ou atualizar um valor de TTL personalizado, a operação poderá falhar com o seguinte erro.

```
TTL is not yet supported.
```

Para especificar valores TTL personalizados para linhas ou colunas as operações `INSERT` ou `UPDATE`, você deve primeiro habilitar a TTL para a tabela. Você pode habilitar a TTL para uma tabela usando a propriedade personalizada `ttl`.

Para obter mais informações sobre como habilitar configurações TTL personalizadas para tabelas, consulte the section called “Como habilitar a vida útil (TTL) em tabelas existentes usando propriedades personalizadas”.

## Estou tentando fazer o carregamento de dados para minha tabela do Amazon Keyspaces e recebo um erro sobre exceder o número de colunas

Você está carregando dados e excedeu o número de colunas que podem ser atualizadas simultaneamente.

Esse erro ocorre quando o esquema da sua tabela excede o tamanho máximo de 350 KB. Para ter mais informações, consulte Cotas.

## Estou tentando excluir dados na minha tabela do Amazon Keyspaces e a exclusão falha no intervalo

Você está tentando excluir dados por chave de partição e recebe um erro de exclusão de intervalo.

Esse erro ocorre quando você está tentando excluir mais de 1.000 linhas em uma operação de exclusão.

```
Range delete requests are limited by the amount of items that can be deleted in a
 single range.
```

Para ter mais informações, consulte the section called “Intervalo de exclusão”.

Para excluir mais de 1.000 linhas em uma única partição, considere as opções a seguir.

- Excluir por partição – Se a maioria das partições tiver menos de 1.000 linhas, você poderá tentar excluir dados por partição. Se as partições contiverem mais de 1.000 linhas, tente excluí-las usando a coluna de clustering.
- Excluir por coluna de clustering – Se seu modelo contiver várias colunas de clustering, você poderá usar a hierarquia de colunas para excluir várias linhas. As colunas de clustering são uma estrutura aninhada, e você pode excluir muitas linhas operando na coluna de nível superior.
- Excluir por linha individual – Você pode iterar pelas linhas e excluir cada linha usando sua chave primária completa (colunas de partição e colunas de clustering).
- Como melhor prática, considere dividir suas linhas em partições — No Amazon Keyspaces, recomendamos que você distribua seu throughput entre as partições da tabela. Isso distribui os dados e o acesso uniformemente entre os recursos físicos, o que proporciona melhor throughput. Para ter mais informações, consulte the section called “Modelagem de dados”.

Considere também as recomendações a seguir ao planejar operações de exclusão para workload pesadas.

- Com o Amazon Keyspaces, as partições podem conter um número praticamente ilimitado de linhas. Isso permite que você escale partições de forma “mais ampla” do que a orientação tradicional do Cassandra de 100 MB. Não é incomum que séries temporais ou livros contábeis cresçam mais de um gigabyte de dados ao longo do tempo.
- Com o Amazon Keyspaces, não há estratégias de compactação ou marcas de exclusão a serem consideradas quando você precisa realizar operações de exclusão para workloads pesadas. Você pode excluir quantos dados quiser sem afetar o desempenho de leitura.

# Gerenciamento de recursos de tecnologia sem servidor no Amazon Keyspaces (para Apache Cassandra)

O Amazon Keyspaces (para Apache Cassandra) é uma tecnologia sem servidor. Em vez de implantar, gerenciar e manter recursos de armazenamento e computação para sua workload por meio de nós em um cluster, o Amazon Keyspaces aloca recursos de armazenamento e throughput de leitura/gravação diretamente às tabelas.

Este capítulo fornece detalhes sobre o gerenciamento de recursos de tecnologia sem servidor no Amazon Keyspaces. Para saber como monitorar recursos sem servidor com a Amazon CloudWatch, consulte. the section called “Monitoramento com CloudWatch”

Tópicos


- Armazenamento no Amazon Keyspaces
- Modos de capacidade de leitura/gravação no Amazon Keyspaces
- Gerencie a capacidade de processamento automaticamente com o escalonamento automático do Amazon Keyspaces
- Usando a capacidade de intermitência de forma eficaz no Amazon Keyspaces
- Como estimar o consumo de capacidade no Amazon Keyspaces


## Armazenamento no Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) provisiona o armazenamento para tabelas automaticamente com base nos dados reais armazenados em sua tabela. Você não precisa provisionar o armazenamento para tabelas com antecedência. O Amazon Keyspaces aumenta e diminui automaticamente o armazenamento de tabelas à medida que seu aplicativo grava, atualiza e exclui dados. Diferentemente dos clusters Apache Cassandra tradicionais, o Amazon Keyspaces não exige armazenamento adicional para suportar operações de sistema de baixo nível, como compactação. Você paga apenas pelo armazenamento que usar.

O Amazon Keyspaces configura espaços de chaves com um fator de replicação de três por padrão. Você não pode modificar o fator de replicação. O Amazon Keyspaces replica os dados da tabela três vezes automaticamente em várias zonas de disponibilidade para obter alta AWS disponibilidade. O preço por GB do armazenamento do Amazon Keyspaces já inclui a replicação. Consulte os preços do Amazon Keyspaces (para Apache Cassandra) para obter mais informações.

O Amazon Keyspaces monitora continuamente o tamanho de suas tabelas para determinar suas cobranças de armazenamento. Para obter mais informações sobre como o Amazon Keyspaces calcula o tamanho faturável dos dados, consulte the section called “Como calcular o tamanho da linha”.

# Modos de capacidade de leitura/gravação no Amazon Keyspaces

O Amazon Keyspaces oferece dois modos de capacidade de leitura/gravação para processar leituras e gravações em suas tabelas:

- Sob demanda (padrão)
- Provisionada

O modo de capacidade de leitura/gravação controla como você é cobrado por throughput de leitura e gravação e como a capacidade de throughput da tabela é gerenciada.

Tópicos


- Modo de capacidade sob demanda
- Modo de capacidade de throughput provisionada
- Gerenciamento e visualização de modos de capacidade
- Considerações ao mudar os modos de capacidade


## Modo de capacidade sob demanda

O modo de capacidade sob demanda do Amazon Keyspaces (para Apache Cassandra) é uma opção de faturamento flexível capaz de servir centenas de solicitações por segundo sem planejamento de capacidade. Essa opção oferece pay-per-request preços para solicitações de leitura e gravação, de forma que você pague somente pelo que usar.

Quando você seleciona o modo sob demanda, o Amazon Keyspaces pode escalar instantaneamente a capacidade de throughput de sua tabela para qualquer nível de tráfego previamente registrado e, em seguida, voltar para baixo quando o tráfego do aplicativo diminuir. Se o nível de tráfego de uma workload atingir um novo pico, o serviço se adapta rapidamente para aumentar a capacidade de throughput de sua tabela. Você pode ativar o modo de capacidade sob demanda para tabelas novas e existentes.

Modo sob demanda é uma boa opção se qualquer uma das declarações a seguir for verdadeira:

- Você cria novas tabelas com workloads desconhecidas.
- Você tem tráfego de aplicativos imprevisível.
- Você prefere a facilidade de pagar somente pelo que usar.

Para começar com o modo sob demanda, você pode criar uma nova tabela ou atualizar uma tabela existente para usar o modo de capacidade sob demanda usando o console ou com algumas linhas do código Cassandra Query Language (CQL). Para ter mais informações, consulte the section called “Tabelas”.

Tópicos


- Unidades de solicitação de leitura e unidades de solicitação de gravação
- Tráfego de pico e propriedades de dimensionamento
- Throughput inicial para modo de capacidade sob demanda


## Unidades de solicitação de leitura e unidades de solicitação de gravação

Com tabelas de modo de capacidade sob demanda, você não precisa especificar antecipadamente quanto de throughput de leitura e gravação espera que seu aplicativo use. O Amazon Keyspaces cobra pelas leituras e gravações realizadas em suas tabelas em termos de unidades de solicitação de leitura (RRUs) e unidades de solicitação de gravação (WRUs).

- Um RRU representa uma solicitação de leitura `LOCAL_QUORUM` ou duas solicitações de leitura `LOCAL_ONE` para uma linha com até 4 KB de tamanho. Se você precisar ler uma linha maior que 4 KB, a operação de leitura usará RRUs adicionais. O número total de RRUs necessários varia de acordo com o tamanho da linha e se você deseja usar consistência de leitura `LOCAL_QUORUM` ou `LOCAL_ONE`. Por exemplo, a leitura de uma linha de 8 KB exige 2 RRUs usando consistência de leitura `LOCAL_QUORUM` e 1 RRU se você selecionar consistência de leitura `LOCAL_ONE`.
- Um WRU representa uma gravação para uma linha com até 1 KB de tamanho. Todas as gravações estão usando consistência `LOCAL_QUORUM` e não há cobrança adicional pelo uso de transações leves (LWTs). Se você precisar gravar uma linha maior que 1 KB, a operação de gravação usará WRUs adicionais. O número total de WRUs necessários depende do tamanho da linha. Por exemplo, se o tamanho da sua linha for 2 KB, você precisa de 2 WRUs para realizar uma solicitação de gravação.

Para obter informações sobre os níveis de consistência suportados, consulte the section called "Níveis de consistência suportados do Cassandra".

## Tráfego de pico e propriedades de dimensionamento

As tabelas do Amazon Keyspaces que usam modo de capacidade sob demanda automaticamente adaptam-se ao volume de tráfego da sua aplicação. O modo de capacidade sob demanda acomoda instantaneamente até o dobro do pico de tráfego anterior em uma tabela. Por exemplo, o padrão de tráfego do seu aplicativo pode variar entre 5 mil e 10 mil leituras `LOCAL_QUORUM` por segundo, sendo 10 mil leituras por segundo o pico de tráfego anterior.

Com esse padrão, o modo de capacidade sob demanda acomoda instantaneamente o tráfego sustentado de até 20 mil leituras por segundo. Se a sua aplicação sustentar o tráfego de 20 mil leituras por segundo, esse pico torna-se o novo pico anterior, habilitando o tráfego subsequente de até 40 mil leituras por segundo.

Se você precisar de mais que o dobro do pico anterior em uma tabela, o Amazon Keyspaces alocará automaticamente mais capacidade à medida que o volume de tráfego aumentar. Isso ajuda a garantir que sua tabela tenha capacidade de throughput suficiente para processar as solicitações adicionais. No entanto, você pode observar erros de capacidade de throughput insuficientes se exceder o dobro do pico anterior em 30 minutos.

Por exemplo, suponha que o padrão de tráfego do seu aplicativo varie entre 5 mil e 10 mil leituras altamente consistentes por segundo, onde 20 mil leituras por segundo é o pico de tráfego atingido anteriormente. Nesse caso, o serviço recomenda que você espace o crescimento do tráfego em pelo menos 30 minutos antes de gerar até 40 mil leituras por segundo.

Para saber como estimar o consumo da capacidade de leitura e gravação de uma tabela, consultethe section called "Estime o consumo de capacidade".

Para saber mais sobre as cotas padrão da sua conta e como aumentá-las, consulte Cotas.

## Throughput inicial para modo de capacidade sob demanda

Se você criar uma nova tabela com o modo de capacidade sob demanda habilitado ou alternar uma tabela existente para o modo de capacidade sob demanda pela primeira vez, a tabela terá as configurações de pico anteriores a seguir, mesmo que não tenha servido tráfego anteriormente usando o modo de capacidade sob demanda:

- Tabela recém-criada com modo de capacidade sob demanda: o pico anterior era de 2 mil WRUs e 6 mil RRUs. Você pode originar até o dobro do pico anterior imediatamente. Isso possibilita que tabelas sob demanda criadas recentemente atendam até 4 mil WRUs e 12 mil RRUs.
- Tabela existente alternada para o modo de capacidade sob demanda: o pico anterior é metade das WCUs e RCUs anteriores provisionadas para a tabela ou as configurações de uma tabela recém-criada com modo de capacidade sob demanda, o que for maior.

## Modo de capacidade de throughput provisionada

Se você selecionar o modo de capacidade de throughput provisionada, especifique o número de leituras e gravações por segundo necessárias para seu aplicativo. Isso o ajuda a gerenciar seu uso do Amazon Keyspaces para permanecer em ou abaixo de uma taxa de solicitação definida para otimizar o preço e manter a previsibilidade. Para saber mais sobre escalabilidade automática para throughput provisionada, consulte the section called “Gerencie a capacidade de produção com escalabilidade automática”.

Modo de capacidade de throughput provisionada é uma boa opção se qualquer uma das declarações a seguir for verdadeira:

- Você tem tráfego de aplicativos previsível.
- Você executa aplicativos cujo tráfego é consistente ou aumenta gradualmente.
- Você pode prever os requisitos de capacidade para otimizar o preço.

### Unidades de capacidade de leitura e unidades de capacidade de gravação

Para tabelas de modo de capacidade de throughput provisionada, você especifica a capacidade de throughput em termos de unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs):

- Uma RCU representa uma leitura `LOCAL_QUORUM` por segundo, ou duas leituras por `LOCAL_ONE` segundo, para uma linha com até 4 KB de tamanho. Se você precisar ler uma linha maior que 4 KB, a operação de leitura usará RCUs adicionais.

  O número total de RCUs necessárias varia de acordo com o tamanho da linha e se você deseja leituras `LOCAL_QUORUM` ou `LOCAL_ONE`. Por exemplo, se o tamanho da sua linha for 8 KB, você precisa de 2 RCUs para sustentar uma leitura `LOCAL_QUORUM` por segundo e 1 RCU se você selecionar leituras `LOCAL_ONE`.

- Um WCU representa uma gravação por segundo para uma linha com até 1 KB de tamanho. Todas as gravações estão usando consistência `LOCAL_QUORUM` e não há cobrança adicional pelo uso de transações leves (LWTs). Se você precisar gravar uma linha maior que 1 KB, a operação de gravação usará WCUs adicionais.

  O número total de WCUs necessárias depende do tamanho da linha. Por exemplo, se o tamanho da sua linha for 2 KB, serão necessárias 2 WCUs para sustentar uma solicitação de gravação por segundo. Para obter mais informações sobre como estimar o consumo da capacidade de leitura e gravação de uma tabela, consultethe section called “Estime o consumo de capacidade”.

Se seu aplicativo ler ou gravar linhas maiores (até o tamanho máximo de linha do Amazon Keyspaces de 1 MB), ele consumirá mais unidades de capacidade. Para saber mais sobre como estimar o tamanho da linha, consulte the section called “Como calcular o tamanho da linha”. Por exemplo, suponha que você crie uma tabela provisionada com 6 RCUs e 6 WCUs. Com essas configurações, sua aplicação pode fazer o seguinte:

- Execute leituras `LOCAL_QUORUM` de até 24 KB por segundo (4 KB × 6 RCUs).
- Execute leituras `LOCAL_ONE` de até 48 KB por segundo (o dobro do throughput de leitura).
- Grave até 6 KB por segundo (1 KB × 6 WCUs).

O throughput provisionado é a quantidade máxima de capacidade de throughput que um aplicativo pode consumir de uma tabela. Se seu aplicativo exceder sua capacidade de throughput provisionada, você poderá observar erros de capacidade insuficientes.

Por exemplo, uma solicitação de leitura que não tem capacidade de throughput suficiente falha com uma exceção `Read_Timeout` e é publicada na métrica `ReadThrottleEvents`. Uma solicitação de gravação que não tem capacidade de throughput suficiente falha com uma exceção `Write_Timeout` e é publicada na métrica `WriteThrottleEvents`.

Você pode usar CloudWatch a Amazon para monitorar suas métricas de taxa de transferência provisionadas e reais e eventos de capacidade insuficiente. Para ter mais informações sobre essas métricas, consulte the section called “Métricas e dimensões”.

> **Note**
>
> Erros repetidos devido à capacidade insuficiente podem levar a exceções específicas do driver do lado do cliente, por exemplo, o driver DataStax Java falha com um. `NoHostAvailableException`

Para alterar as configurações de capacidade de throughput para tabelas, você pode usar a instrução AWS Management Console ou `ALTER TABLE` usando CQL. Para obter mais informações, consulte the section called “ALTER TABLE”.

Para saber mais sobre as cotas padrão da sua conta e como aumentá-las, consulte Cotas.

## Gerenciamento e visualização de modos de capacidade

Você pode consultar a tabela do sistema no espaço de chaves do sistema Amazon Keyspaces para revisar as informações do modo de capacidade sobre uma tabela. Você também pode ver se uma tabela está usando o modo de capacidade de throughput sob demanda ou provisionada. Se a tabela estiver configurada com o modo de capacidade de throughput provisionada, você poderá ver a capacidade de throughput provisionada para a tabela.

Exemplo

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'mykeyspace' and
 table_name = 'mytable';
```

Uma tabela configurada com o modo de capacidade sob demanda retorna o seguinte.

```
                    {
                        'capacity_mode': {
                            'last_update_to_pay_per_request_timestamp':
'1579551547603',
                            'throughput_mode': 'PAY_PER_REQUEST'
                         }
                    }
```

Uma tabela configurada com o modo de capacidade de throughput provisionada retorna o seguinte.

```
                    {
                        'capacity_mode': {
                            'last_update_to_pay_per_request_timestamp':
'1579048006000',
                            'read_capacity_units': '5000',
                            'throughput_mode': 'PROVISIONED',
                            'write_capacity_units': '6000'
                        }
                     }
```

O valor `last_update_to_pay_per_request_timestamp` é medido em milissegundos.

Para alterar a capacidade de throughput provisionada de uma tabela, use the section called “ALTER TABLE”.

### Considerações ao mudar os modos de capacidade

Quando você troca uma tabela de modo de capacidade provisionada para modo de capacidade sob demanda, o Amazon Keyspaces faz várias alterações na estrutura de sua tabela e suas partições. Esse processo pode levar alguns minutos. Durante o período de troca, sua tabela entrega throughput consistente com os valores de WCU e RCU provisionados anteriormente.

Quando você alterna do modo de capacidade sob demanda de volta para o modo de capacidade provisionada, sua tabela fornece um throughput consistente com o pico anterior atingido quando a tabela foi definida para o modo de capacidade sob demanda.

> **Note**
>
> Você pode alternar os modos de capacidade de provisionado para sob demanda somente uma vez em um período de 24 horas.

## Gerencie a capacidade de processamento automaticamente com o escalonamento automático do Amazon Keyspaces

Muitas cargas de trabalho de banco de dados são cíclicas por natureza ou são difíceis de prever com antecedência. Por exemplo, considere um aplicativo de rede social na qual a maioria dos usuários

está ativa durante o horário diurno. O banco de dados deve ser capaz de lidar com a atividade durante o dia, mas não há necessidade dos mesmos níveis de throughput à noite.

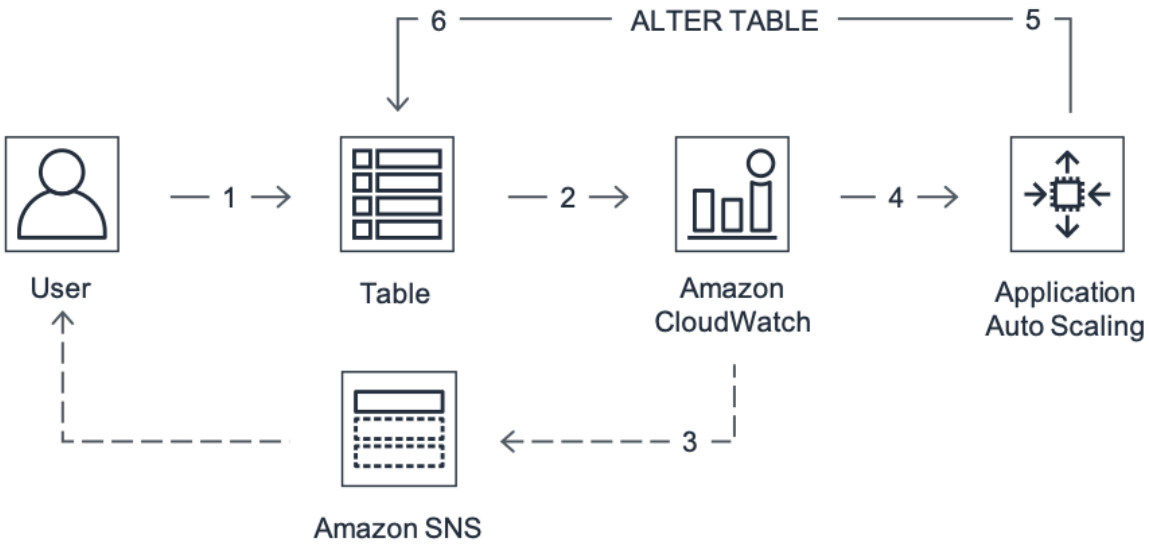
Outro exemplo pode ser um novo aplicativo de jogos para celular que está passando por uma rápida adoção. Se o jogo se tornar muito popular, talvez ele exceda os recursos de banco de dados disponíveis, podendo resultar em desempenho lento e clientes insatisfeitos. Esses tipos de cargas de trabalho muitas vezes exigem intervenção manual para dimensionar recursos de banco de dados, aumentando-os ou diminuindo-os em resposta a diferentes níveis de uso.

O Amazon Keyspaces (para Apache Cassandra) ajuda você a provisionar a capacidade de throughput de forma eficiente para cargas de trabalho variáveis, ajustando automaticamente a capacidade de throughput em resposta ao tráfego real do aplicativo. O Amazon Keyspaces usa o serviço Application Auto Scaling para aumentar e diminuir a capacidade de leitura e gravação de uma tabela por você. Para obter mais informações sobre o Application Auto Scaling, consulte o Guia do usuário do Application Auto Scaling.

> **Note**
>
> Para começar a trabalhar rapidamente com o ajuste de escala automático do Amazon Keyspaces, consulte the section called “Como usar o console”. Para gerenciar as políticas de escalabilidade do Amazon Keyspaces com a Cassandra Query Language (CQL), consulte. the section called “Usar SSL” Para saber como gerenciar as políticas de escalabilidade do Amazon Keyspaces usando a CLI, consulte. the section called “Uso da CLI”

## Como funciona o ajuste de escala automático do Amazon Keyspaces

O diagrama a seguir fornece uma visão geral de alto nível de como o ajuste de escala automático do Amazon Keyspaces gerencia a capacidade de throughput de uma tabela.




Para habilitar o ajuste de escala automático para uma tabela, você cria uma política de escalabilidade. A política de escalabilidade especifica se você deseja escalar a capacidade de leitura ou a capacidade de gravação (ou ambas), bem como as configurações mínimas e máximas de unidades de capacidade provisionadas para a tabela.

A política de escalabilidade também define uma meta de utilização. A utilização pretendida é a proporção entre unidades de capacidade consumidas e unidades de capacidade provisionadas em um dado momento, expressa como um percentual. O ajuste de escala automático usa um algoritmo de rastreamento para ajustar o throughput provisionado da tabela para cima ou para baixo em resposta a cargas de trabalho reais. Isso é feito para que a utilização real da capacidade permaneça igual ou próxima à sua meta de utilização.

É possível definir os valores de utilização do destino de ajuste de escala automático entre 20% e 90% como sua capacidade de gravação e leitura. A meta padrão de taxa de utilização é de 70 por cento. Você pode definir a meta de utilização como uma porcentagem menor se o tráfego mudar rapidamente e quiser que a capacidade comece a aumentar mais cedo. Você também pode definir a taxa de utilização desejada para uma taxa mais alta se o tráfego do aplicativo mudar mais lentamente e você quiser reduzir o custo do throughput.

Para obter mais informações sobre políticas de escalabilidade, consulte Políticas de escalabilidade de rastreamento de metas para Application Auto Scaling no Guia do Usuário do Application Auto Scaling.

Quando você cria uma política de escalabilidade, o Amazon Keyspaces cria dois pares de alarmes da CloudWatch Amazon em seu nome. Cada par representa seus limites superiores e inferiores para configurações de throughput provisionado e consumido. Esses CloudWatch alarmes são acionados quando a utilização real da tabela se desvia da sua meta de utilização por um longo período de tempo. Para saber mais sobre a Amazon CloudWatch, consulte o Guia CloudWatch do usuário da Amazon.

Quando um dos CloudWatch alarmes é acionado, o Amazon Simple Notification Service (Amazon SNS) envia uma notificação (se você a tiver ativado). O CloudWatch alarme então invoca o Application Auto Scaling para avaliar sua política de escalabilidade. Isso, por sua vez, emite uma solicitação Alter Table para o Amazon Keyspaces para ajustar a capacidade provisionada da tabela para cima ou para baixo conforme apropriado. Para saber mais sobre as notificações do Amazon SNS, consulte Como configurar as notificações do Amazon SNS.

O Amazon Keyspaces processa a solicitação de alteração da tabela, aumentando (ou diminuindo) a capacidade de throughput provisionado da tabela de forma que ela se aproxime da sua utilização prevista.

> **Note**
>
> O escalonamento automático do Amazon Keyspaces modifica as configurações de taxa de transferência provisionada somente quando a carga de trabalho real permanece elevada (ou deprimida) por um período sustentado de vários minutos. O algoritmo de rastreamento previsto do procura manter a utilização prevista em ou perto do seu valor escolhido em longo prazo. Picos de atividade súbitos de curta duração são acomodados pela capacidade de expansão interna da tabela.

## Como o escalonamento automático funciona para tabelas multirregionais

Para garantir que sempre haja capacidade suficiente de leitura e gravação para todas as réplicas de tabelas em toda uma tabela multirregional no modo Regiões da AWS de capacidade provisionada, recomendamos que você configure o escalonamento automático do Amazon Keyspaces.

Ao usar uma tabela multirregional no modo provisionado com escalonamento automático, você não pode desativar o escalonamento automático para uma única réplica de tabela. Mas você pode ajustar as configurações de escalonamento automático de leitura da tabela para diferentes regiões. Por exemplo, você pode especificar diferentes configurações de capacidade de leitura e escalabilidade automática de leitura para cada região na qual a tabela é replicada.

As configurações de escalonamento automático de leitura que você define para uma réplica de tabela em uma região especificada substituem as configurações gerais de escalonamento automático da tabela. No entanto, a capacidade de gravação precisa permanecer sincronizada em todas as réplicas da tabela para garantir que haja capacidade suficiente para replicar gravações em todas as regiões.

O auto scaling do Amazon Keyspaces atualiza de forma independente a capacidade provisionada da tabela em cada uma Região da AWS com base no uso naquela região. Como resultado, a capacidade provisionada em cada região para uma tabela multirregional pode ser diferente quando o auto scaling está ativo.

Você pode definir as configurações de auto scaling de uma tabela multirregional e suas réplicas usando o console, a API ou a CQL do Amazon Keyspaces. AWS CLI Para obter mais informações sobre como criar e atualizar as configurações de escalonamento automático para tabelas multirregionais, consulte. the section called “Como usar a replicação multirregional”

> **Note**
>
> Se você usa o escalonamento automático para tabelas multirregionais, deve sempre usar as operações de API do Amazon Keyspaces para definir as configurações de escalabilidade automática. Se você usa as operações da API Application Auto Scaling diretamente para definir as configurações de escalonamento automático, você não tem a capacidade de especificar a tabela Regiões da AWS multirregional. Isso pode resultar em configurações não suportadas.

## Observações de uso

Antes de começar a usar o ajuste de escala automático do Amazon Keyspaces, você deve estar ciente do seguinte:

- O ajuste de escala automático do Amazon Keyspaces pode aumentar a capacidade de leitura ou gravação sempre que necessário, de acordo com a sua política de ajuste de escala automático. Todas as cotas do Amazon Keyspaces permanecem em vigor, conforme descrito em Cotas.
- O ajuste de escala automático do Amazon Keyspaces não impede a modificação manual de configurações de throughput provisionado. Esses ajustes manuais não afetam nenhum CloudWatch alarme existente anexado à política de escalabilidade.

- Se você usar o console para criar uma tabela com capacidade de throughput provisionada, o ajuste de escala automático do Amazon Keyspaces será habilitado por padrão. É possível modificar as configurações de ajuste de escala automático a qualquer momento. Para ter mais informações, consulte the section called “Como usar o console”.
- Se você estiver usando AWS CloudFormation para criar políticas de escalabilidade, você deve gerenciar as políticas de escalabilidade AWS CloudFormation para que a pilha esteja sincronizada com o modelo da pilha. Se você alterar as políticas de escalabilidade do Amazon Keyspaces, elas serão substituídas pelos valores originais do modelo da pilha quando AWS CloudFormation a pilha for redefinida.
- Se você usa CloudTrail para monitorar a escalabilidade automática do Amazon Keyspaces, você pode ver alertas de chamadas feitas pelo Application Auto Scaling como parte de seu processo de validação de configuração. Para filtrar esses alertas, use o campo `invokedBy`, que conterá `application-autoscaling.amazonaws.com` para essas verificações de validação.

## Como gerenciar políticas de ajuste de escala automático do Amazon Keyspaces com o console

Você pode usar o console para habilitar o ajuste de escala automático do Amazon Keyspaces para tabelas novas e existentes. Você também pode usar o console para modificar as configurações de ajuste de escala automático ou desabilitá-lo.

> **Note**
>
> Para recursos mais avançados, como definir tempos de espera de expansão e redução, use o CQL ou o () para AWS CLI gerenciar as políticas de escalabilidade do Amazon AWS Command Line Interface Keyspaces de forma programática. Para obter mais informações, consulte Gerenciando a escalabilidade automática do Amazon Keyspaces com a Cassandra Query Language (CQL) ou Gerenciando políticas de escalabilidade do Amazon Keyspaces com a CLI.

Tópicos

- Antes de começar: concessão de permissões de usuário para o ajuste de escala automático do Amazon Keyspaces
- Como criar uma nova tabela com ajuste de escala automático habilitado no Amazon Keyspaces

- Como habilitar o ajuste de escala automático do Amazon Keyspaces em tabelas existentes
- Como modificar ou desabilitar configurações de ajuste de escala automático do Amazon Keyspaces
- Como visualizar atividades de ajuste de escala automático do Amazon Keyspaces no console

## Antes de começar: concessão de permissões de usuário para o ajuste de escala automático do Amazon Keyspaces

Para começar, confirme se o usuário tem as permissões apropriadas para criar e gerenciar as configurações de ajuste de escala automático. No AWS Identity and Access Management (IAM), a política AWS gerenciada `AmazonKeyspacesFullAccess` é necessária para gerenciar as políticas de escalabilidade do Amazon Keyspaces.

> **Important**
>
> Permissões do `application-autoscaling:*` são necessárias para desativar o ajuste de escala automático em uma tabela. Você deve desativar o escalonamento automático de uma tabela antes de excluí-la.

Para configurar um usuário do IAM para acesso ao console do Amazon Keyspaces e ajuste de escala automático do Amazon Keyspaces, adicione a política a seguir.

Para anexar a política **`AmazonKeyspacesFullAccess`**

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel do console do IAM, escolha Users (Usuários) e, em seguida, escolha seu usuário do IAM na lista.
3. Na página Summary (Resumo), escolha Add permissions (Adicionar permissões).
4. Escolha Attach existing policies directly (Anexar políticas existentes diretamente).
5. Na lista de políticas, escolha e AmazonKeyspacesFullAccess, em seguida, escolha Avançar: Revisão.
6. Escolha Add permissions (Adicionar permissões).

# Como criar uma nova tabela com ajuste de escala automático habilitado no Amazon Keyspaces

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para criar uma nova tabela com ajuste de escala automático habilitado

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas) e Create table (Criar tabela).
3. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a nova tabela.
4. Na seção Colunas, crie o esquema para sua tabela.
5. Na seção Chave primária, defina a chave primária da tabela e selecione colunas de agrupamento opcionais.
6. Na seção Configurações da tabela, selecione Personalizar configurações.
7. Continue com as configurações de capacidade de leitura/gravação.
8. Para o Modo de capacidade, escolha Provisionado.
9. Na seção Capacidade de leitura, confirme se a opção Escalar automaticamente está selecionada.

   Nesta etapa, você seleciona as unidades de capacidade de leitura mínima e máxima para a tabela, bem como a utilização desejada.

   - Unidades de capacidade mínima - insira o valor do nível mínimo de throughput que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).

- Unidades de capacidade máxima - insira a quantidade máxima de throughput que você deseja provisionar para a tabela. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).
- Utilização desejada- insira uma taxa de utilização desejada entre 20% e 90%. Quando o tráfego excede a taxa de utilização desejada definida, a capacidade é automaticamente aumentada. Quando o tráfego fica abaixo da meta definida, ela é automaticamente reduzida novamente.

> **Note**
>
> Para saber mais sobre as cotas padrão da sua conta e como aumentá-las, consulte Cotas.

10. Na seção Capacidade de gravação, escolha as mesmas configurações definidas na etapa anterior para capacidade de leitura ou defina os valores de capacidade manualmente.
11. Escolha Create table. Sua tabela é criada com os parâmetros padrão de ajuste de escala automático.

## Como habilitar o ajuste de escala automático do Amazon Keyspaces em tabelas existentes

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para habilitar o ajuste de escala automático do Amazon Keyspaces para uma tabela existente

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela com a qual você deseja trabalhar e vá até a guia Capacidade.
3. Na seção Configurações de capacidade, escolha Editar.

4. Em Modo de capacidade, verifique se a tabela está usando o modo de capacidade provisionada.
5. Selecione Escalar automaticamente e veja a etapa 6 em Como criar uma nova tabela com ajuste de escala automático habilitado no Amazon Keyspaces para editar a capacidade de leitura e gravação.
6. Quando as configurações de escala automática estiverem definidas, escolha Salvar.

## Como modificar ou desabilitar configurações de ajuste de escala automático do Amazon Keyspaces

Você pode usar o AWS Management Console para modificar suas configurações de escalabilidade automática do Amazon Keyspaces. Para fazer isso, escolha a tabela que você deseja editar e vá até a guia Capacidade. Na seção Configurações de capacidade, escolha Editar. Agora você pode modificar as configurações nas seções Capacidade de leitura ou Capacidade de gravação. Para ter mais informações sobre essas configurações, consulte Como criar uma nova tabela com ajuste de escala automático habilitado no Amazon Keyspaces.

Para desativar a escalabilidade automática do Amazon Keyspaces, desmarque a caixa de seleção Escalar automaticamente. A desativação do escalonamento automático cancela o registro da tabela como um destino escalável com o Application Auto Scaling. Para excluir a função vinculada ao serviço usada pelo Application Auto Scaling para acessar a tabela do Amazon Keyspaces, siga as etapas descritas em the section called “Exclusão de um perfil vinculado ao serviço do Amazon Keyspaces”.

> **Note**
>
> Para excluir a função vinculada ao serviço que o Application Auto Scaling usa, você deve desativar o escalonamento automático em todas as tabelas da conta. Regiões da AWS

## Como visualizar atividades de ajuste de escala automático do Amazon Keyspaces no console

Você pode monitorar como a escalabilidade automática do Amazon Keyspaces usa recursos usando a Amazon CloudWatch, que gera métricas sobre seu uso e desempenho. Siga as etapas no Guia do Application Auto Scaling usuário para criar um CloudWatch painel.

# Gerenciando a escalabilidade automática do Amazon Keyspaces com a Cassandra Query Language (CQL)

Para criar e gerenciar configurações de auto scaling para tabelas do Amazon Keyspaces com a Cassandra Query Language (CQL), você pode usar. `cqlsh` Este tópico fornece uma visão geral das tarefas de escalonamento automático que você pode gerenciar programaticamente usando o CQL.

Para obter mais informações sobre as instruções CQL descritas neste tópico, consultethe section called “Instruções DDL”.

Tópicos

- Antes de começar
- Crie uma nova tabela com escalabilidade automática usando CQL
- Ative o escalonamento automático em uma tabela existente usando CQL
- Veja a configuração de auto scaling do Amazon Keyspaces da sua tabela usando CQL
- Desative o escalonamento automático do Amazon Keyspaces para uma tabela usando CQL

## Antes de começar

Você precisa concluir as tarefas seguintes antes de iniciar.

Configurar permissões do

Se você ainda não fez isso, configure as permissões apropriadas para o usuário criar e gerenciar configurações de escalabilidade automática. No AWS Identity and Access Management (IAM), a política AWS gerenciada `AmazonKeyspacesFullAccess` é necessária para gerenciar as políticas de escalabilidade do Amazon Keyspaces. Para obter detalhes das etapas, consulte, the section called “Antes de começar: concessão de permissões de usuário para o ajuste de escala automático do Amazon Keyspaces”.

Configurar o **`cqlsh`**

Se ainda não tiver feito isso, você deve instalar e configurar`cqlsh`. Para fazer isso, siga as instruções emthe section called “Usar a `cqlsh-expansion`”. Em seguida, você pode usar o AWS CloudShell para executar os comandos nas seções a seguir.

## Crie uma nova tabela com escalabilidade automática usando CQL

Ao criar uma nova tabela do Amazon Keyspaces, você pode habilitar automaticamente o auto scaling para a capacidade de gravação ou leitura da tabela na declaração. `CREATE TABLE` Isso permite que o Amazon Keyspaces entre em contato com o Application Auto Scaling em seu nome para registrar a tabela como uma meta escalável e ajustar a capacidade provisionada de gravação ou leitura.

Para obter mais informações sobre como criar uma tabela multirregional e definir diferentes configurações de escalonamento automático para réplicas de tabela, consulte. the section called “Criação de uma tabela multirregional com configurações padrão (CQL)”

> **Note**
>
> O escalonamento automático do Amazon Keyspaces exige a presença de uma função vinculada ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) para realizar ações automáticas de escalabilidade em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para definir as configurações de escalabilidade automática para uma tabela de forma programática, você usa a `AUTOSCALING_SETTINGS` instrução que contém os parâmetros para o escalonamento automático do Amazon Keyspaces. Os parâmetros definem as condições que orientam o Amazon Keyspaces a ajustar a taxa de transferência provisionada da sua tabela e quais ações opcionais adicionais devem ser tomadas. Neste exemplo, você define as configurações de escalonamento automático para mytable.

A política contém os elementos a seguir:

- `AUTOSCALING_SETTINGS`— Especifica se o Amazon Keyspaces tem permissão para ajustar a capacidade de processamento em seu nome. Os seguintes valores são obrigatórios:
  - `provisioned_write_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `provisioned_read_capacity_autoscaling_update`:
    - `minimum_units`

  - `maximum_units`
- `scaling_policy`— O Amazon Keyspaces oferece suporte à política de rastreamento de alvos. Para definir a política de rastreamento de alvos, você configura os seguintes parâmetros.
  - `target_value`— O escalonamento automático do Amazon Keyspaces garante que a proporção entre a capacidade consumida e a capacidade provisionada permaneça igual ou próxima a esse valor. Você define `target_value` como uma porcentagem.
  - `disableScaleIn`: (Opcional) Um `boolean` que especifica se `scale-in` está desativado ou ativado para a tabela. Esse parâmetro está desativado por padrão. Para ativar`scale-in`, defina o `boolean` valor como`FALSE`. Isso significa que a capacidade é reduzida automaticamente para uma tabela em seu nome.
  - `scale_out_cooldown`: A atividade de expansão aumenta a capacidade de throughput provisionado de sua tabela. Para adicionar um período de desaquecimento nas atividades de aumento da escala na horizontal, especifique um valor, em segundos, para `scale_out_cooldown`. Se você não especificar um valor, o valor padrão será 0. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.
  - `scale_in_cooldown`: Uma atividade de redução da escala diminui o throughput provisionado de sua tabela. Para adicionar um período de desaquecimento nas atividades de redução da escala na horizontal, especifique um valor, em segundos, para `scale_in_cooldown`. Se você não especificar um valor, o valor padrão será 0. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.

> **Note**
>
> Para compreender melhor como o `target_value` funciona, suponha que você tenha uma tabela com uma configuração de throughput provisionado de 200 unidades de capacidade de gravação. Você decide criar uma política de dimensionamento para essa tabela, com um `target_value` de 70%.
>
> Agora, suponha que você comece a direcionar tráfego de gravação para a tabela de forma que o throughput de gravação real seja de 150 unidades de capacidade. A consumed-to-provisioned proporção agora é (150/200), ou 75 por cento. Essa proporção excede sua meta, então o auto scaling aumenta a capacidade de gravação provisionada para 215, de modo

que a proporção seja (150/215), ou 69,77 por cento — o mais próxima possível da sua, mas não a excedendo. `target_value`

Para mytable, você define a capacidade `TargetValue` de leitura e gravação em 50 por cento. O auto scaling do Amazon Keyspaces ajusta a taxa de transferência provisionada da tabela na faixa de 5 a 10 unidades de capacidade para que a proporção permaneça em ou perto de 50%. consumed-to-provisioned Para a capacidade de leitura, você define os valores para `ScaleOutCooldown` e `ScaleInCooldown` para 60 segundos.

Você pode usar a seguinte declaração para criar uma nova tabela do Amazon Keyspaces com o auto scaling ativado.

```
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
            }
        }
    }
```

```
};
```

### Ative o escalonamento automático em uma tabela existente usando CQL

Para uma tabela existente do Amazon Keyspaces, você pode ativar o escalonamento automático para a capacidade de gravação ou leitura da tabela usando a instrução. `ALTER TABLE` Se você estiver atualizando uma tabela que está atualmente no modo de capacidade sob demanda, isso `capacity_mode` é necessário. Se sua tabela já estiver no modo de capacidade provisionada, esse campo poderá ser omitido.

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

No exemplo a seguir, a instrução atualiza a tabela mytable, que está no modo de capacidade sob demanda. A instrução altera o modo de capacidade da tabela para o modo provisionado com o escalonamento automático ativado.

A capacidade de gravação é configurada na faixa de 5 a 10 unidades de capacidade com um valor alvo de 50%. A capacidade de leitura também é configurada na faixa de 5 a 10 unidades de capacidade com um valor alvo de 50%. Para a capacidade de leitura, você define os valores para `scale_out_cooldown` e `scale_in_cooldown` para 60 segundos.

```
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
            }
        }
    }
};
```

## Veja a configuração de auto scaling do Amazon Keyspaces da sua tabela usando CQL

Para ver detalhes da configuração de escalonamento automático de uma tabela, use o comando a seguir.

```
SELECT * FROM system_schema_mcs.autoscaling WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

A saída desse comando é semelhante a esta.

```
 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}}}
```

### Desative o escalonamento automático do Amazon Keyspaces para uma tabela usando CQL

Você pode desativar o escalonamento automático do Amazon Keyspaces para sua tabela a qualquer momento. Se você não precisar mais escalar a capacidade de leitura ou gravação da sua tabela, considere desativar o auto scaling para que o Amazon Keyspaces não continue modificando as configurações de capacidade de leitura ou gravação da sua tabela. Você pode atualizar a tabela com uma `ALTER TABLE` declaração.

A instrução a seguir desativa o escalonamento automático para a capacidade de gravação da tabela mytable. Também exclui os CloudWatch alarmes que foram criados em seu nome.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Para excluir a função vinculada ao serviço que o Application Auto Scaling usa, você deve desativar o escalonamento automático em todas as tabelas da conta. Regiões da AWS

## Gerenciando políticas de escalabilidade do Amazon Keyspaces com a CLI

Para atualizar e gerenciar as configurações de auto scaling do Amazon Keyspaces de forma programática, você pode usar o AWS Command Line Interface () AWS CLI ou a API. AWS Para gerenciar as políticas de escalabilidade automática do Amazon Keyspaces usando a Cassandra Query Language (CQL), consulte. the section called “Usar SSL” Este tópico fornece uma visão geral das tarefas de escalonamento automático que você pode gerenciar programaticamente usando o. AWS CLI

Para obter mais informações sobre os AWS CLI comandos do Amazon Keyspaces descritos neste tópico, consulte a Referência de AWS CLI comandos.

Tópicos

- Antes de começar

- Crie uma nova tabela com escala automática usando o AWS CLI
- Ative o escalonamento automático em uma tabela existente usando o AWS CLI
- Visualize a configuração de auto scaling do Amazon Keyspaces da sua tabela usando o AWS CLI
- Desative o escalonamento automático do Amazon Keyspaces para uma tabela usando o AWS CLI

## Antes de começar

Você precisa concluir as tarefas seguintes antes de iniciar.

Configurar permissões do

Se você ainda não fez isso, configure as permissões apropriadas para o usuário criar e gerenciar configurações de escalabilidade automática. No AWS Identity and Access Management (IAM), a política AWS gerenciada `AmazonKeyspacesFullAccess` é necessária para gerenciar as políticas de escalabilidade do Amazon Keyspaces. Para obter detalhes das etapas, consulte, the section called “Antes de começar: concessão de permissões de usuário para o ajuste de escala automático do Amazon Keyspaces”.

Instalar a AWS CLI

Caso ainda não tenha feito isso, você deve instalar e configurar a AWS CLI. Para fazer isso, acesse o Guia AWS Command Line Interface do usuário e siga estas instruções:

- Instalar a AWS CLI
- Configurar a AWS CLI

## Crie uma nova tabela com escala automática usando o AWS CLI

Ao criar uma nova tabela do Amazon Keyspaces, você pode habilitar automaticamente o auto scaling para a capacidade de gravação ou leitura da tabela na operação. `CreateTable` Isso permite que o Amazon Keyspaces entre em contato com o Application Auto Scaling em seu nome para registrar a tabela que você especifica como meta escalável e ajustar a capacidade provisionada de gravação ou leitura.

Para obter mais informações sobre como criar uma tabela multirregional com configuração de escalonamento automático, consulte. the section called “Criação de uma nova tabela multirregional no modo provisionado com escalabilidade automática (CLI)”

> **Note**
>
> O escalonamento automático do Amazon Keyspaces exige a presença de uma função vinculada ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) para realizar ações automáticas de escalabilidade em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para definir as configurações de escalabilidade automática para uma tabela de forma programática, você usa a `autoScalingSpecification` ação que define os parâmetros para o escalonamento automático do Amazon Keyspaces. Os parâmetros definem as condições que orientam o Amazon Keyspaces a ajustar a taxa de transferência provisionada da sua tabela e quais ações opcionais adicionais devem ser tomadas. Neste exemplo, você define as configurações de escalonamento automático para mytable.

A política contém os elementos a seguir:

- `autoScalingSpecification`— Especifica se o Amazon Keyspaces tem permissão para ajustar a taxa de transferência da capacidade em seu nome. Você pode ativar o escalonamento automático para capacidade de leitura e gravação separadamente. Em seguida, você deve especificar os seguintes parâmetros para`autoScalingSpecification`:
  - `writeCapacityAutoScaling`— As unidades de capacidade máxima e mínima de gravação.
  - `readCapacityAutoScaling`— As unidades de capacidade máxima e mínima de leitura.
  - `scalingPolicy`— O Amazon Keyspaces oferece suporte à política de rastreamento de alvos. Para definir a política de rastreamento de alvos, você configura os seguintes parâmetros.
    - `targetValue`— O escalonamento automático do Amazon Keyspaces garante que a proporção entre a capacidade consumida e a capacidade provisionada permaneça igual ou próxima a esse valor. Você define `targetValue` como uma porcentagem.
    - `disableScaleIn`: (Opcional) Um `boolean` que especifica se `scale-in` está desativado ou ativado para a tabela. Esse parâmetro está desativado por padrão. Para ativar`scale-in`, defina o `boolean` valor como`FALSE`. Isso significa que a capacidade é reduzida automaticamente para uma tabela em seu nome.
    - `scaleOutCooldown`: A atividade de expansão aumenta a capacidade de throughput provisionado de sua tabela. Para adicionar um período de desaquecimento nas

atividades de aumento da escala na horizontal, especifique um valor, em segundos, para `ScaleOutCooldown`. O valor padrão é 0. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.

- `scaleInCooldown`: Uma atividade de redução da escala diminui o throughput provisionado de sua tabela. Para adicionar um período de desaquecimento nas atividades de redução da escala na horizontal, especifique um valor, em segundos, para `ScaleInCooldown`. O valor padrão é 0. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.

> **Note**
>
> Para compreender melhor como o `TargetValue` funciona, suponha que você tenha uma tabela com uma configuração de throughput provisionado de 200 unidades de capacidade de gravação. Você decide criar uma política de dimensionamento para essa tabela, com um `TargetValue` de 70%.
> Agora, suponha que você comece a direcionar tráfego de gravação para a tabela de forma que o throughput de gravação real seja de 150 unidades de capacidade. A consumed-to-provisioned proporção agora é (150/200), ou 75 por cento. Essa proporção excede sua meta, então o auto scaling aumenta a capacidade de gravação provisionada para 215, de modo que a proporção seja (150/215), ou 69,77 por cento — o mais próxima possível da sua, mas não a excedendo. `TargetValue`

Para mytable, você define a capacidade `TargetValue` de leitura e gravação em 50 por cento. O auto scaling do Amazon Keyspaces ajusta a taxa de transferência provisionada da tabela na faixa de 5 a 10 unidades de capacidade para que a proporção permaneça em ou perto de 50%. consumed-to-provisioned Para a capacidade de leitura, você define os valores para `ScaleOutCooldown` e `ScaleInCooldown` para 60 segundos.

Ao criar tabelas com configurações complexas de escalonamento automático, é útil carregar as configurações de escalonamento automático de um arquivo JSON. Para o exemplo a seguir, você pode baixar o arquivo JSON de exemplo do auto-scaling.zip e extrair`auto-scaling.json`, anotando o caminho para o arquivo. Neste exemplo, o arquivo JSON está localizado no diretório

atual. Para diferentes opções de caminho de arquivo, consulte Como carregar parâmetros de um arquivo.

```
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
```

### Ative o escalonamento automático em uma tabela existente usando o AWS CLI

Para uma tabela existente do Amazon Keyspaces, você pode ativar o escalonamento automático para a capacidade de gravação ou leitura da tabela usando a operação. `UpdateTable` Para obter mais informações sobre como atualizar as configurações de escalonamento automático para uma tabela multirregional, consulte. the section called “Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CLI)”

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Você pode usar o comando a seguir para ativar o auto scaling do Amazon Keyspaces para uma tabela existente. As configurações de escalonamento automático da tabela são carregadas de um arquivo JSON. Para o exemplo a seguir, você pode baixar o arquivo JSON de exemplo do auto-scaling.zip e extrair`auto-scaling.json`, anotando o caminho para o arquivo. Neste exemplo, o arquivo JSON está localizado no diretório atual. Para diferentes opções de caminho de arquivo, consulte Como carregar parâmetros de um arquivo.

Para obter mais informações sobre as configurações de escalonamento automático usadas no exemplo a seguir, consultethe section called “Crie uma nova tabela com escala automática usando o 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
```

## Visualize a configuração de auto scaling do Amazon Keyspaces da sua tabela usando o AWS CLI

Para visualizar a configuração de escalonamento automático de uma tabela, você pode usar a `get-table-auto-scaling-settings` operação. O comando CLI a seguir é um exemplo disso.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

A saída desse comando é semelhante a esta.

```
{
    "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
                }
            }
        }
    }
}
```

### Desative o escalonamento automático do Amazon Keyspaces para uma tabela usando o AWS CLI

Você pode desativar o escalonamento automático do Amazon Keyspaces para sua tabela a qualquer momento. Se você não precisar mais escalar a capacidade de leitura ou gravação da sua tabela, considere desativar o auto scaling para que o Amazon Keyspaces não continue modificando as configurações de capacidade de leitura ou gravação da sua tabela. Você pode atualizar a tabela com uma `UpdateTable` operação.

O comando a seguir desativa o escalonamento automático da capacidade de leitura da tabela. Também exclui os CloudWatch alarmes que foram criados em seu nome.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
            \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Para excluir a função vinculada ao serviço que o Application Auto Scaling usa, você deve desativar o escalonamento automático em todas as tabelas da conta. Regiões da AWS

## Usando a capacidade de intermitência de forma eficaz no Amazon Keyspaces

O Amazon Keyspaces fornece alguma flexibilidade no provisionamento de throughput por partição ao oferecer capacidade de expansão. Quando você não está usando totalmente o throughput de uma partição, o Amazon Keyspaces reserva uma parte dessa capacidade não utilizada para expansões de throughput posteriores para lidar com os picos de uso.

No momento, o Amazon Keyspaces mantém até cinco minutos (300 segundos) de capacidade de leitura e gravação não utilizada. Durante uma intermitência ocasional de atividades de leitura e gravação, essas unidades de capacidade extra podem ser consumidas muito rápido, ainda mais rápido do que a capacidade de throughput provisionada por segundo que você definiu para a sua tabela.

O Amazon Keyspaces também pode consumir capacidade de expansão para manutenção em segundo plano e para outras tarefas sem prévio aviso.

Note que esses detalhes da capacidade de intermitência podem mudar no futuro.

# Como estimar o consumo de capacidade no Amazon Keyspaces

Quando você lê ou grava dados no Amazon Keyspaces, a quantidade de unidades de solicitação de leitura/gravação (RRUS/WRUs) ou unidades de capacidade de leitura/gravação (RCUs/WCUs) que sua consulta consome depende da quantidade total de dados que o Amazon Keyspaces precisa processar para executar a consulta. Em alguns casos, os dados retornados ao cliente podem ser um subconjunto dos dados que o Amazon Keyspaces precisou ler para processar a consulta. Para gravações condicionais, o Amazon Keyspaces consome capacidade de gravação mesmo se a verificação condicional falhar.

Para estimar a quantidade total de dados que estão sendo processados para uma solicitação, é preciso considerar o tamanho codificado de uma linha e o número total de linhas. Este tópico aborda alguns exemplos de cenários comuns e padrões de acesso para mostrar como o Amazon Keyspaces processa consultas e como isso afeta o consumo de capacidade. Você pode seguir os exemplos para estimar os requisitos de capacidade de suas tabelas e usar CloudWatch a Amazon para observar o consumo de capacidade de leitura e gravação nesses casos de uso.

Para obter informações sobre como calcular o tamanho codificado das linhas no Amazon Keyspaces, consulte. the section called “Como calcular o tamanho da linha”

Tópicos


- Intervalo de consultas
- Limitar consultas
- Digitalizações de tabela
- Transações leves
- Estime o consumo da capacidade de leitura e gravação com a Amazon CloudWatch

## Intervalo de consultas

Para analisar o consumo de capacidade de leitura de uma consulta de intervalo, usamos a tabela de exemplo a seguir, que está usando o modo de capacidade sob demanda.

```
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>
```

Agora, execute a consulta a seguir nessa tabela.

```
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;
```

Você recebe o seguinte conjunto de resultados da consulta e a operação de leitura realizada pelo Amazon Keyspaces consome 2 RRUs no modo de consistência. `LOCAL_QUORUM`

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 60 | value_1
```

O Amazon Keyspaces consome 2 RRUs para avaliar as linhas com os valores `ck3=60` e `ck3=70` processar a consulta. No entanto, o Amazon Keyspaces retorna somente a linha em que a `WHERE` condição especificada na consulta é verdadeira, que é a linha com valor. `ck3=60` Para avaliar o intervalo especificado na consulta, o Amazon Keyspaces lê a linha correspondente ao limite superior do intervalo, nesse caso`ck3 = 70`, mas não retorna essa linha no resultado. O consumo da capacidade de leitura é baseado nos dados lidos durante o processamento da consulta, não nos dados retornados.

## Limitar consultas

Ao processar uma consulta que usa a `LIMIT` cláusula, o Amazon Keyspaces lê as linhas até o tamanho máximo da página ao tentar corresponder à condição especificada na consulta. Se o Amazon Keyspaces não conseguir encontrar dados correspondentes suficientes que correspondam ao `LIMIT` valor na primeira página, uma ou mais chamadas paginadas poderão ser necessárias. Para continuar lendo na próxima página, você pode usar um token de paginação. O tamanho padrão

da página é de 1 MB. Para consumir menos capacidade de leitura ao usar `LIMIT` cláusulas, você pode reduzir o tamanho da página. Para obter mais informações sobre paginação, consulte the section called “Como paginar resultados”.

Por exemplo, vejamos a consulta a seguir.

```
SELECT * FROM my_table WHERE partition_key=1234 LIMIT 1;”
```

Se você não definir o tamanho da página, o Amazon Keyspaces lê 1 MB de dados, mesmo que retorne apenas 1 linha para você. Para que o Amazon Keyspaces leia apenas uma linha, você pode definir o tamanho da página como 1 para essa consulta. Nesse caso, o Amazon Keyspaces só leria uma linha, desde que você não tivesse linhas expiradas com base nas ime-to-live configurações T ou nos timestamps do lado do cliente. Para consumir menos capacidade de leitura, recomendamos definir o tamanho da página igual ao `LIMIT` valor para reduzir a quantidade de dados que o Amazon Keyspaces lê.

## Digitalizações de tabela

As consultas que resultam em varreduras completas da tabela, por exemplo, consultas usando a `ALLOW FILTERING` opção, são outro exemplo de consultas que processam mais leituras do que as que retornam como resultados. E o consumo da capacidade de leitura é baseado nos dados lidos, não nos dados retornados.

Para o exemplo de varredura de tabela, usamos a tabela de exemplo a seguir no modo de capacidade sob demanda.

```
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>
```

O Amazon Keyspaces cria uma tabela no modo de capacidade sob demanda com quatro partições por padrão. Nesta tabela de exemplo, todos os dados são armazenados em uma partição e as três partições restantes estão vazias.

Agora, execute a consulta a seguir na tabela.

```
SELECT * from amazon_keyspaces.example_table_2;
```

Essa consulta resulta em uma operação de varredura de tabela em que o Amazon Keyspaces escaneia todas as quatro partições da tabela e consome 6 RRUs no modo de consistência. `LOCAL_QUORUM` Primeiro, o Amazon Keyspaces consome 3 RRUs para ler as três linhas com. `pk='pk'` Em seguida, o Amazon Keyspaces consome as 3 RRUs adicionais para escanear as três partições vazias da tabela. Como essa consulta resulta em uma varredura de tabela, o Amazon Keyspaces escaneia todas as partições na tabela, incluindo partições sem dados.

## Transações leves

As transações leves (LWT) permitem que você execute operações de gravação condicional nos dados da tabela. As operações de atualização condicional são úteis ao inserir, atualizar e excluir registros com base nas condições que avaliam o estado atual.

No Amazon Keyspaces, todas as operações de gravação exigem a consistência LOCAL_QUORUM e não há cobrança adicional pelo uso de LWTs. A diferença para LWTs é que, quando uma verificação de condição de LWT resulta em FALSE, ela consome unidades de capacidade de gravação. O número de unidades de capacidade de gravação consumidas depende do tamanho da linha. Se o tamanho da linha for de 2 KB, a falha na gravação condicional consumirá duas unidades de capacidade de gravação. Se a linha não existir atualmente na tabela, a operação consumirá uma unidade de capacidade de gravação. Ao monitorar a `ConditionalCheckFailed` métrica, CloudWatch você pode determinar a capacidade consumida pelas falhas de verificação de condição do LWT.

## Estime o consumo da capacidade de leitura e gravação com a Amazon CloudWatch

Para estimar e monitorar o consumo da capacidade de leitura e gravação, você pode usar um CloudWatch painel. Para obter mais informações sobre as métricas disponíveis para o Amazon Keyspaces, consulte. the section called “Métricas e dimensões”

Para monitorar as unidades de capacidade de leitura e gravação consumidas por uma instrução específica com CloudWatch, você pode seguir estas etapas.

1. Crie uma nova tabela com dados de amostra
2. Configure um CloudWatch painel do Amazon Keyspaces para a tabela. Para começar, você pode usar um modelo de painel disponível no Github.
3. Execute a instrução CQL, por exemplo, usando a `ALLOW FILTERING` opção, e verifique as unidades de capacidade de leitura consumidas para a verificação completa da tabela no painel.

# Como trabalhar com espaço de chaves, tabelas e linhas no Amazon Keyspaces (para Apache Cassandra)

Este capítulo fornece detalhes sobre como trabalhar com espaços de chaves, tabelas, linhas e muito mais no Amazon Keyspaces (para Apache Cassandra). Para saber como monitorar espaços de chave e tabelas com a Amazon CloudWatch, consultethe section called “Monitoramento com CloudWatch”.

Tópicos


- Como trabalhar com espaços de chaves no Amazon Keyspaces
- Como trabalhar com tabelas no Amazon Keyspaces
- Como trabalhar com linhas no Amazon Keyspaces
- Como trabalhar com consultas no Amazon Keyspaces
- Como trabalhar com particionadores no Amazon Keyspaces
- Trabalhando com tags e rótulos para recursos do Amazon Keyspaces


## Como trabalhar com espaços de chaves no Amazon Keyspaces

Esta seção fornece detalhes sobre como trabalhar com espaços de chaves no Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como trabalhar com espaços de chaves do sistema no Amazon Keyspaces
- Como criar espaços de chaves no Amazon Keyspaces


### Como trabalhar com espaços de chaves do sistema no Amazon Keyspaces

O Amazon Keyspaces usa quatro espaços de chaves do sistema:

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

As seções a seguir fornecem detalhes sobre os espaços de chave do sistema e as tabelas do sistema que são compatíveis com o Amazon Keyspaces.

### `system`

Este é um espaço de chaves do Cassandra. O Amazon Keyspaces usa as tabelas a seguir.

| Nomes das tabelas | Nomes de colunas | Comentários |
|---|---|---|
| `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` | Informações sobre o espaço de chaves local. |
| `peers` | `peer, data_center, host_id, preferred_ip, rack, release_version, rpc_address, schema_version, tokens` | Consulte essa tabela para ver os endpoints disponíveis. Por exemplo, se você estiver se conectando por meio de um endpoint público, verá uma lista de nove endereços IP disponíveis. Se você estiver se conectando por meio de um endpoint FIPS, verá uma lista de três endereços IP. Se você estiver se conectando por meio de um AWS PrivateLink VPC endpoint, |

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| | | verá a lista de endereços IP que você configurou. Para ter mais informações, consulte the section called “Como preencher entradas da tabela `system.peers` com informações do endpoint da VPC de interface”. |
| `size_estimates` | `keyspace_name, table_name, range_start, range_end, mean_partition_size, partitions_count` | Essa tabela define o tamanho total e o número de partições para cada intervalo de tokens para cada tabela. Isso é necessário para o conector Apache Cassandra do Spark, que usa o tamanho estimado da partição para distribuir o trabalho. |
| `prepared_statements` | `prepared_id, logged_keyspace, query_string` | Essa tabela contém informaçõ es sobre consultas salvas. |

## `system_schema`

Este é um espaço de chaves do Cassandra. O Amazon Keyspaces usa as tabelas a seguir.

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| `keyspaces` | `keyspace_name, durable_writes, replication` | Informações sobre um espaço de chaves específico. |
| `tables` | `keyspace_name, table_name, bloom_fil` | Informações sobre uma tabela específica. |

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| | `ter_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` | |
| `columns` | `keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type` | Informações sobre uma coluna específica. |

## `system_schema_mcs`

Esse é um keyspace do Amazon Keyspaces que armazena informações sobre ou configurações específicas do AWS Amazon Keyspaces.

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| `keyspaces` | `keyspace_name, durable_writes, replication` | Consulte essa tabela para descobrir programaticamente se um espaço de chaves foi criado. Para ter mais |

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| | | informações, consulte the section called “Como criar espaços de chaves”. |
| 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 | Consulte essa tabela para descobrir o status de uma tabela específica. Para ter mais informações, consulte the section called “Criar tabelas”.<br><br>Você também pode consultar essa tabela para listar as configurações específicas do Amazon Keyspaces e armazenadas como. `custom_properties` Por exemplo: .<br><br>• `capacity_mode`<br>• `client_side_timestamps`<br>• `encryption_specification`<br>• `point_in_time_recovery`<br>• `ttl` |
| tables_history | keyspace_name, table_name, event_time, creation_time, custom_properties, event | Consulte essa tabela para saber mais sobre as alterações de esquema de uma tabela específica. |

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| `columns` | `keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type` | Essa tabela é idêntica à tabela Cassandra no espaço de chaves `system_schema` . |
| `tags` | `resource_id, keyspace_name, resource_name, resource_type, tags` | Consulte essa tabela para descobrir se um espaço de chaves tem tags. Para ter mais informações, consulte the section called “Adição de tags a espaços de chaves e tabelas novos ou existentes usando o CQL”. |
| `autoscaling` | `keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update` | Consulte essa tabela para obter as configurações de escalonamento automático de uma tabela provisionada. Observe que essas configurações não estarão disponíveis até que a tabela esteja ativa. Para consultar essa tabela, você precisa especificar `keyspace_name` e `table_name` na cláusula `WHERE`. Para ter mais informações, consulte the section called “Usar SSL”. |

## `system_multiregion_info`

Esse é um espaço de chaves do Amazon Keyspaces que armazena informações sobre replicação multirregional.

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| `tables` | `keyspace_name, table_name, region, status` | Essa tabela contém informaçõ es sobre tabelas multirreg ionais — por exemplo, em Regiões da AWS que a tabela é replicada e o status da tabela. Você também pode consultar essa tabela para listar as configurações específicas do Amazon Keyspaces que são armazenadas como. `custom_properties` Por exemplo: .<br><br>• `capacity_mode`<br><br>Para consultar essa tabela, você precisa especific ar `keyspace_name` e `table_name` na cláusula `WHERE`. Para ter mais informações, consulte the section called “Como criar um espaço de chaves multirreg ional (CQL)”. |
| `autoscaling` | `keyspace_name, table_name, provision ed_read_capacity_a utoscaling_update, provisioned_write_ capacity_autoscali ng_update, region` | Consulte essa tabela para obter as configurações de escalonamento automátic o de uma tabela provision ada por várias regiões. Observe que essas configura ções não estarão disponíve is até que a tabela esteja |

| Nomes das tabelas | Nomes de colunas | Comentários |
| --- | --- | --- |
| | | ativa. Para consultar essa tabela, você precisa especific ar `keyspace_name` e `table_name` na cláusula `WHERE`. Para ter mais informações, consulte the section called “Usar SSL”. |

## Como criar espaços de chaves no Amazon Keyspaces

O Amazon Keyspaces executa operações de linguagem de definição de dados (DDL), como criar e excluir espaços de chaves de forma assíncrona.

Você pode monitorar o status de criação de novos espaços de chave no AWS Management Console, que indica quando um espaço de tecla está pendente ou ativo. Você também pode monitorar o status de criação de um novo espaço de chaves programaticamente usando o espaço de chaves `system_schema_mcs`. Um espaço de teclas fica visível na `system_schema_mcs keyspaces` tabela quando está pronto para uso.

O padrão de design recomendado para verificar quando um novo espaço de chaves está pronto para uso é pesquisar as tabelas `system_schema_mcs keyspaces` do Amazon Keyspaces (system_schema_mcs.*). Para obter uma lista de instruções DDL para espaços de chaves, consulte a seção the section called “Keyspaces” na referência da linguagem CQL.

A consulta a seguir mostra se um espaço de chaves foi criado com sucesso.

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

Para um espaço de chave que foi criado com sucesso, a saída da consulta é semelhante à seguinte.

```
keyspace_name | durable_writes  | replication
--------------+-----------------+--------------
   mykeyspace | true            |{...} 1 item
```

# Como trabalhar com tabelas no Amazon Keyspaces

Esta seção fornece detalhes sobre como trabalhar com tabelas no Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como criar tabelas no Amazon Keyspaces
- Trabalhando com tabelas multirregionais no Amazon Keyspaces
- Colunas estáticas no Amazon Keyspaces


## Como criar tabelas no Amazon Keyspaces

O Amazon Keyspaces executa operações de linguagem de definição de dados (DDL), como criar e excluir tabelas de forma assíncrona. Você pode monitorar o status de criação de novas tabelas no AWS Management Console, que indica quando uma tabela está pendente ou ativa. Você também pode monitorar programaticamente o status de criação de uma nova tabela usando a tabela de esquema do sistema.

Uma tabela é exibida como ativa no esquema do sistema quando está pronta para uso. O padrão de design recomendado para verificar quando uma nova tabela está pronta para uso é pesquisar as tabelas de esquema do sistema do Amazon Keyspaces (`system_schema_mcs.*`). Para obter uma lista de instruções DDL para tabelas, consulte a seção the section called “Tabelas” na referência da linguagem CQL.

A consulta a seguir mostra o status de uma tabela.

```
SELECT keyspace_name, table_name, status FROM system_schema_mcs.tables WHERE
 keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Para uma tabela que ainda está sendo criada e está pendente, a saída da consulta é semelhante a esta.

```
keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | CREATING
```

Para uma tabela que foi criada com sucesso e está ativa, a saída da consulta é semelhante à seguinte.

```
keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | ACTIVE
```

## Trabalhando com tabelas multirregionais no Amazon Keyspaces

Uma tabela multirregional deve ter a capacidade de taxa de transferência de gravação configurada de duas maneiras:

- Modo de capacidade sob demanda, medido em unidades de solicitação de gravação (WRUs)
- Modo de capacidade provisionada com escalabilidade automática, medido em unidades de capacidade de gravação (WCUs)

Você pode usar o modo de capacidade provisionada com escalabilidade automática ou o modo de capacidade sob demanda para ajudar a garantir que uma tabela multirregional tenha capacidade suficiente para realizar gravações replicadas em todas. Regiões da AWS

> **Note**
>
> A alteração do modo de capacidade da tabela em uma das regiões altera o modo de capacidade de todas as réplicas.

Por padrão, o Amazon Keyspaces usa o modo sob demanda para tabelas multirregionais. Com o modo sob demanda, você não precisa especificar a taxa de transferência de leitura e gravação que espera que seu aplicativo execute. O Amazon Keyspaces acomoda instantaneamente suas cargas de trabalho à medida que elas aumentam ou diminuem para qualquer nível de tráfego atingido anteriormente. Se o nível de tráfego de uma carga de trabalho atingir um novo pico, o Amazon Keyspaces se adapta rapidamente para acomodar a carga de trabalho.

Se você escolher o modo de capacidade provisionada para uma tabela, precisará configurar o número de unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs) por segundo que seu aplicativo exige.

Para planejar as necessidades de capacidade de taxa de transferência de uma tabela multirregional, você deve primeiro estimar o número de WCUs por segundo necessário para cada região. Em seguida, você adiciona as gravações de todas as regiões nas quais sua tabela é replicada e usa a soma para provisionar a capacidade de cada região. Isso é necessário porque cada gravação realizada em uma região também deve ser repetida em cada região de réplica.

Se a tabela não tiver capacidade suficiente para lidar com as gravações de todas as regiões, ocorrerão exceções de capacidade. Além disso, os tempos de espera de replicação inter-regional aumentarão.

Por exemplo, se você tiver uma tabela multirregional na qual espera 5 gravações por segundo no Leste dos EUA (Norte da Virgínia), 10 gravações por segundo no Leste dos EUA (Ohio) e 5 gravações por segundo na Europa (Irlanda), você deve esperar que a tabela consuma 20 WCUs em cada região: Leste dos EUA (Norte da Virgínia), Leste dos EUA (Ohio) e Europa (Irlanda). Isso significa que, neste exemplo, você precisa provisionar 20 WCUs para cada uma das réplicas da tabela. Você pode monitorar o consumo de capacidade da sua mesa usando a Amazon CloudWatch. Para ter mais informações, consulte the section called “Monitoramento com CloudWatch”.

Como cada gravação multirregional é cobrada como 1,25 vezes WCUs, você veria um total de 75 WCUs cobradas neste exemplo. Para obter mais informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

Para obter mais informações sobre a capacidade provisionada com o escalonamento automático do Amazon Keyspaces, consulte. the section called “Gerencie a capacidade de produção com escalabilidade automática”

> **Note**
>
> Se uma tabela estiver sendo executada no modo de capacidade provisionada com escalonamento automático, a capacidade de gravação provisionada poderá flutuar dentro dessas configurações de escalabilidade automática para cada região.

## Colunas estáticas no Amazon Keyspaces

Em uma tabela do Amazon Keyspaces com colunas de clustering, você pode usar a palavra-chave `STATIC` para criar uma coluna estática. O valor armazenado em uma coluna estática é compartilhado entre todas as linhas em uma partição lógica. Quando você atualiza o valor dessa coluna, o Amazon Keyspaces aplica a alteração automaticamente a todas as linhas na partição.

Esta seção descreve como calcular o tamanho codificado dos dados ao gravar em colunas estáticas. Esse processo é tratado separadamente do processo que grava dados nas colunas não estáticas de uma linha. Além das cotas de tamanho para dados estáticos, as operações de leitura e gravação em colunas estáticas também afetam a medição e capacidade de throughput das tabelas de forma independente.

## Como calcular o tamanho estático da coluna por partição lógica no Amazon Keyspaces

Esta seção fornece detalhes sobre como estimar o tamanho codificado das colunas estáticas no Amazon Keyspaces. O tamanho codificado é usado ao calcular o uso da fatura e da cota. Você também deve usar o tamanho codificado ao calcular os requisitos de capacidade de throughput provisionada para tabelas. Para calcular o tamanho codificado das colunas estáticas no Amazon Keyspaces, é possível usar as diretrizes a seguir.

- As chaves de partição podem conter até 2.048 bytes de dados. Cada coluna de chave na chave de partição requer até 3 bytes de metadados. Esses bytes de metadados contam para sua cota de tamanho de dados estáticos de 1 MB por partição. Ao calcular o tamanho de seus dados estáticos, você deve assumir que cada coluna de chave de partição usa os 3 bytes completos de metadados.
- Use o tamanho bruto dos valores de dados da coluna estática com base no tipo de dados. Para obter informações sobre tipos de dados, consulte the section called “Tipos de dados”.
- Adicione 104 bytes ao tamanho dos dados estáticos para metadados.
- Colunas de clustering e colunas de chave não primária regulares não contam para o tamanho dos dados estáticos. Para saber como estimar o tamanho dos dados não estáticos dentro das linhas, consulte the section called “Como calcular o tamanho da linha”.

O tamanho total codificado de uma coluna estática é baseado na seguinte fórmula:

```
partition key columns + static columns + metadata = total encoded size of static data
```

Considere o exemplo a seguir de uma tabela em que todas as colunas são do tipo inteiro. A tabela tem duas colunas de chave de partição, duas colunas de clustering, uma coluna regular e uma coluna estática.

```
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));
```

Neste exemplo, calculamos o tamanho dos dados estáticos da seguinte instrução:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, static_col1) values(1,2,6);
```

Para estimar o total de bytes exigidos por essa operação de gravação, você pode usar as etapas a seguir.

1. Calcule o tamanho de uma coluna de chave de partição adicionando os bytes do tipo de dados armazenado na coluna e os bytes de metadados. Repita isso para todas as colunas da chave de partição.

   a. Calcule o tamanho da primeira coluna da chave de partição (pk_col1):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   b. Calcule o tamanho da segunda coluna da chave de partição (pk_col2):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   c. Adicione as duas colunas para obter o tamanho total estimado das colunas da chave de partição:

   ```
   7 bytes + 7 bytes = 14 bytes for the partition key columns
   ```

2. Adicione o tamanho das colunas estáticas. Neste exemplo, temos apenas uma coluna estática que armazena um número inteiro (o que exige 4 bytes).
3. Por fim, para obter o tamanho total codificado dos dados da coluna estática, some os bytes das colunas da chave primária e das colunas estáticas e adicione os 104 bytes adicionais dos metadados:

   ```
   14 bytes for the partition key columns + 4 bytes for the static column + 104 bytes
    for metadata = 122 bytes.
   ```

Você também pode atualizar dados estáticos e não estáticos com a mesma instrução. Para estimar o tamanho total da operação de gravação, você deve primeiro calcular o tamanho da atualização não

estática de dados. Em seguida, calcule o tamanho da atualização da linha, conforme mostrado no exemplo em the section called “Como calcular o tamanho da linha”, e adicione os resultados.

Nesse caso, você pode gravar um total de 2 MB — 1 MB é a cota máxima de tamanho de linha e 1 MB é a cota para o tamanho máximo de dados estáticos por partição lógica.

Para calcular o tamanho total de uma atualização de dados estáticos e não estáticos na mesma instrução, você pode usar a seguinte fórmula:

```
(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
```

Considere o exemplo a seguir de uma tabela em que todas as colunas são do tipo inteiro. A tabela tem duas colunas de chave de partição, duas colunas de clustering, uma coluna regular e uma coluna estática.

```
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));
```

Neste exemplo, calculamos o tamanho dos dados quando escrevemos uma linha na tabela, conforme mostrado na instrução a seguir:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1,
 static_col1) values(2,3,4,5,6,7);
```

Para estimar o total de bytes exigidos por essa operação de gravação, você pode usar as etapas a seguir.

1. Calcule o tamanho total codificado dos dados estáticos conforme mostrado anteriormente. Neste exemplo, são 122 bytes.
2. Adicione o tamanho do tamanho total codificado da linha com base na atualização de dados não estáticos, seguindo as etapas em the section called “Como calcular o tamanho da linha”. Neste exemplo, o tamanho total da atualização da linha é 134 bytes.

   ```
   122 bytes for static data + 134 bytes for nonstatic data = 256 bytes.
   ```

### Como medir operações de leitura/gravação de dados estáticos no Amazon Keyspaces

Os dados estáticos estão associados a partições lógicas no Cassandra, não a linhas individuais. As partições lógicas no Amazon Keyspaces podem ter tamanho praticamente ilimitado, abrangendo várias partições de armazenamento físico. Como resultado, os medidores do Amazon Keyspaces gravam operações em dados estáticos e não estáticos separadamente. Além disso, gravações que incluem dados estáticos e não estáticos exigem operações subjacentes adicionais para fornecer consistência de dados.

Se você realizar uma operação de gravação mista de dados estáticos e não estáticos, isso resultará em duas operações de gravação separadas — uma para dados não estáticos e outra para dados estáticos. Isso se aplica aos modos de capacidade de leitura/gravação sob demanda e provisionada.

O exemplo a seguir fornece detalhes sobre como estimar as unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs) necessárias ao calcular os requisitos de capacidade de throughput provisionada para tabelas no Amazon Keyspaces que têm colunas estáticas. Você pode estimar quanta capacidade sua tabela precisa para processar gravações que incluam dados estáticos e não estáticos usando a seguinte fórmula:

```
2 x WCUs required for nonstatic data + 2 x WCUs required for static data
```

Por exemplo, se seu aplicativo grava 27 KBs de dados por segundo e cada gravação inclui 25,5 KBs de dados não estáticos e 1,5 KBs de dados estáticos, sua tabela exige 56 WCUs (2 x 26 WCUs + 2 x 2 WCUs).

O Amazon Keyspaces mede as leituras de dados estáticos e não estáticos da mesma forma que as leituras de várias linhas. Como resultado, o preço da leitura de dados estáticos e não estáticos na mesma operação é baseado no tamanho agregado dos dados processados para realizar a leitura.

Para saber como monitorar recursos sem servidor com a Amazon CloudWatch, consulte. the section called “Monitoramento com CloudWatch”

## Como trabalhar com linhas no Amazon Keyspaces

Esta seção fornece detalhes sobre como trabalhar com linhas no Amazon Keyspaces (para Apache Cassandra). As tabelas são as principais estruturas de dados no Amazon Keyspaces e os dados nas tabelas são organizados em colunas e linhas.

Tópicos

- Como calcular o tamanho da linha no Amazon Keyspaces


# Como calcular o tamanho da linha no Amazon Keyspaces

O Amazon Keyspaces fornece armazenamento totalmente gerenciado que oferece desempenho de leitura e gravação de milissegundos de um dígito e armazena dados de forma durável em várias zonas de disponibilidade AWS. O Amazon Keyspaces anexa metadados a todas as linhas e colunas de chave primária para oferecer suporte ao acesso eficiente aos dados e à alta disponibilidade.

Esta seção fornece detalhes sobre como estimar o tamanho codificado das linhas no Amazon Keyspaces. O tamanho codificado das linhas é usado ao calcular o uso da fatura e da cota. Você também deve usar o tamanho codificado das linhas ao calcular os requisitos de capacidade de throughput provisionada para tabelas. Para calcular o tamanho codificado das linhas no Amazon Keyspaces, é possível usar as diretrizes a seguir.

- Para colunas regulares, que não são chaves primárias, colunas de clustering ou colunas `STATIC`, use o tamanho bruto dos dados da célula com base no tipo de dados e adicione os metadados necessários. Para obter mais informações sobre os tipos de dados suportados no Amazon Keyspaces, consulte the section called “Tipos de dados”. Algumas diferenças importantes na forma como o Amazon Keyspaces armazena valores de tipos de dados e metadados estão listadas abaixo.
- O espaço necessário para cada nome de coluna é armazenado usando um identificador de coluna e adicionado a cada valor de dados armazenado na coluna. O valor de armazenamento do identificador da coluna depende do número geral de colunas em sua tabela:
  - 1 a 62 colunas: 1 byte
  - 63 a 124 colunas: 2 bytes
  - 125 a 186 colunas: 3 bytes

  Para cada 62 colunas adicionais, adicione 1 byte. Observe que, no Amazon Keyspaces, até 225 colunas regulares podem ser modificadas com uma única instrução `INSERT` ou `UPDATE`. Para obter mais informações, consulte the section called “Service Quotas do Amazon Keyspaces”.
- As chaves de partição podem conter até 2.048 bytes de dados. Cada coluna de chave na chave de partição requer até 3 bytes de metadados. Ao calcular o tamanho da sua linha, você deve assumir que cada coluna de chave de partição usa os 3 bytes completos de metadados.
- As colunas de clustering podem armazenar até 850 bytes de dados. Além do tamanho do valor dos dados, cada coluna de clustering exige até 20% do tamanho do valor dos dados para metadados.

Ao calcular o tamanho da sua linha, você deve adicionar 1 byte de metadados para cada 5 bytes do valor dos dados da coluna de clustering.

- O Amazon Keyspaces armazena o valor dos dados de cada chave de partição e coluna de chave de clustering duas vezes. A sobrecarga extra é usada para consultas eficientes e indexação integrada.
- Os tipos de dados string Cassandra `ASCII`, `TEXT` e `VARCHAR` são todos armazenados no Amazon Keyspaces usando Unicode com codificação binária UTF-8. O tamanho de uma string no Amazon Keyspaces é igual ao número de bytes codificados em UTF-8.
- Os tipos de dados Cassandra `INT`, `BIGINT`, `SMALLINT` e `TINYINT` são armazenados no Amazon Keyspaces como valores de dados com comprimento variável, com até 38 dígitos significativos. Zeros iniciais e finais são cortados. O tamanho de qualquer um desses tipos de dados é de aproximadamente 1 byte por dois dígitos significativos + 1 byte.
- Um `BLOB` no Amazon Keyspaces é armazenado com o comprimento bruto do byte do valor.
- O tamanho de um valor `Null` ou valor `Boolean` é de 1 byte.
- Uma coluna que armazena tipos de dados de coleção como `LIST` ou `MAP` exige 3 bytes de metadados, independentemente de seu conteúdo. O tamanho de um `LIST` ou `MAP` é (id da coluna) + soma (tamanho dos elementos aninhados) + (3 bytes). O tamanho de um `LIST` ou `MAP` vazio é (id da coluna) + (3 bytes). Cada elemento individual `LIST` ou `MAP` também exige 1 byte de metadados.
- Os dados da coluna `STATIC` não contam para o tamanho máximo da linha de 1 MB. Para calcular o tamanho dos dados das colunas estáticas, consulte the section called “Como calcular o tamanho estático da coluna por partição lógica”.
- Os registros de data/hora do lado do cliente são armazenados para cada coluna em cada linha quando o atributo é ativado. Esses carimbos de data/hora ocupam aproximadamente 20 a 40 bytes (dependendo dos seus dados) e contribuem para o custo de armazenamento e throughput da linha. Para obter mais informações, consulte the section called “Carimbos de data/hora do lado do cliente no Amazon Keyspaces”.
- Adicione 100 bytes ao tamanho de cada linha para os metadados da linha.

O tamanho total de uma linha de dados codificada é baseado na seguinte fórmula:

```
partition key columns + clustering columns + regular columns + row metadata = total
 encoded size of row
```

> **Important**
>
> Todos os metadados da coluna, por exemplo, ids de coluna, metadados de chave de partição, metadados de colunas de agrupamento, bem como registros de data/hora do lado do cliente e metadados de linha, contam para o tamanho máximo de linha de 1 MB.

Considere o exemplo a seguir de uma tabela em que todas as colunas são do tipo inteiro. A tabela tem duas colunas de chave de partição, duas colunas de clustering e uma coluna regular. Como essa tabela tem cinco colunas, o espaço necessário para o identificador do nome da coluna é de 1 byte.

```
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));
```

Neste exemplo, calculamos o tamanho dos dados quando escrevemos uma linha na tabela, conforme mostrado na instrução a seguir:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1)
 values(1,2,3,4,5);
```

Para estimar o total de bytes exigidos por essa operação de gravação, você pode usar as etapas a seguir.

1. Calcule o tamanho de uma coluna de chave de partição adicionando os bytes do tipo de dados armazenado na coluna e os bytes de metadados. Repita isso para todas as colunas da chave de partição.

   a. Calcule o tamanho da primeira coluna da chave de partição (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. Calcule o tamanho da segunda coluna da chave de partição (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. Adicione as duas colunas para obter o tamanho total estimado das colunas da chave de partição:

```
8 bytes + 8 bytes = 16 bytes for the partition key columns
```

2. Calcule o tamanho de uma coluna de clustering adicionando os bytes do tipo de dados armazenado na coluna e os bytes de metadados. Repita isso para todas as colunas de clustering.

a. Calcule o tamanho da primeira coluna 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. Calcule o tamanho da segunda coluna 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. Adicione as duas colunas para obter o tamanho total estimado das colunas de clustering:

```
6 bytes + 6 bytes = 12 bytes for the clustering columns
```

3. Adicione o tamanho das colunas regulares. Neste exemplo, temos apenas uma coluna que armazena um número inteiro de um único dígito, o que requer 2 bytes com 1 byte para o id da coluna.

4. Por fim, para obter o tamanho total da linha codificada, some os bytes para todas as colunas e adicione os 100 bytes adicionais para os metadados da linha:

```
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.
```

Para saber como monitorar recursos de tecnologia sem servidor com o Amazon CloudWatch, consulte the section called “Monitoramento com CloudWatch”.

# Como trabalhar com consultas no Amazon Keyspaces

Esta seção fornece uma introdução ao trabalho com consultas no Amazon Keyspaces (para Apache Cassandra). As instruções CQL disponíveis para consultar, transformar e gerenciar dados são `SELECT`, `INSERT`, `UPDATE` e `DELETE`. Os tópicos a seguir descrevem algumas das opções mais complexas disponíveis ao se trabalhar com consultas. Para obter a sintaxe completa da linguagem com exemplos, consulte the section called “Instruções DML”.

Tópicos


- Como usar o operador IN com a Instrução SELECT no Amazon Keyspaces
- Como ordenar resultados no Amazon Keyspaces
- Como paginar resultados no Amazon Keyspaces


## Como usar o operador **`IN`** com a Instrução **`SELECT`** no Amazon Keyspaces

SELECT IN

Você pode consultar dados de tabelas usando a instrução `SELECT`, que lê uma ou mais colunas para uma ou mais linhas em uma tabela e retorna um conjunto de resultados contendo as linhas correspondentes à solicitação. Uma instrução `SELECT` contém uma `select_clause` que determina quais colunas devem ser lidas e retornadas no conjunto de resultados. A cláusula pode conter instruções para transformar os dados antes de retorná-los. A cláusula opcional `WHERE` especifica quais linhas devem ser consultadas e é composta por relações nas colunas que fazem parte da chave primária. O Amazon Keyspaces oferece suporte à palavra-chave `IN` na cláusula `WHERE`. Esta seção usa exemplos para mostrar como o Amazon Keyspaces processa instruções `SELECT` com a palavra-chave `IN`.

Esses exemplos demonstram como o Amazon Keyspaces divide a instrução `SELECT` com a palavra-chave `IN` em subconsultas. Neste exemplo, usamos uma tabela com o nome `my_keyspace.customers`. A tabela tem uma coluna de chave primária `department_id`, duas colunas de clustering `sales_region_id` e `sales_representative_id` e uma coluna que contém o nome do cliente na coluna `customer_name`.

```
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
```

Usando essa tabela, você pode executar a instrução `SELECT` a seguir para encontrar os clientes nos departamentos e regiões de vendas nos quais você está interessado com a palavra-chave `IN` na cláusula `WHERE`. A instrução a seguir é um exemplo disso.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1) AND sales_region_id
 IN (0, 1);
```

O Amazon Keyspaces divide essa instrução em quatro subconsultas, conforme mostrado na saída a seguir.

```
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
```

Quando a palavra-chave `IN` é usada, o Amazon Keyspaces pagina automaticamente os resultados em qualquer um dos seguintes casos:

- Depois que cada 10ª subconsulta for processada.
- Depois de processar 1MB de E/S lógica.
- Se você configurou um `PAGE SIZE`, o Amazon Keyspaces pagina depois de ler o número de consultas para processamento com base no conjunto `PAGE SIZE`.
- Quando você usa a palavra-chave `LIMIT` para reduzir o número de linhas retornadas, o Amazon Keyspaces pagina depois de ler o número de consultas para processamento com base no conjunto `LIMIT`.

A tabela a seguir é usada para ilustrar isso com um exemplo.

Para obter mais informações sobre paginação, consulte the section called “Como paginar resultados”.

```
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
```

Você pode executar a instrução a seguir nessa tabela para ver como a paginação funciona.

```
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);
```

O Amazon Keyspaces processa essa declaração como 24 subconsultas, porque a cardinalidade do produto cartesiano de todos os termos `IN` contidos nessa consulta é 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
```

Este exemplo mostra como você pode usar a cláusula `ORDER BY` em uma instrução `SELECT` com a palavra-chave `IN`.

```
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
```

As subconsultas são processadas na ordem em que as colunas da chave de partição e da chave clustering são apresentadas na consulta. No exemplo abaixo, as subconsultas para o valor da chave de partição “2” são processadas primeiro, seguidas pelas subconsultas para o valor da chave de partição “3” e “1”. Os resultados de uma determinada subconsulta são ordenados de acordo com a cláusula de ordenação da consulta, se presente, ou com a ordem de clustering da tabela definida durante sua criação.

```
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
```

## Como ordenar resultados no Amazon Keyspaces

A cláusula `ORDER BY` especifica a ordem de classificação dos resultados retornados em uma instrução `SELECT`. A instrução usa uma lista de nomes de colunas como argumentos e, para cada coluna, você pode especificar a ordem de classificação dos dados. Você só pode especificar colunas de clustering em cláusulas de ordenação; colunas sem clustering não são permitidas.

As duas opções de ordem de classificação disponíveis para os resultados retornados são `ASC` para ordem de classificação crescente e `DESC` para decrescente.

```
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
```

Se você não especificar a ordem de classificação na instrução de consulta, a ordem padrão da coluna de clustering será usada.

As possíveis ordens de classificação que você pode usar em uma cláusula de ordenação dependem da ordem de classificação atribuída a cada coluna de clustering na criação da tabela. Os resultados da consulta só podem ser classificados na ordem definida para todas as colunas de clustering na criação da tabela ou no inverso da ordem de classificação definida. Outras combinações possíveis não são permitidas.

Por exemplo, se a `CLUSTERING ORDER` da tabela for (col1 ASC, col2 DESC, col3 ASC), os parâmetros válidos para `ORDER BY` serão (col1 ASC, col2 DESC, col3 ASC) ou (col1 DESC, col2 ASC, col3 DESC). Para obter mais informações sobre a `CLUSTERING ORDER`, consulte `table_options` em the section called “CRIAR TABELA”.

## Como paginar resultados no Amazon Keyspaces

O Amazon Keyspaces pagina automaticamente os resultados das instruções `SELECT` quando os dados lidos para processar a instrução `SELECT` excedem 1 MB. Com a paginação, os resultados da instrução `SELECT` são divididos em "páginas" de dados com 1 MB de tamanho (ou menos). Uma aplicação pode processar a primeira página de resultados e, em seguida, a segunda página, e assim por diante. Os clientes devem sempre verificar os tokens de paginação ao processar consultas `SELECT` que retornam várias linhas.

Se um cliente fornecer um `PAGE SIZE` que exija a leitura de mais de 1 MB de dados, o Amazon Keyspaces divide os resultados automaticamente em várias páginas com base nos incrementos de 1 MB de leitura de dados.

Por exemplo, se o tamanho médio de uma linha for 100 KB e você especificar um `PAGE SIZE` de 20, o Amazon Keyspaces pagina os dados automaticamente depois de ler 10 linhas (1000 KB de dados lidos).

Como o Amazon Keyspaces pagina os resultados com base no número de linhas que lê para processar uma solicitação e não no número de linhas retornadas no conjunto de resultados, algumas páginas podem não conter nenhuma linha se você estiver executando consultas filtradas.

Por exemplo, se você definir `PAGE SIZE` como 10 e o Keyspaces avaliar 30 linhas para processar sua consulta `SELECT`, o Amazon Keyspaces retornará três páginas. Se apenas um subconjunto das linhas corresponder à sua consulta, algumas páginas poderão ter menos de 10 linhas. Para ver um exemplo de como `LIMIT` as consultas `PAGE SIZE` of podem afetar a capacidade de leitura, consultethe section called “Limitar consultas”.

# Como trabalhar com particionadores no Amazon Keyspaces

No Apache Cassandra, os particionadores controlam em quais nós os dados são armazenados no cluster. Os particionadores criam um token numérico usando um valor de hash da chave de partição. O Cassandra usa esse token para distribuir dados entre os nós. Os clientes também podem usar esses tokens em operações `SELECT` e cláusulas `WHERE` para otimizar as operações de leitura e gravação. Por exemplo, os clientes podem realizar consultas paralelas de forma eficiente em tabelas grandes especificando intervalos de tokens distintos a serem consultados em cada trabalho paralelo.

O Amazon Keyspaces fornece três particionadores diferentes.

Murmur3Partitioner (padrão)

> Compatível com Apache Cassandra `Murmur3Partitioner`. O `Murmur3Partitioner` é o particionador Cassandra padrão no Amazon Keyspaces e no Cassandra 1.2 e versões posteriores.

RandomPartitioner

> Compatível com Apache Cassandra `RandomPartitioner`. O `RandomPartitioner` é o particionador Cassandra padrão para versões anteriores ao Cassandra 1.2.

Particionador padrão do Keyspaces

O `DefaultPartitioner` retorna os mesmos resultados da função `token` que o `RandomPartitioner`.

A configuração do particionador é aplicada por região no nível da conta. Por exemplo, se você alterar o particionador no Leste dos EUA (Norte da Virgínia), a alteração será aplicada a todas as tabelas na mesma conta nessa Região. É possível alterar seu particionador com segurança a qualquer momento. Observe que a alteração da configuração leva cerca de 10 minutos para ser concluída. Você não precisa recarregar seus dados do Amazon Keyspaces ao alterar a configuração do particionador. Os clientes usarão automaticamente a nova configuração do particionador na próxima vez que se conectarem.

Você pode alterar o particionador usando o Cassandra Query Language (CQL). AWS Management Console

AWS Management Console

Para alterar o particionador usando o console Amazon Keyspaces

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, escolha Configuration (Configuração).
3. Na página Configuração, acesse Editar particionador.
4. Selecione o particionador compatível com sua versão do Cassandra. Essa alteração do particionador leva aproximadamente 10 minutos para ser aplicada.

> **Note**
>
> Depois que a alteração da configuração for concluída, você deverá se desconectar e reconectar ao Amazon Keyspaces para solicitar o uso do novo particionador.

Cassandra Query Language (CQL)

1. Para ver qual particionador está configurado para a conta, você pode usar a consulta a seguir.

```
SELECT partitioner from system.local;
```

Se o particionador não tiver sido alterado, a consulta terá a seguinte saída.

```
partitioner
-----------------------------------------
com.amazonaws.cassandra.DefaultPartitioner
```

2. Para atualizar o particionador para o particionador `Murmur3`, você pode usar a seguinte declaração.

```
UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

3. Observe que esta alteração da configuração leva cerca de 10 minutos para ser concluída. Para confirmar que o particionador foi definido, você pode executar a consulta `SELECT` novamente. Observe que, devido à eventual consistência de leitura, a resposta pode ainda não refletir os resultados da alteração recém-concluída do particionador. Se você repetir a operação `SELECT` novamente após um curto período de tempo, a resposta deverá retornar os dados mais recentes.

```
SELECT partitioner from system.local;
```

> **Note**
>
> Você precisa se desconectar e reconectar ao Amazon Keyspaces para que as solicitações usem o novo particionador.

# Trabalhando com tags e rótulos para recursos do Amazon Keyspaces

Você pode rotular recursos do Amazon Keyspaces (para Apache Cassandra) usando tags. As tags permitem categorizar recursos de diferentes maneiras, por exemplo, por finalidade, proprietário, ambiente ou outros critérios. As tags podem ajudar a fazer o seguinte:

- Identificar rapidamente um recurso com base nas tags que você atribuiu a ele.

- Veja AWS as contas divididas por etiquetas.
- Controle o acesso aos recursos do Amazon Keyspaces com base em tags. Para ver exemplos de política do IAM usando tags, consulte the section called “Autorização baseada em tags do Amazon Keyspaces”.

A marcação é suportada por AWS serviços como Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Keyspaces e muito mais. Uma marcação eficiente é capaz de fornecer insights de custos, permitindo criar relatórios entre serviços que possuem uma tag específica.

Para começar a usar tags, faça o seguinte:

1. Compreender Restrições de marcação para o Amazon Keyspaces.
2. Criar tags usando Operações de marcação para o Amazon Keyspaces.
3. Use Relatórios de alocação de custos para Amazon Keyspaces para monitorar seus AWS custos por tag ativa.

Por fim, é recomendável seguir estratégias de marcação ideais. Para obter informações, consulte Estratégias de marcação da AWS.

## Restrições de marcação para o Amazon Keyspaces

Cada tag consiste em uma chave e um valor, ambos definidos por você. As seguintes restrições são aplicáveis:

- Cada espaço de chaves ou tabela do Amazon Keyspaces pode ter apenas uma tag com a mesma chave. Se você tentar adicionar uma tag existente (mesma chave), o valor da tag existente será atualizado para o novo valor.
- As tags aplicadas a um espaço de chaves não se aplicam automaticamente às tabelas dentro desse espaço de chaves. Para aplicar a mesma tag a um espaço de chaves e a todas as suas tabelas, cada recurso deve ser marcado individualmente.
- Quando você cria uma tabela ou espaço de chaves multirregional, todas as tags que você define durante o processo de criação são aplicadas automaticamente a todos os espaços de chaves e tabelas em todas as Regiões. Quando você altera as tags existentes usando `ALTER KEYSPACE` ou`ALTER TABLE`, a atualização é aplicada somente ao espaço de chaves ou à tabela na Região em que você está fazendo a alteração.

- Um valor atua como um descritor dentro de uma categoria de tag (chave). No Amazon Keyspaces, o valor não pode estar em branco nem ser nulo.
- As chaves e os valores de tags diferenciam maiúsculas de minúsculas.
- O comprimento máximo da chave é 128 caracteres Unicode.
- O comprimento máximo do valor é 256 caracteres Unicode.
- Os caracteres permitidos são letras, espaço em branco e números, além dos seguintes caracteres especiais: `+ - = . _ : /`
- O número máximo de tags por recurso é 50.
- Nomes e valores de tags atribuídos pela AWS recebem automaticamente o prefixo `aws:`, o qual não pode ser atribuído por você. Nomes de tags atribuídos pela AWS não contam para o limite de tag de recurso definido pelo usuário de 50. Nomes de tags atribuídos pelo usuário têm o prefixo `user:` no relatório de alocação de custos.
- Não é possível colocar uma data retroativa na aplicação de uma tag.

## Operações de marcação para o Amazon Keyspaces

Você pode adicionar, listar, editar ou excluir tags para espaços de chaves e tabelas usando o console Amazon Keyspaces (para Apache Cassandra), a CLI da AWS ou o Cassandra Query Language (CQL). Em seguida, você pode ativar essas tags definidas pelo usuário para que elas apareçam no console do AWS Billing and Cost Management para o controle da alocação de custos. Para obter mais informações, consulte Relatórios de alocação de custos para Amazon Keyspaces.

Para edição em massa, também é possível usar o Tag Editor no console. Para obter mais informações, consulte Como trabalhar com o Tag Editor no Guia do usuário do AWS Resource Groups.

Tópicos

- Adição de tags a espaços de chaves e tabelas novos ou existentes usando o console
- Adição de tags a espaços de chaves e tabelas novos ou existentes usando o CLI da AWS
- Adição de tags a espaços de chaves e tabelas novos ou existentes usando o CQL

### Adição de tags a espaços de chaves e tabelas novos ou existentes usando o console

Você pode usar o console do Amazon Keyspaces para adicionar tags a novos espaços de chaves e tabelas ao criá-los. Você também pode adicionar, editar ou excluir tags para tabelas existentes.

Para marcar espaços de chaves ao criá-los (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Espaços de chaves e, em seguida, escolha Criar espaço de chaves.
3. Na página Criar espaço de chaves, forneça um nome para o espaço de chaves. Insira uma chave e um valor para a tag e selecione Adicionar nova tag.
4. Selecione Criar espaço de chaves.

Para marcar tabelas ao criá-las (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas) e Create table (Criar tabela).
3. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a tabela.
4. Na seção Esquema, crie o esquema para sua tabela.
5. Na seção Configurações da tabela, selecione Personalizar configurações.
6. Vá até a seção Tags de tabela – opcional e selecione Adicionar nova tag para criar novas tags.
7. Escolha Create table.

Para marcar recursos existentes (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Espaços de chaves ou Tabelas.
3. Na lista, selecione um espaço de chaves ou uma tabela. Em seguida, selecione Gerenciar tags para adicionar, editar ou excluir tags.

Para obter informações sobre a estrutura da tag, consulte Restrições de marcação para o Amazon Keyspaces.

## Adição de tags a espaços de chaves e tabelas novos ou existentes usando o CLI da AWS

Os exemplos nesta seção demonstram como usar o CLI da AWS para especificar tags ao criar espaços de chaves e tabelas, como adicionar ou remover tags de recursos existentes e como listar tags.

O exemplo a seguir mostra como criar uma nova tabela com tags. O comando cria uma tabela myTable em um espaço de chaves myKeyspace existente. Observe que o comando foi dividido em linhas diferentes para ajudar na legibilidade.

```
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'
```

O seguinte exemplo mostra como adicionar novas tags à tabela existente.

```
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'
```

O próximo exemplo mostra como listar as tags do recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111222333444:/keyspace/myKeyspace/table/myTable'
```

A saída do último comando tem a aparência a seguir.

```
{
    "tags": [
        {
            "key": "key1",
            "value": "val1"
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
```

```
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

## Adição de tags a espaços de chaves e tabelas novos ou existentes usando o CQL

Os exemplos a seguir mostram como usar o CQL para especificar tags ao criar espaços de chaves e tabelas, como marcar recursos existentes e como ler tags.

O exemplo a seguir cria um novo espaço de chaves com tags.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'} ;
```

O exemplo a seguir cria uma nova tabela com tags.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

Para marcar recursos em uma instrução com outros comandos.

```
CREATE KEYSPACE mykeyspace WITH REPLICATION = {'class': 'Simple Strategy'} AND TAGS
 = {'key1':'val1', 'key2':'val2'};
```

O exemplo a seguir mostra como adicionar ou excluir tags a espaços de chaves ou tabelas existentes.

```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

```
ALTER TABLE mytable DROP TAGS {'key1':'val1', 'key2':'val2'};
```

Para ler as tags anexadas a um recurso, use a instrução CQL a seguir.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

A cláusula WHERE é obrigatória e deve ter um dos seguintes formatos:

- keyspace_name = **'mykeyspace'** AND resource_type = 'keyspace'
- keyspace_name = **'mykeyspace'** AND resource_name = **'mytable'**
- resource_id = *arn*

Exemplos:

A consulta a seguir mostra se um espaço de chaves tem tags.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_type = 'keyspace';
```

A saída da consulta é semelhante ao seguinte.

```
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'}
```

A consulta a seguir mostra as tags de uma tabela.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_name = 'mytable';
```

A saída dessa consulta é semelhante à seguinte.

```
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'}
```

## Relatórios de alocação de custos para Amazon Keyspaces

A AWS usa tags para organizar os custos de recursos no seu relatório de alocação de custos. A AWS fornece dois tipos de tags alocação de custos:

- Uma tag gerada pela AWS. A AWS define, cria e aplica essa tag para você.
- Tags definidas pelo usuário. Você define, cria e aplica essas tags.

É necessário ativar os dois tipos de tags separadamente para que elas possam ser exibidas no Cost Explorer ou em um relatório de alocação de custos.

Para ativar tags geradas pela AWS:

1. Faça login no AWS Management Console e abra o console do Billing and Cost Management em https://console.aws.amazon.com/billing/home#/.
2. No painel de navegação, escolha Cost Allocation Tags.
3. Em AWSGenerated Cost Allocation Tags (Etiquetas de alocação de custos geradas), escolha Activate (Ativar).

Para ativar tags definidas pelo usuário:

1. Faça login no AWS Management Console e abra o console do Billing and Cost Management em https://console.aws.amazon.com/billing/home#/.
2. No painel de navegação, escolha Cost Allocation Tags.
3. Em Tags de alocação de custos definidos pelo usuário, escolha Ativar.

Após você criar e ativar tags, a AWS gera um relatório de alocação de custos com a utilização e os custos agrupados de acordo com as tags ativas. O relatório de alocação de custos inclui todos os seus custos da AWS para cada período de faturamento. O relatório inclui recursos com e sem tags, para que você possa organizar claramente as cobranças de cada um deles.

> **Note**
>
> Atualmente, qualquer dado transferido do Amazon Keyspaces não é categorizado por tags em relatórios de alocação de custos.

Para obter mais informações, consulte Usar tags de alocação de custos.

# Práticas recomendadas de design e arquitetura com o Amazon Keyspaces

Use esta seção para localizar rapidamente as recomendações para maximizar a performance e minimizar os custos de throughput para trabalhar com o Amazon Keyspaces.

Sumário


- Design em NoSQL para Amazon Keyspaces
  - Diferenças entre design relacional de dados e NoSQL
  - Dois conceitos-chave para design em NoSQL
  - Abordar o design em NoSQL
- Conexões do driver do cliente com o Amazon Keyspaces (para Apache Cassandra)
  - Como as conexões funcionam no Amazon Keyspaces
  - Como configurar conexões no Amazon Keyspaces
  - Como configurar conexões por meio de endpoints da VPC no Amazon Keyspaces
  - Como monitorar conexões no Amazon Keyspaces
  - Como lidar com erros de conexão no Amazon Keyspaces
- Modelagem de dados no Amazon Keyspaces (para Apache Cassandra)
  - Como usar chaves de partição de forma eficaz no Amazon Keyspaces
    - Usar fragmentação de gravação para distribuir workloads uniformemente no Amazon Keyspaces
      - Fragmentação usando chaves de partição compostas e valores randomizados
      - Fragmentação usando chaves de partição compostas e valores calculados
- Como otimizar os custos das tabelas do Amazon Keyspaces
  - Avaliar seus custos no nível da tabela
    - Como visualizar os custos de uma única tabela do Amazon Keyspaces
    - Visualização padrão do Explorador de Custos
    - Como usar e aplicar tags de tabela no Explorador de Custos
  - Avaliar o modo de capacidade de sua tabela
    - Quais modos de capacidade de tabela estão disponíveis
    - Quando selecionar o modo de capacidade sob demanda

    - Quando selecionar o modo de capacidade provisionada
    - Fatores adicionais a serem considerados ao escolher um modo de capacidade de tabela
- Avalie as configurações do Application Auto Scaling da sua tabela
    - Entenda as configurações do Application Auto Scaling
    - Como identificar tabelas com baixa utilização prevista (<= 50%)
    - Como lidar com cargas de trabalho com variação sazonal
    - Como lidar com cargas de trabalho com pico com padrões desconhecidos
    - Como lidar com cargas de trabalho com aplicações vinculadas
- Identifique seus recursos não utilizados
    - Como identificar recursos não utilizados
    - Identificar recursos de tabela não utilizados
    - Limpar recursos de tabela não utilizados
    - Limpando backups de point-in-time recuperação não utilizados (PITR)
- Avaliar seus padrões de uso de tabelas
    - Realizar menos operações de leitura altamente consistente
    - Habilitar a vida útil (TTL)
- Avaliar sua capacidade provisionada para o provisionamento do tamanho certo
    - Como recuperar métricas de consumo em suas tabelas do Amazon Keyspaces
    - Como identificar tabelas do Amazon Keyspaces subprovisionadas
    - Como identificar tabelas superprovisionadas do Amazon Keyspaces


# Design em NoSQL para Amazon Keyspaces

Os sistemas de bancos de dados NoSQL, como o Amazon Keyspaces, usam os modelos alternativos para o gerenciamento de dados, como pares de chave-valor ou armazenamento de documentos. Ao mudar de um sistema de gerenciamento de banco de dados relacional de dados relacionais para um sistema de banco de dados NoSQL como o Amazon Keyspaces, é importante compreender as principais diferenças e as abordagens específicas de design.

Tópicos


- Diferenças entre design relacional de dados e NoSQL
- Dois conceitos-chave para design em NoSQL

- Abordar o design em NoSQL

## Diferenças entre design relacional de dados e NoSQL

Os sistemas de bancos de dados relacionais (RDBMS) e os bancos de dados NoSQL têm diferentes pontos fortes e fracos:

- No RDBMS, as consultas de dados são flexíveis, mas têm um custo relativamente alto e não escalam com facilidade em situações de grande volume de tráfego (consulte the section called “Modelagem de dados”).
- Em um banco de dados NoSQL como o Amazon Keyspaces, há formas limitadas de consultar dados com eficiência. As demais formas de consulta podem apresentar alto custo e baixa performance.

Essas diferenças tornam o design de banco de dados diferente entre os dois sistemas:

- Em RDBMS, você cria o design para obter flexibilidade sem se preocupar com detalhes de implementação ou desempenho. A otimização de consultas geralmente não afeta o design do esquema, mas a normalização é importante.
- No Amazon Keyspaces, você projeta o esquema especificamente para fazer as consultas mais comuns e importantes do modo mais rápido e barato possível. Suas estruturas de dados são adaptadas aos requisitos específicos de seus casos de uso de negócios.

## Dois conceitos-chave para design em NoSQL

O design do NoSQL exige uma visão diferente daquela no design do RDBMS. Para um RDBMS, você pode criar um modelo de dados normalizado sem pensar nos padrões de acesso. Você poderá estendê-lo posteriormente quando surgirem novas perguntas e requisitos de consulta. Você pode organizar cada tipo de dados em sua própria tabela.

Como o design de NoSQL é diferente

- Em comparação, você não deve começar a projetar o esquema do Amazon Keyspaces até que saiba quais as perguntas que ele precisa responder. É essencial compreender os problemas de negócios e os casos de uso de aplicativo antecipadamente.
- Você deve manter o mínimo de tabelas possível em um aplicativo do Amazon Keyspaces. Com menos tabelas, há mais escalabilidade, menos gerenciamento de permissões e menor sobrecarga

para o Amazon Keyspaces. Isso também pode ajudar a manter os custos de backup mais baixos em geral.

## Abordar o design em NoSQL

A primeira etapa ao projetar a aplicação do Amazon Keyspaces é identificar os padrões específicos de consulta aos quais o sistema deve atender.

Especificamente, é importante compreender três propriedades fundamentais de padrões de acesso de seu aplicativo antes de começar:

- Tamanho de dados: saber o volume de dados que serão armazenados e solicitados ao mesmo tempo ajuda a determinar a maneira mais eficiente de particionar os dados.
- Forma dos dados: em vez de remodelar dados quando uma consulta é processada (como um sistema RDBMS faz), um banco de dados NoSQL organiza os dados para que sua forma no banco de dados corresponda ao que será consultado. Esse é um fator importante no aumento da velocidade e da escalabilidade.
- Velocidade dos dados: o Amazon Keyspaces é escalado aumentando-se o número de partições físicas que estão disponíveis para processar consultas e distribuindo-se os dados com eficiência entre essas partições. Saber antecipadamente qual é o pico das cargas de consulta pode ajudar a determinar como particionar os dados para melhor utilizar a capacidade de E/S.

Após identificar os requisitos específicos da consulta, você pode organizar dados de acordo com os princípios gerais que regem o desempenho:

- Mantenha os dados relacionados juntos.  Uma pesquisa sobre a otimização de tabelas de rotas há 20 anos descobriu que a "localidade de referência" era o único fator o mais importante para agilizar o tempo de resposta: manter dados relacionados juntos em um só lugar. Isso também se aplica aos sistemas NoSQL hoje, em que manter dados relacionados juntos tem um impacto significativo no custo e no desempenho. Em vez da distribuição de itens de dados relacionados entre várias tabelas, você deve manter itens relacionados no sistema de NoSQL o mais próximo possível.

  Como regra geral, você deve manter o mínimo de tabelas possível em um aplicativo do Amazon Keyspaces.

  As exceções são os casos que envolvem dados de séries temporais de alto volume ou conjuntos de dados que têm padrões muito diferentes de acesso. Uma única tabela com índices invertidos

pode normalmente habilitar consultas simples para criar e recuperar estruturas de dados hierárquicas e complexas, exigidas pelo aplicativo.

- Use a ordem de classificação. Os itens relacionados podem ser agrupados juntos e consultados de modo eficiente se o design da chave fizer com que sejam classificados juntos. Essa é uma estratégia importante de design do NoSQL.
- Distribua as consultas. É importante também que um alto volume de consultas não se concentre em uma parte do banco de dados, onde podem exceder a capacidade de E/S. Em vez disso, você deve projetar chaves de dados para distribuir o tráfego entre as partições do modo mais uniforme possível, evitando “pontos de atividade”.

Esses princípios gerais traduzem-se em alguns padrões comuns de design que você pode usar para modelar dados no Amazon Keyspaces com eficiência.

# Conexões do driver do cliente com o Amazon Keyspaces (para Apache Cassandra)

Para se comunicar com o Amazon Keyspaces, você pode usar qualquer um dos drivers de cliente Apache Cassandra existentes de sua escolha. Como o Amazon Keyspaces é um serviço de tecnologia sem servidor, recomendamos que você otimize a configuração de conexão do driver de cliente de acordo com as necessidades de throughput do seu aplicativo. Este tópico apresenta as práticas recomendadas, incluindo como calcular quantas conexões seu aplicativo requer, bem como monitoramento e tratamento de erros de conexões.

Tópicos

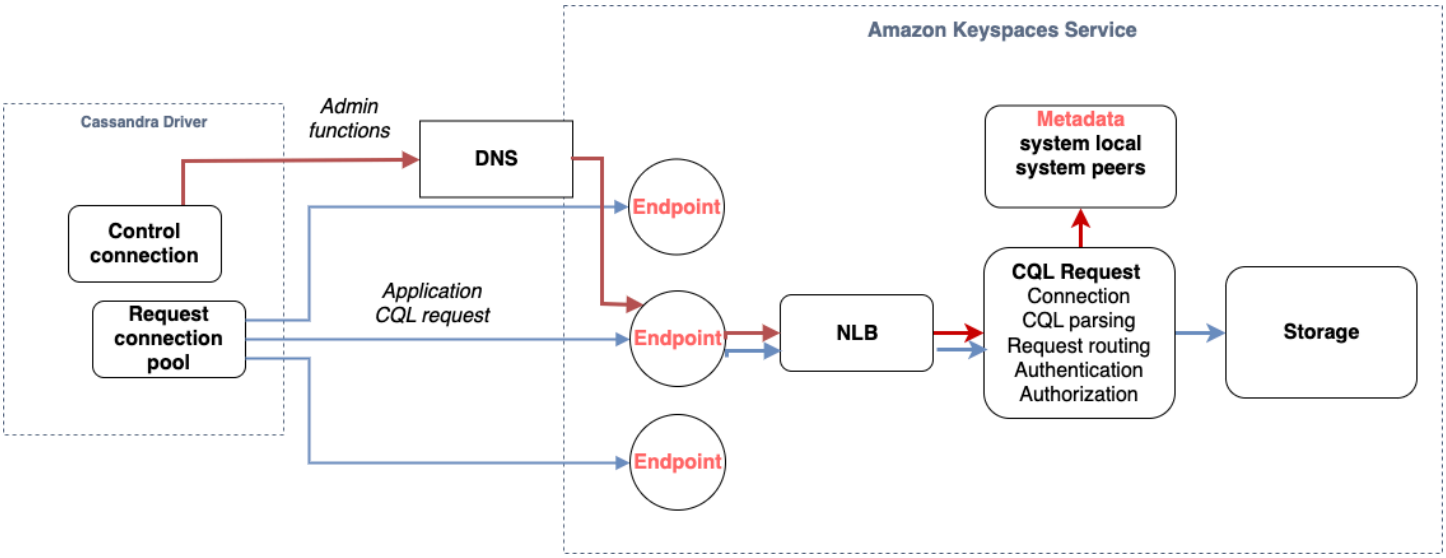

- Como as conexões funcionam no Amazon Keyspaces
- Como configurar conexões no Amazon Keyspaces
- Como configurar conexões por meio de endpoints da VPC no Amazon Keyspaces
- Como monitorar conexões no Amazon Keyspaces
- Como lidar com erros de conexão no Amazon Keyspaces


## Como as conexões funcionam no Amazon Keyspaces

Esta seção fornece uma visão geral de como as conexões de driver de clientes funcionam no Amazon Keyspaces. Como a configuração incorreta do driver de cliente Cassandra pode resultar em eventos de `PerConnectionRequestExceeded` no Amazon Keyspaces, é necessário configurar a

quantidade certa de conexões na configuração do driver de cliente para evitar esses e outros erros de conexão semelhantes.

Ao se conectar ao Amazon Keyspaces, o driver precisa de um endpoint inicial para estabelecer uma conexão inicial. O Amazon Keyspaces usa o DNS para rotear a conexão inicial para um dos muitos endpoints disponíveis. Os endpoints são conectados a Network Load Balancers que, por sua vez, estabelecem uma conexão com um dos manipuladores de solicitações na frota. Depois que a conexão inicial é estabelecida, o driver de cliente coleta informações sobre todos os endpoints disponíveis na tabela `system.peers`. Com essas informações, o driver de cliente pode criar conexões adicionais com os endpoints listados. O número de conexões que o driver de cliente pode criar é limitado pelo número de conexões locais especificadas nas configurações do driver de cliente. Por padrão, a maioria dos drivers de clientes estabelece uma conexão por endpoint e estabelece um pool de conexões com o Cassandra e balanceia a carga de consultas sobre esse pool de conexões. Embora várias conexões possam ser estabelecidas no mesmo endpoint, por trás do Network Load Balancer elas podem estar conectadas a vários manipuladores de solicitações diferentes. Ao se conectar por meio do endpoint público, estabelecer uma conexão com cada um dos nove endpoints listados na tabela `system.peers` resulta em nove conexões com diferentes manipuladores de solicitações.




## Como configurar conexões no Amazon Keyspaces

O Amazon Keyspaces suporta até 3.000 consultas CQL por conexão TCP por segundo. Como não há limite no número de conexões que um driver pode estabelecer, recomendamos direcionar apenas 500 solicitações de CQL por segundo por conexão para permitir sobrecarga, picos de tráfego e melhor balanceamento de carga. Siga estas etapas para garantir que a conexão do driver esteja configurada corretamente para as necessidades do seu aplicativo.

Aumente o número de conexões por endereço IP que seu driver mantém em seu pool de conexões.

- A maioria dos drivers do Cassandra estabelece um pool de conexão com o Cassandra e balanceia a carga das consultas sobre esse pool de conexões. O comportamento padrão da maioria dos drivers é estabelecer uma única conexão com cada endpoint. O Amazon Keyspaces expõe nove endereços IP de mesmo nível aos drivers. Portanto, com base no comportamento padrão da maioria dos drivers, isso resulta em 9 conexões. O Amazon Keyspaces suporta até 3.000 consultas CQL por conexão TCP por segundo, portanto, a taxa máxima de throughput de consultas CQL de um driver usando as configurações padrão é 27.000 consultas CQL por segundo. Se você usar as configurações padrão do driver, uma única conexão pode precisar processar mais do que o throughput máximo de consultas CQL de 3.000 consultas CQL por segundo. Isso pode resultar em eventos `PerConnectionRequestExceeded`.
- Para evitar eventos `PerConnectionRequestExceeded`, você deve configurar o driver para criar conexões adicionais por endpoint para distribuir o throughput.
- Como melhor prática no Amazon Keyspaces, suponha que cada conexão possa suportar 500 consultas CQL por segundo.
- Isso significa que, para um aplicativo de produção que precisa suportar cerca de 27.000 consultas CQL por segundo distribuídas pelos nove endpoints disponíveis, você deve configurar seis conexões por endpoint. Isso garante que cada conexão processe no máximo 500 solicitações por segundo.

Calcule o número de conexões por endereço IP que você precisa configurar para seu driver com base nas necessidades do seu aplicativo.

Para saber o número de conexões que você precisa configurar por endpoint para seu aplicativo, considere o exemplo a seguir. Você tem um aplicativo que precisa oferecer suporte a 20.000 consultas CQL por segundo, consistindo em 10.000 `INSERT`, 5.000 `SELECT` e 5.000 operações `DELETE`. O aplicativo Java está sendo executado em três instâncias no Amazon Elastic Container Service (Amazon ECS), onde cada instância estabelece uma única sessão no Amazon Keyspaces. O cálculo que você pode usar para estimar quantas conexões você precisa configurar para seu driver usa a seguinte entrada.

1. O número de solicitações por segundo que seu aplicativo precisa oferecer suporte.
2. O número de instâncias disponíveis com uma subtraída para contabilizar manutenção ou falha.

3. O número de endpoints disponíveis. Se você estiver se conectando por meio de endpoints públicos, você tem nove endpoints disponíveis. Se você estiver usando VPC endpoints, terá entre dois e cinco endpoints disponíveis, dependendo da região.
4. Use 500 consultas CQL por segundo por conexão como uma prática recomendada para o Amazon Keyspaces.
5. Arredonde o resultado.

Neste exemplo, a fórmula tem a seguinte aparência.

```
20,000 CQL queries / (3 instances - 1 failure) / 9 public endpoints / 500 CQL queries
 per second = ROUND(2.22) = 3
```

Com base nesse cálculo, você precisa especificar três conexões locais por endpoint na configuração do driver. Para conexões remotas, configure somente uma conexão por endpoint.

## Como configurar conexões por meio de endpoints da VPC no Amazon Keyspaces

Ao se conectar por meio de endpoints VPC privados, você provavelmente tem 3 endpoints disponíveis. O número de VPC endpoints pode ser diferente por região, com base no número de zonas de disponibilidade e no número de sub-redes na VPC atribuída. A região Leste dos EUA (Norte da Virgínia) tem cinco zonas de disponibilidade, e você pode ter até cinco endpoints do Amazon Keyspaces. A região Oeste dos EUA (Norte da Califórnia) tem duas zonas de disponibilidade, e você pode ter até dois endpoints do Amazon Keyspaces. O número de endpoints não afeta a escala, mas aumenta o número de conexões que você precisa estabelecer na configuração do driver. Considere o seguinte exemplo. Seu aplicativo precisa suportar 20.000 consultas CQL e está sendo executado em três instâncias no Amazon ECS, onde cada instância estabelece uma única sessão no Amazon Keyspaces. A única diferença é quantos endpoints estão disponíveis nos diferentes Regiões da AWS.

Conexões necessárias na região Leste dos EUA (Norte da Virgínia):

```
20,000 CQL queries / (3 instances - 1 failure) / 5 private VPC endpoints / 500 CQL
 queries per second = 4 local connections
```

Conexões necessárias na região Oeste dos EUA (Norte da Califórnia):

```
20,000 CQL queries / (3 instances - 1 failure) / 2 private VPC endpoints / 500 CQL
 queries per second = 10 local connections
```

> **Important**
>
> Ao usar endpoints da VPC privados, permissões adicionais são necessárias para que o Amazon Keyspaces descubra dinamicamente os endpoints da VPC disponíveis e preencha a tabela `system.peers`. Para ter mais informações, consulte the section called “Como preencher entradas da tabela `system.peers` com informações do endpoint da VPC de interface”.

Ao acessar o Amazon Keyspaces por meio de um endpoint VPC privado usando outro Conta da AWS, é provável que você veja apenas um único endpoint do Amazon Keyspaces. Novamente, isso não afeta a escala da possível throughput para o Amazon Keyspaces, mas pode exigir que você aumente o número de conexões na configuração do driver. Este exemplo mostra o mesmo cálculo para um único endpoint disponível.

```
20,000 CQL queries / (3 instances - 1 failure) / 1 private VPC endpoints / 500 CQL
 queries per second = 20 local connections
```

Para saber mais sobre o acesso entre contas ao Amazon Keyspaces usando uma VPC compartilhada, consulte the section called “Acesso entre contas em uma VPC compartilhada”.

## Como monitorar conexões no Amazon Keyspaces

Para ajudar a identificar o número de endpoints aos quais seu aplicativo está conectado, você pode registrar o número de pares descobertos na tabela `system.peers`. O exemplo a seguir é um exemplo de código Java que imprime o número de pares após o estabelecimento da conexão.

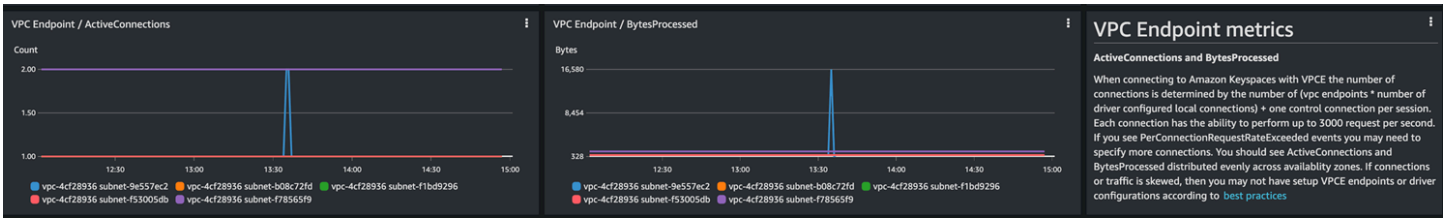
```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

> **Note**
>
> O console ou AWS o console CQL não são implantados em uma VPC e, portanto, usam o endpoint público. Como resultado, a execução da consulta `system.peers` a partir de

aplicativos localizados fora do VPCE geralmente resulta em 9 pares. Também pode ser útil imprimir os endereços IP de cada par.

Você também pode observar o número de pares ao usar um VPC endpoint configurando as métricas do VPCE Amazon. CloudWatch Em CloudWatch, você pode ver o número de conexões estabelecidas com o VPC endpoint. Os drivers do Cassandra estabelecem uma conexão para cada endpoint para enviar consultas CQL e uma conexão de controle para coletar informações da tabela do sistema. A imagem abaixo mostra as CloudWatch métricas do VPC endpoint depois de se conectar ao Amazon Keyspaces com 1 conexão configurada nas configurações do driver. A métrica mostra seis conexões ativas que consistem em uma conexão de controle e cinco conexões (1 por endpoint nas zonas de disponibilidade).




Para começar a monitorar o número de conexões usando um CloudWatch gráfico, você pode implantar esse AWS CloudFormation modelo disponível GitHub no repositório de modelos do Amazon Keyspaces.

## Como lidar com erros de conexão no Amazon Keyspaces

Ao exceder a cota de 3.000 solicitações por conexão, o Amazon Keyspaces retorna um evento `PerConnectionRequestExceeded` e o driver do Cassandra recebe uma exceção `WriteTimeout` ou `ReadTimeout`. Você deve repetir essa exceção com um recuo exponencial em sua política de repetição do Cassandra ou em seu aplicativo. Você deve fornecer um recuo exponencial para evitar o envio de solicitações adicionais.

A política de repetição padrão tenta o `try next host` no plano de consulta. Como o Amazon Keyspaces pode ter de um a três endpoints disponíveis ao se conectar ao endpoint da VPC, você também pode ver o `NoHostAvailableException` além das exceções `WriteTimeout` e `ReadTimeout` nos logs do seu aplicativo. Você pode usar as políticas de repetição fornecidas pelo Amazon Keyspaces, que tentam novamente no mesmo endpoint, mas em conexões diferentes.

Você pode encontrar exemplos de políticas de repetição exponencial para Java GitHub no repositório de exemplos de código Java do Amazon Keyspaces. Você pode encontrar outros exemplos de idiomas no Github no repositório de exemplos de código do Amazon Keyspaces.

# Modelagem de dados no Amazon Keyspaces (para Apache Cassandra)

Este tópico apresenta conceitos de modelagem de dados no Amazon Keyspaces (para Apache Cassandra). Use esta seção para encontrar recomendações para criar modelos de dados que se alinhem aos padrões de acesso aos dados do seu aplicativo. A implementação das práticas recomendadas de modelagem de dados melhora o desempenho e minimiza os custos de throughput ao trabalhar com o Amazon Keyspaces.

Para visualizar e projetar modelos de dados com mais facilidade, você pode usar o NoSQL Workbench.

Tópicos


- Como usar chaves de partição de forma eficaz no Amazon Keyspaces


## Como usar chaves de partição de forma eficaz no Amazon Keyspaces

A chave primária que identifica exclusivamente cada linha em uma tabela do Amazon Keyspaces pode consistir em uma ou várias colunas de chave de partição, que determinam em quais partições os dados são armazenados, e uma ou mais colunas de cluster opcionais, que definem como os dados são agrupados e classificados dentro de uma partição.

Como a chave de partição estabelece o número de partições em que seus dados são armazenados e como os dados são distribuídos entre essas partições, a forma como você escolhe sua chave de partição pode ter um impacto significativo no desempenho de suas consultas. Em geral, seu aplicativo deve ser projetado para ter uma atividade uniforme em todas as partições no disco.

Distribuir uniformemente a atividade de leitura e gravação do seu aplicativo em todas as partições ajuda a minimizar os custos de throughput, e isso se aplica aos modos de capacidade de leitura/gravação sob demanda e provisionada. Por exemplo, se você estiver usando o modo de capacidade provisionada, é possível determinar os padrões de acesso que o aplicativo precisa e estimar o total de unidades de capacidade de leitura (RCU) e unidades de capacidade de gravação (WCU) que cada tabela requer. O Amazon Keyspaces é compatível com seus padrões de acesso usando o throughput provisionado, contanto que o tráfego em uma determinada partição não exceda 3.000 RCUs e 1.000 WCUs.

O Amazon Keyspaces oferece flexibilidade adicional no provisionamento de throughput por partição ao oferecer capacidade de expansão. Para obter mais informações, consulte the section called "Capacidade de expansão".

Tópicos


- Usar fragmentação de gravação para distribuir workloads uniformemente no Amazon Keyspaces


## Usar fragmentação de gravação para distribuir workloads uniformemente no Amazon Keyspaces

Uma forma de distribuir melhor as gravações por uma partição no Amazon Keyspaces é aumentar o espaço. É possível fazer isso de várias formas diferentes. Você pode adicionar uma coluna de chave de partição adicional na qual você grava números randomizados para distribuir as linhas entre as partições. Ou pode usar um número calculado com base em algo que você está consultando.

Fragmentação usando chaves de partição compostas e valores randomizados

Uma estratégia para distribuir cargas de forma mais uniforme em uma partição é adicionar uma coluna de chave de partição adicional na qual você grava números randomizados. Então, você randomiza as gravações no espaço maior.

Por exemplo, considere a tabela a seguir, que tem uma única chave de partição representando uma data.

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   title text,
   description int,
   PRIMARY KEY (publish_date));
```

Para distribuir mais uniformemente essa tabela entre as partições, você pode incluir uma coluna adicional de chave de partição `shard` que armazene números randomizados. Por exemplo: .

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   shard int,
   title text,
   description int,
   PRIMARY KEY ((publish_date, shard)));
```

Ao inserir dados, escolha um número randomizado entre 1 e 200 para a coluna `shard`. Isso produz valores de chave de partição compostos como `(2020-07-09, 1)`, `(2020-07-09, 2)` e assim por diante, até `(2020-07-09, 200)`. Como você está randomizando a chave de partição, as gravações na tabela em cada dia são espalhadas uniformemente entre várias partições. Isso resulta em melhor paralelismo e throughput geral mais alto.

Contudo, para ler todas as linhas de um determinado dia, você teria que consultar as linhas quanto a todos os fragmentos e mesclar os resultados. Por exemplo, você primeiro emite uma instrução de `SELECT` do valor da chave de partição `(2020-07-09, 1)`. Depois emite outra instrução `SELECT` de `(2020-07-09, 2)`, e assim por diante, por meio de `(2020-07-09, 200)`. Por fim, seu aplicativo precisaria mesclar os resultados de todas essas instruções `SELECT`.

Fragmentação usando chaves de partição compostas e valores calculados

Uma estratégia de randomização pode melhorar bastante o throughput de gravação. Mas é difícil ler uma linha específica porque você não sabe qual valor foi gravado na coluna `shard` quando a linha foi gravada. Para facilitar a leitura de linhas individuais, você pode usar uma estratégia diferente. Em vez de usar um número aleatório para distribuir as linhas entre as partições, use um número calculado com base em algo que você deseja consultar.

Considere o exemplo anterior, em que uma tabela usa a data de hoje na chave de partição. Agora considere que cada linha conta com uma coluna `title` acessível e que você precisa com frequência de localizar as linhas pelo título, além da data. Antes de seu aplicativo gravar a linha na tabela, ele pode calcular um valor de hash com base no título e usá-lo para preencher a coluna `shard`. O cálculo pode gerar um número entre 1 e 200 que é distribuído uniformemente, semelhante à estratégia aleatória.

Bastaria um cálculo simples, como o produto dos valores de pontos de código UTF-8 para os caracteres no título, módulo 200, + 1. O valor da chave de partição composta seria então a combinação da data e do resultado do cálculo.

Com essa estratégia, as gravações são distribuídas uniformemente entre os valores de chaves de partição e, portanto, entre as partições físicas. É possível executar uma instrução `SELECT` facilmente para uma determinada linha e data, pois você pode calcular o valor de chave da partição de um valor `title` específico.

Para ler todos as linhas de um determinado dia, você ainda precisa realizar uma operação `SELECT` em cada uma das chaves `(2020-07-09, N)` (em que `N` é de 1 a 200), e sua aplicação então deve mesclar todos os resultados. O benefício é evitar de ter um valor de chave de partição "hot" único consumindo toda a carga de trabalho.

# Como otimizar os custos das tabelas do Amazon Keyspaces

Esta seção aborda as práticas recomendadas sobre como otimizar os custos das tabelas do Amazon Keyspaces. Você deve examinar as estratégias a seguir para ver qual estratégia de otimização de custos é a melhor para atender às suas necessidades e abordá-las de forma iterativa. Cada estratégia fornece uma visão geral do que pode estar afetando seus custos, como procurar oportunidades para otimizar custos e orientações prescritivas sobre como implementar essas práticas recomendadas para ajudar a economizar.

Tópicos


- Avaliar seus custos no nível da tabela
- Avaliar o modo de capacidade de sua tabela
- Avalie as configurações do Application Auto Scaling da sua tabela
- Identifique seus recursos não utilizados
- Avaliar seus padrões de uso de tabelas
- Avaliar sua capacidade provisionada para o provisionamento do tamanho certo


## Avaliar seus custos no nível da tabela

A ferramenta Cost Explorer encontrada no AWS Management Console permite que você veja os custos divididos por tipo, por exemplo, taxas de leitura, gravação, armazenamento e backup. Também é possível ver esses custos resumidos por período, como mês ou dia.

Um desafio comum com o Cost Explorer é que você não pode revisar facilmente os custos de apenas uma tabela específica, porque o Cost Explorer não permite filtrar ou agrupar por custos de uma tabela específica. Você pode visualizar o tamanho métrico da tabela faturável (bytes) de cada tabela no console do Amazon Keyspaces na guia Monitor da tabela. Se você precisar de mais informações relacionadas ao custo por tabela, esta seção mostra como usar a marcação para realizar análises individuais de custos de tabelas no Cost Explorer.

Tópicos


- Como visualizar os custos de uma única tabela do Amazon Keyspaces
- Visualização padrão do Explorador de Custos
- Como usar e aplicar tags de tabela no Explorador de Custos

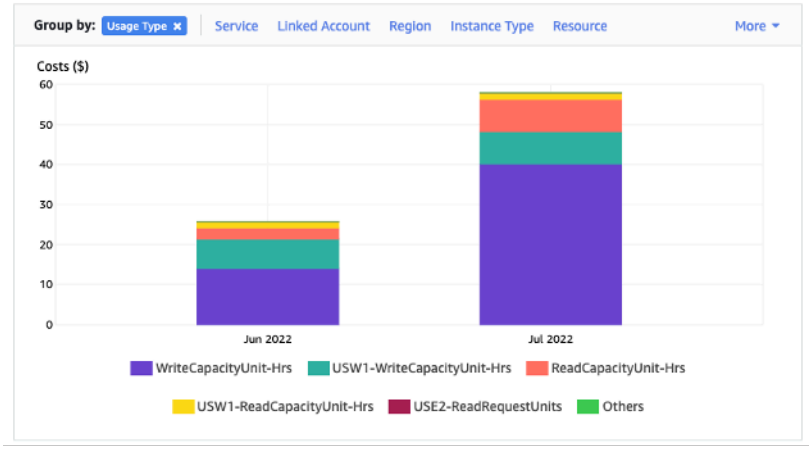
## Como visualizar os custos de uma única tabela do Amazon Keyspaces

Você pode ver informações básicas sobre uma tabela do Amazon Keyspaces no console, incluindo o esquema da chave primária, o tamanho da tabela faturável e as métricas relacionadas à capacidade. Você pode usar o tamanho da tabela para calcular o custo mensal de armazenamento da tabela. Por exemplo, 0,25 USD por GB no Leste dos EUA (Norte da Virgínia). Região da AWS

Se a tabela estiver no modo de capacidade provisionada, as configurações atuais de RCU (Read Capacity Unit) e WCU(Write Capacity Unit) também serão retornadas. Você pode usar essas informações para calcular os custos atuais de leitura e gravação da tabela. Observe que esses custos podem mudar, especialmente se você tiver configurado a tabela com a escalabilidade automática do Amazon Keyspaces.

## Visualização padrão do Explorador de Custos

A visualização padrão do Explorador de Custos fornece gráficos que mostram o custo dos recursos consumidos, como throughput e armazenamento. É possível optar por agrupar estes custos por período, como totais por mês ou por dia. Os custos de armazenamento, leituras, gravações e outras categorias também podem ser divididos e comparados.




## Como usar e aplicar tags de tabela no Explorador de Custos

Por padrão, o Explorador de Custos não fornece um resumo dos custos de nenhuma tabela específica, pois combina os custos de várias tabelas em um total. No entanto, é possível usar a marcação de recursos da AWS para identificar cada tabela por uma tag de metadados. As tags são pares de chave-valor que podem ser usadas para diversas finalidades, como identificar todos os recursos pertencentes a um projeto ou departamento. Para ter mais informações, consulte the section called “Trabalhando com tags”.

Neste exemplo, usamos uma tabela com o nome MyTable.

1. Defina uma tag com a chave de table_name e o valor de. MyTable
2. Ative a tag no Explorador de Custos e filtre o valor da tag para obter mais visibilidade dos custos de cada tabela.

> **Note**
>
> Pode levar um ou dois dias para que a tag comece a aparecer no Explorador de Custos.

Você mesmo pode definir tags de metadados no console ou programaticamente com o CQL AWS CLI, o ou o SDK. AWS Considere exigir que uma tag table_name seja definida como parte do processo de criação de tabelas da sua organização. Para ter mais informações, consulte the section called “Relatórios de alocação de custos para Amazon Keyspaces”.

## Avaliar o modo de capacidade de sua tabela

Esta seção apresenta uma visão geral de como selecionar o modo de capacidade adequado para a tabela do Amazon Keyspaces. Cada modo é ajustado para atender às necessidades de uma carga de trabalho diferente em termos de capacidade de resposta a mudanças no throughput, bem como de como esse uso é cobrado. Você deve ponderar sobre esses fatores ao tomar sua decisão.

Tópicos

- Quais modos de capacidade de tabela estão disponíveis
- Quando selecionar o modo de capacidade sob demanda
- Quando selecionar o modo de capacidade provisionada
- Fatores adicionais a serem considerados ao escolher um modo de capacidade de tabela

### Quais modos de capacidade de tabela estão disponíveis

Ao criar uma tabela do Amazon Keyspaces, é necessário selecionar o modo de capacidade sob demanda ou provisionada. Para ter mais informações, consulte the section called “Modos de capacidade de leitura/gravação”.

Modo de capacidade sob demanda

O modo de capacidade sob demanda foi projetado para eliminar a necessidade de planejar ou provisionar a capacidade da tabela do Amazon Keyspaces. Nesse modo, a tabela atende instantaneamente às solicitações de acomodação feitas sem a necessidade de escalar recursos (até o dobro do throughput máximo anterior da tabela).

As tabelas sob demanda são cobradas pela contagem do número de solicitações reais à tabela. Portanto, você paga apenas pelo que usar e não pelo que foi provisionado.

Modo de capacidade provisionada

O modo de capacidade provisionada é um modelo mais tradicional em que é possível definir que capacidade a tabela tem disponível para solicitações feitas diretamente ou com a ajuda do Application Auto Scaling. Como uma capacidade específica é provisionada para a tabela a qualquer momento, o faturamento baseia-se na capacidade provisionada e não no número de solicitações. Ultrapassar a capacidade alocada também pode fazer com que a tabela rejeite solicitações e reduza a experiência dos usuários de seu aplicativo.

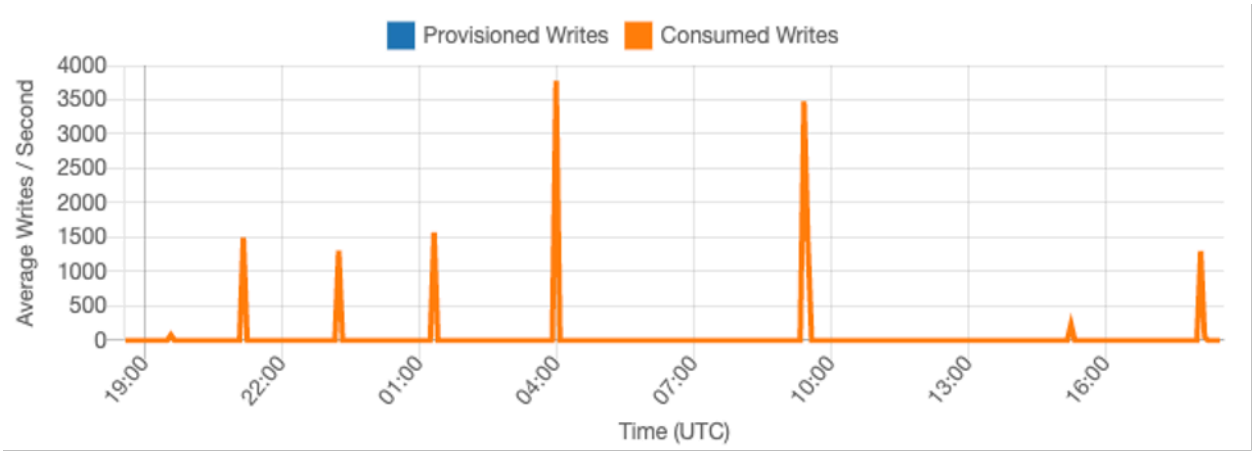
O modo de capacidade provisionada exige um equilíbrio entre não superprovisionar ou subprovisionar a tabela para obter a baixa ocorrência de erros de capacidade de throughput insuficiente e os custos otimizados.

## Quando selecionar o modo de capacidade sob demanda

Para otimizar o custo, o modo sob demanda é a melhor opção quando se tem uma carga de trabalho semelhante ao gráfico a seguir.

Esses fatores contribuem para esse tipo de carga de trabalho:

- Tempo de solicitação imprevisível (resultando em picos de tráfego)
- Volume variável de solicitações (resultante de cargas de trabalho em lote)
- Queda para zero ou abaixo de 18% do pico em determinada hora (resultante de ambientes de desenvolvimento ou teste)




Em workloads com as características acima, usar o Application Auto Scaling para manter a capacidade suficiente para que a tabela responda aos picos de tráfego pode levar a resultados indesejáveis. Ou a tabela pode estar superprovisionada e custar mais do que o necessário, ou a tabela pode estar subprovisionada e as solicitações estão causando erros desnecessários de throughput de baixa capacidade. Em casos como esse, as mesas sob demanda são a melhor opção.

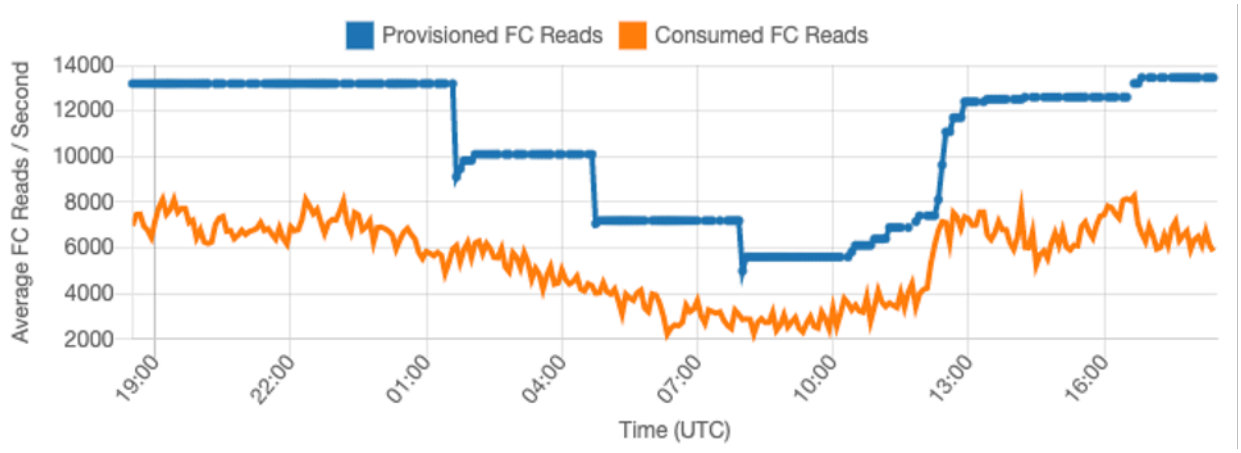
Como as tabelas sob demanda são cobradas por solicitação, não há mais nada a fazer no nível da tabela para otimizar o custo. É necessário avaliar regularmente as tabelas sob demanda para verificar se a carga de trabalho ainda tem as características acima. Caso a carga de trabalho esteja estabilizada, considere a possibilidade de mudar para o modo provisionado a fim de otimizar ainda mais os custos.

## Quando selecionar o modo de capacidade provisionada

Uma carga de trabalho ideal para o modo de capacidade provisionada é aquela com um padrão de uso mais previsível, como o gráfico abaixo.

Os seguintes fatores contribuem para esse tipo de carga de trabalho previsível:

- Tráfego previsível/cíclico em determinada hora ou determinado dia
- Intermitências limitadas de tráfego de curto prazo




Como os volumes de tráfego em determinada hora ou determinado dia são mais estáveis, podemos definir a capacidade provisionada relativamente próxima à capacidade real consumida. A otimização de custos de uma tabela de capacidade provisionada é, em última análise, um exercício para obter a capacidade provisionada (linha azul) o mais próximo possível da capacidade consumida (linha laranja) sem aumentar os eventos de `ThrottledRequests` na tabela. O espaço entre as duas linhas representa tanto capacidade desperdiçada quanto garantia contra uma experiência inadequada do usuário devido aos erros de capacidade de throughput.

O Amazon Keyspaces fornece Application Auto Scaling para tabelas de capacidade provisionada que equilibrarão isso automaticamente por você. Isso permite que você acompanhe a capacidade consumida ao longo do dia e configure a capacidade provisionada da tabela com base em algumas variáveis.

Unidades de capacidade mínima

É possível definir a capacidade mínima de uma tabela para limitar a ocorrência de erros de capacidade de throughput insuficiente, mas isso não reduz o custo da tabela. Se a tabela tiver períodos de baixo uso seguidos de uma expansão repentina de alto uso, definir o mínimo poderá impedir que o Application Auto Scaling defina a capacidade da tabela com um valor muito baixo.

Unidades de capacidade máxima

É possível definir a capacidade máxima de uma tabela para impedir que a tabela escale acima do pretendido. Considere a possibilidade de aplicar um máximo para tabelas de desenvolvimento ou teste em que os testes de carga em grande escala não são desejados. É possível definir um máximo para qualquer tabela, mas avalie regularmente essa configuração em relação à linha de base da tabela ao usá-la na produção para evitar erros de capacidade de throughput insuficiente.

Utilização pretendida

Definir a utilização pretendida da tabela é o principal meio de otimização de custos para uma tabela de capacidade provisionada. Definir um valor percentual mais baixo aqui aumenta o superprovisionamento da tabela, o que eleva os custos, mas reduz os erros de capacidade de throughput insuficiente. Definir um valor percentual mais alto diminui o quanto a tabela está superprovisionada, mas aumenta o risco de erros de capacidade de throughput insuficiente.

## Fatores adicionais a serem considerados ao escolher um modo de capacidade de tabela

Ao decidir entre os dois modos de capacidade, há alguns fatores adicionais que vale a pena considerar.

Ao decidir entre os dois modos de tabela, considere quanto esse desconto adicional afeta o custo da tabela. Em muitos casos, até mesmo uma carga de trabalho relativamente imprevisível pode ser mais barata de ser executada em uma tabela de capacidade provisionada em excesso com capacidade reservada.

Melhorar a previsibilidade da carga de trabalho

Em algumas situações, uma carga de trabalho pode aparentemente ter um padrão previsível e imprevisível. Embora isso possa ser facilmente atendido com uma tabela sob demanda, os custos provavelmente serão mais baixos se for possível melhorar os padrões imprevisíveis da carga de trabalho.

Uma das causas mais comuns para esses padrões são as importações em lote. Esse tipo de tráfego geralmente pode exceder a capacidade básica da tabela a tal ponto que erros de capacidade de throughput insuficiente ocorreriam se ela fosse executada. Para manter uma carga de trabalho como essa em execução em uma tabela de capacidade provisionada, considere as seguintes opções:

- Se o lote ocorrer em horários programados, será possível programar um aumento na capacidade de ajuste de escala automático do seu aplicativo antes da execução.
- Caso o lote ocorra aleatoriamente, considere a possibilidade de tentar estender o tempo de execução em vez de executá-lo o mais rápido possível.
- Adicione um período de aceleração à importação no qual a velocidade da importação começa pequena, mas aumenta lentamente em alguns minutos até que o ajuste de escala automático do seu aplicativo tenha a oportunidade de começar a ajustar a capacidade da tabela.

# Avalie as configurações do Application Auto Scaling da sua tabela

Esta seção apresenta uma visão geral de como avaliar as configurações de Application Auto Scaling nas tabelas do Amazon Keyspaces. O Application Auto Scaling do Amazon Keyspaces é um atributo que gerencia o throughput da tabela com base no tráfego do seu aplicativo e na métrica de utilização desejada. Isso garante que suas tabelas tenham a capacidade necessária para os padrões de seu aplicativo.

O serviço Application Auto Scaling monitora a utilização atual da tabela e compara com o valor de utilização pretendido: `TargetValue`. Ele te notifica se for hora de aumentar ou diminuir a capacidade alocada.

Tópicos


- Entenda as configurações do Application Auto Scaling
- Como identificar tabelas com baixa utilização prevista (<= 50%)
- Como lidar com cargas de trabalho com variação sazonal
- Como lidar com cargas de trabalho com pico com padrões desconhecidos
- Como lidar com cargas de trabalho com aplicações vinculadas


## Entenda as configurações do Application Auto Scaling

Definir o valor correto para a utilização pretendida, a etapa inicial e os valores finais é uma atividade que exige o envolvimento de sua equipe de operações. Isso permite que você defina adequadamente os valores com base no histórico de uso do aplicativo, que será usado para acionar as políticas de Application Auto Scaling. A utilização prevista é a porcentagem de sua capacidade total que deve ser atingida durante um período antes que as regras do Application Auto Scaling sejam aplicadas.

Quando você define uma utilização prevista alta (uma utilização de cerca de 90%), isso significa que seu tráfego deve ser superior a 90% por um período de tempo antes que o Application Auto Scaling seja ativado. Você não deve usar uma alta utilização prevista, a menos que sua aplicação seja muito constante e não receba picos de tráfego.

Quando você define uma utilização prevista muito baixa (uma utilização inferior a 50%), isso significa que seu aplicativo deveria atingir 50% da capacidade provisionada antes de acionar uma política de Application Auto Scaling. A menos que o tráfego da sua aplicação cresça a uma taxa muito agressiva, isso geralmente se traduz em capacidade não utilizada e desperdício de recursos.

## Como identificar tabelas com baixa utilização prevista (<= 50%)

Você pode usar o AWS CLI ou AWS Management Console para monitorar e identificar as políticas do `TargetValues` Application Auto Scaling em seus recursos do Amazon Keyspaces:

AWS CLI

1. Retorne a lista completa de recursos executando o seguinte comando:

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra
```

   Esse comando retornará a lista completa de políticas de Application Auto Scaling emitidas para qualquer recurso do Amazon Keyspaces. Se quiser apenas recuperar os recursos de uma tabela específica, você poderá adicionar o `–resource-id parameter`. Por exemplo: .

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra --resource-id "keyspace/keyspace-name/table/table-name”
```

2. Retorne somente as políticas de ajuste de escala automático para uma tabela específica executando o seguinte comando

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra --resource-id "keyspace/keyspace-name/table/table-name”
```

   Os valores nas políticas de Application Auto Scaling estão destacados abaixo. É necessário garantir que o valor alvo seja maior que 50% para evitar o provisionamento excessivo. Você deverá obter um resultado semelhante ao seguinte:

```
{
    "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. Faça login no AWS Management Console e navegue até a página de CloudWatch serviço em Introdução ao AWS Management Console. Selecione o apropriado, Região da AWS se necessário.
2. Na barra de navegação à esquerda, selecione Tabelas. Na página Tabelas, selecione o Nome da tabela.

3. Na página Detalhes da tabela, na guia Capacidade, revise as configurações de Application Auto Scaling da sua tabela.

Se os valores de suas metas de utilização forem menores ou iguais a 50%, você deverá explorar as métricas de utilização da tabela para ver se elas estão subprovisionadas ou superprovisionadas.

## Como lidar com cargas de trabalho com variação sazonal

Considere o seguinte cenário: sua aplicação está operando abaixo de um valor médio mínimo na maioria das vezes, mas a meta de utilização é baixa para que sua aplicação possa reagir rapidamente a eventos que acontecem em determinadas horas do dia e você tenha capacidade suficiente e evite ter controle de utilização. Esse cenário é comum quando você tem uma aplicação muito movimentada durante o horário normal de expediente (das 9h às 17h), mas funciona em um nível básico após o expediente. Como alguns usuários começam a se conectar antes das 9h, o aplicativo usa esse limite baixo para aumentar rapidamente a capacidade necessária durante os horários de pico.

Esse cenário pode ser parecido com:

- Entre 17h e 9h, as unidades `ConsumedWriteCapacityUnits` ficam entre 90 e 100
- Os usuários começam a se conectar à aplicação antes das 9h e as unidades de capacidade aumentam consideravelmente (o valor máximo que você viu é 1,5 mil WCU)
- Em média, o uso da sua aplicação varia entre 800 e 1.200 durante o horário comercial

Se o cenário anterior for aplicável, considere usar o Application Auto Scaling programado, em que sua tabela ainda pode ter uma regra de Application Auto Scaling de aplicativos configurada, mas com uma utilização pretendida menos agressiva que forneça somente a capacidade extra nos intervalos específicos necessários.

Você pode usar o AWS CLI para executar as etapas a seguir para criar uma regra de escalonamento automático programada que seja executada com base na hora do dia e no dia da semana.

1. Registre sua tabela do Amazon Keyspaces como um destino escalável com Application Auto Scaling. Um destino escalável é um recurso cuja escala pode ser aumentada ou reduzida horizontalmente pelo Application Auto Scaling .

```
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. Configure ações programadas de acordo com seus requisitos.

   São necessárias duas regras para cobrir o cenário: uma para aumentar e outra para reduzir a escala verticalmente. A primeira regra para aumentar a escala verticalmente da ação programada é mostrada no exemplo a seguir.

```
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"
```

   A segunda regra para reduzir a escala verticalmente da ação programada é mostrada nesse exemplo.

```
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. Execute o seguinte comando para validar que ambas as regras foram ativadas:

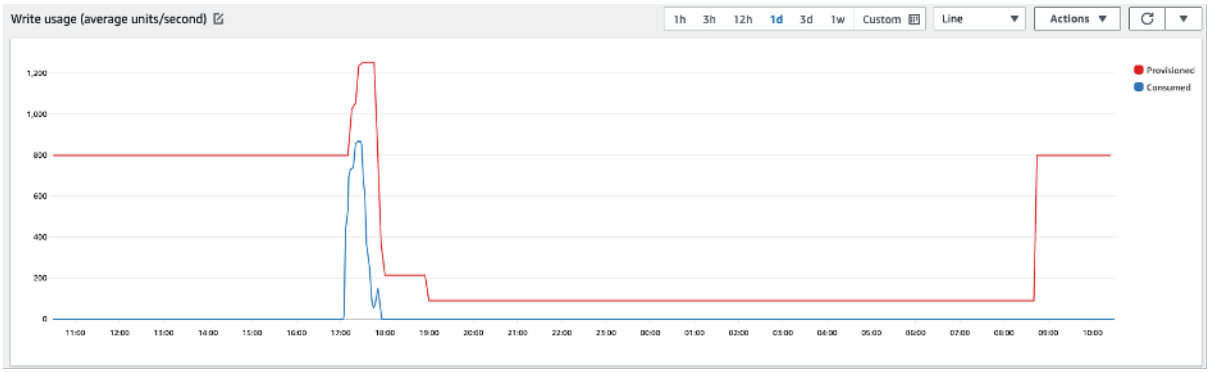
```
aws application-autoscaling describe-scheduled-actions --service-namespace
 cassandra
```

   Você deve obter um resultado parecido com este:

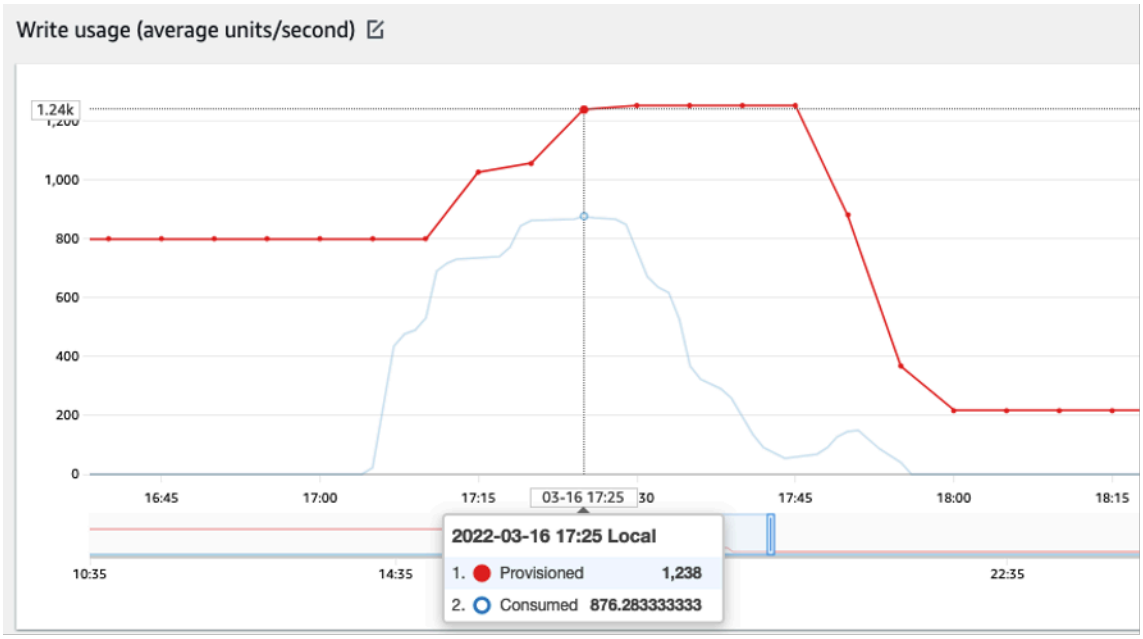
```
{
    "ScheduledActions": [
```

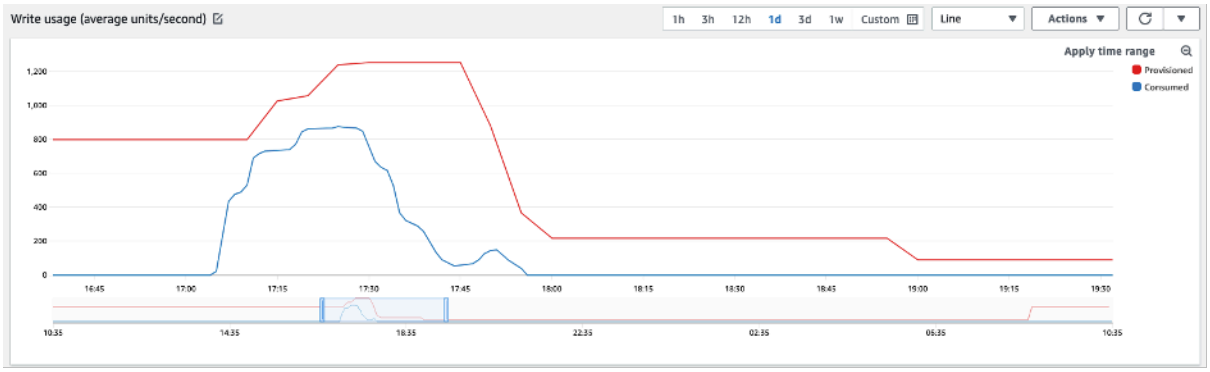
```
        {
            "ScheduledActionName": "my-5-8-scheduled-down-action",
            "ScheduledActionARN":
 "arn:aws:autoscaling:<region>:<account>:scheduledAction:<uuid>:resource/keyspaces/
table/table-name:scheduledActionName/my-5-8-scheduled-down-action",
            "ServiceNamespace": "cassandra",
            "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"
        }
    ]
}
```

A figura a seguir mostra um exemplo de carga de trabalho que sempre mantém a utilização prevista de 70%. Observe como as regras de ajuste de escala automático ainda se aplicam e o throughput não será reduzido.




Ao ampliar, podemos ver que houve um pico na aplicação que acionou o limite de Auto Scaling de 70%, forçando o Auto Scaling a entrar em ação e fornecer a capacidade extra necessária para a tabela. A ação programada de escalonamento automático afetará os valores máximo e mínimo, e é sua responsabilidade configurá-los.



### Como lidar com cargas de trabalho com pico com padrões desconhecidos

Nesse cenário, o aplicativo usa uma utilização prevista muito baixa porque você ainda não conhece os padrões do aplicativo e quer garantir que sua carga de trabalho não passe por erros de capacidade de throughput insuficiente.

Em vez disso, considere usar o modo de capacidade sob demanda. As tabelas sob demanda são perfeitas para cargas de trabalho com pico nas quais você não conhece os padrões de tráfego. Com o modo de capacidade sob demanda, você paga por solicitação pelas leituras e gravações de dados que sua aplicação executa em suas tabelas. Não é necessário especificar o throughput de leitura e gravação que você espera que seu aplicativo execute, pois o Amazon Keyspaces acomoda instantaneamente o crescimento e redução de workloads.

### Como lidar com cargas de trabalho com aplicações vinculadas

Nesse cenário, a aplicação depende de outros sistemas, como cenários de processamento em lote, nos quais você pode ter grandes picos de tráfego de acordo com os eventos na lógica da aplicação.

Considere desenvolver uma lógica personalizada de Application Auto Scaling que reaja aos eventos em que você pode aumentar a capacidade da tabela e dos `TargetValues` dependendo de suas necessidades específicas. Você pode se beneficiar Amazon EventBridge e usar uma combinação de AWS serviços como λ e Step Functions para reagir às necessidades específicas de seu aplicativo.

## Identifique seus recursos não utilizados

Esta seção apresenta uma visão geral de como avaliar recursos não utilizados regularmente. À medida que os requisitos do aplicativo evoluem, você deve garantir que os recursos sejam utilizados e evite custos desnecessários do Amazon Keyspaces. Os procedimentos descritos abaixo usam CloudWatch métricas da Amazon para identificar recursos não utilizados e tomar medidas para reduzir custos.

Você pode monitorar o Amazon Keyspaces usando CloudWatch, que coleta e processa dados brutos do Amazon Keyspaces em métricas legíveis e quase em tempo real. Essas estatísticas são retidas por um período de tempo, para que você possa acessar informações do histórico e entender melhor a utilização. Por padrão, os dados métricos do Amazon Keyspaces são enviados automaticamente para CloudWatch . Para obter mais informações, consulte O que é a Amazon CloudWatch? e retenção de métricas no Guia do CloudWatch usuário da Amazon.

Tópicos

- Como identificar recursos não utilizados
- Identificar recursos de tabela não utilizados
- Limpar recursos de tabela não utilizados
- Limpando backups de point-in-time recuperação não utilizados (PITR)


## Como identificar recursos não utilizados

Para identificar tabelas não utilizadas, você pode examinar as seguintes CloudWatch métricas durante um período de 30 dias para entender se há alguma leitura ou gravação ativa em uma tabela específica:

**`ConsumedReadCapacityUnits`**

O número de unidades de capacidade de leitura consumidas ao longo do período especificado para que você possa acompanhar quanto da capacidade consumida foi usada. Você pode recuperar a capacidade de leitura total consumida para uma tabela.

**`ConsumedWriteCapacityUnits`**

O número de unidades de capacidade de gravação consumidas ao longo do período especificado para que você possa acompanhar quanto da capacidade consumida foi usada. Você pode recuperar a capacidade de gravação total consumida para uma tabela.

## Identificar recursos de tabela não utilizados

CloudWatch A Amazon é um serviço de monitoramento e observabilidade que fornece as métricas da tabela Amazon Keyspaces que você pode usar para identificar recursos não utilizados. CloudWatch as métricas podem ser visualizadas por meio do AWS Management Console , bem como por meio do AWS Command Line Interface.

AWS Command Line Interface

Para visualizar as métricas de suas tabelas por meio do AWS Command Line Interface, você pode usar os seguintes comandos.

1. Primeiro, avalie as leituras da tabela:

> **Note**
>
> Se o nome da tabela não for exclusivo em sua conta, você também deverá especificar o nome do espaço de chaves.

```
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>
```

Para evitar a identificação errada de uma tabela como não usada, avalie as métricas por um período mais longo. Escolha um intervalo adequado de início e término, como 30 dias, e um período apropriado, como 86400.

Nos dados retornados, qualquer soma acima de 0 indica que a tabela que você está avaliando teve tráfego de leitura durante esse período.

O resultado a seguir mostra uma tabela com tráfego de leitura no período avaliado:

```
        {
            "Timestamp": "2022-08-25T19:40:00Z",
            "Sum": 36023355.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-12T19:40:00Z",
            "Sum": 38025777.5,
            "Unit": "Count"
        },
```

O resultado a seguir mostra uma tabela sem tráfego de leitura no período avaliado:

```
        {
            "Timestamp": "2022-08-01T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
```

```
            "Timestamp": "2022-08-20T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

2. Em seguida, avalie as gravações da tabela:

```
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>
```

Para evitar a identificação errada de uma tabela como não usada, avalie as métricas por um período mais longo. Escolha um intervalo adequado de início e término, como 30 dias, e um período apropriado, como 86400.

Nos dados retornados, qualquer soma acima de 0 indica que a tabela que você está avaliando teve tráfego de leitura durante esse período.

O resultado a seguir mostra uma tabela com tráfego de gravação no período avaliado:

```
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 41014457.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-18T20:15:00Z",
            "Sum": 40048531.0,
            "Unit": "Count"
        },
```

O resultado a seguir mostra uma tabela sem tráfego de gravação no período avaliado:

```
        {
            "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

As etapas a seguir permitem avaliar a utilização do seu recurso por meio do AWS Management Console.

1. Faça login AWS Management Console e navegue até a página CloudWatch de serviço em https://console.aws.amazon.com/cloudwatch/. Selecione o apropriado Região da AWS no canto superior direito do console, se necessário.
2. Na barra de navegação à esquerda, localize a seção Metrics (Métricas) e selecione All metrics (Todas as métricas).
3. A ação acima abrirá uma tela com dois painéis. No painel superior, você verá as métricas atualmente representadas graficamente. Na parte inferior, você selecionará as métricas disponíveis para representar graficamente. Escolha Amazon Keyspaces no painel inferior.
4. No painel de seleção de métricas do Amazon Keyspaces, selecione a categoria Table Metrics (Métricas de tabela) para mostrar as métricas das tabelas na região atual.
5. Identifique o nome da sua tabela rolando o menu para baixo e, em seguida, selecione as métricas das `ConsumedReadCapacityUnits` e das `ConsumedWriteCapacityUnits` para a tabela.
6. Selecione a guia Graphed metrics (2) [Métricas gráficas (2)] e ajuste a coluna Statistic (Estatística) como Sum (Soma).
7. Para evitar a identificação errada de uma tabela como não usada, avalie as métricas da tabela por um período mais longo. Na parte superior do painel gráfico, escolha um período de tempo apropriado, como 1 mês, para avaliar sua tabela. Selecione Custom (Personalizado), selecione 1 Months (Um mês) nos menus suspensos e escolha Apply (Aplicar).
8. Avalie as métricas gráficas da tabela para determinar se ela está sendo usada. Métricas acima de 0 indicam que uma tabela foi usada durante o período avaliado. Um gráfico plano em 0 para leitura e gravação indica uma tabela que não está sendo usada.

## Limpar recursos de tabela não utilizados

Se você identificou recursos de tabela não utilizados, poderá reduzir os custos contínuos das seguintes formas.

> **Note**
>
> Se você identificou uma tabela não utilizada, mas ainda gostaria de mantê-la disponível caso ela precise ser acessada no futuro, considere mudá-la para o modo sob demanda. Caso contrário, você pode considerar a exclusão da tabela.

Modos de capacidade

O Amazon Keyspaces cobra pela leitura, gravação e armazenamento de dados em tabelas do Amazon Keyspaces.

O Amazon Keyspaces oferece dois modos de capacidade, que vêm com opções de faturamento específicas do processamento de leitura e gravação nas tabelas: sob demanda e provisionada. O modo de capacidade de leitura/gravação controla como você é cobrado por throughput de leitura e gravação e como você gerencia a capacidade.

Para tabelas do modo sob demanda, não é necessário especificar o throughput de leitura e gravação que você espera que sua aplicação execute. O Amazon Keyspaces cobra você pelas leituras e gravações que sua aplicativo realiza em suas tabelas em termos de unidades de solicitação de leitura e unidades de solicitação de gravação. Se não houver atividade na tabela, você não pagará pelo throughput, mas ainda terá uma taxa de armazenamento.

Excluir tabelas

Se você descobriu uma tabela não utilizada e gostaria de excluí-la, talvez queira fazer um backup ou exportar os dados antes.

Os backups realizados AWS Backup podem aproveitar a hierarquização do armazenamento a frio, reduzindo ainda mais os custos. Consulte a documentação Gerenciar de planos de backup para informações sobre como usar um ciclo de vida para mover o backup para armazenamento frio.

Depois de fazer o backup da tabela, é possível optar por excluí-la por meio do AWS Management Console ou do AWS Command Line Interface.

## Limpando backups de point-in-time recuperação não utilizados (PITR)

O Amazon Keyspaces oferece oint-in-time recuperação P, que fornece backups contínuos por 35 dias para ajudar você a se proteger contra gravações ou exclusões acidentais. Os backups PITR têm custos associados a eles.

Consulte a documentação do Recuperação pontual para saber se as tabelas têm backups habilitados que podem não ser mais necessários.

## Avaliar seus padrões de uso de tabelas

Esta seção apresenta uma visão geral de como avaliar se você está usando suas tabelas do Amazon Keyspaces de forma eficiente. Existem certos padrões de uso que não são ideais para o Amazon Keyspaces e permitem a otimização tanto do ponto de vista do desempenho quanto do custo.

Tópicos


- Realizar menos operações de leitura altamente consistente
- Habilitar a vida útil (TTL)


### Realizar menos operações de leitura altamente consistente

O Amazon Keyspaces permite que você configure a consistência de leitura por solicitação. As solicitações de leitura são finais consistentes por padrão. Leituras finais consistentes são cobradas a 0,5 RCU para até 4 KB de dados.

A maioria das partes das cargas de trabalho distribuídas é flexível e pode tolerar uma eventual consistência. No entanto, pode haver padrões de acesso que exijam leituras altamente consistentes. Leituras altamente consistentes são cobradas em 1 RCU para até 4 KB de dados, basicamente dobrando seus custos de leitura. O Amazon Keyspaces oferece a flexibilidade de usar os dois modelos de consistência na mesma tabela.

Você pode avaliar sua carga de trabalho e o código da aplicação para confirmar se leituras altamente consistentes são usadas somente quando necessário.

### Habilitar a vida útil (TTL)

O tempo de vida (TTL) ajuda você a simplificar a lógica do aplicativo e otimizar o preço do armazenamento ao expirar automaticamente os dados das tabelas. Os dados que você não precisa mais são excluídos automaticamente da sua tabela com base no valor de Vida útil que você definiu.

# Avaliar sua capacidade provisionada para o provisionamento do tamanho certo

Esta seção apresenta uma visão geral de como avaliar o provisionamento adequado para a tabela do Amazon Keyspaces. À medida que sua carga de trabalho evolui, você deve modificar seus procedimentos operacionais adequadamente, especialmente quando sua tabela do Amazon Keyspaces está configurada no modo provisionado e você corre o risco de provisionar demais ou subprovisionar suas tabelas.

Os procedimentos descritos nessa seção exigem informações estatísticas que devem ser capturadas das tabelas do Amazon Keyspaces que oferecem suporte a seu aplicativo de produção. Para entender o comportamento do seu aplicativo, você deve definir um período de tempo significativo o suficiente para capturar a sazonalidade dos dados do aplicativo. Por exemplo, se a aplicação mostrar padrões semanais, usar um período de três semanas deve fornecer espaço suficiente para analisar as necessidades de throughput da aplicação.

Se não souber por onde começar, use pelo menos um mês de uso de dados para os cálculos abaixo.

Ao avaliar a capacidade, as tabelas do Amazon Keyspaces podem configurar unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCU) de forma independente.

Tópicos


- Como recuperar métricas de consumo em suas tabelas do Amazon Keyspaces
- Como identificar tabelas do Amazon Keyspaces subprovisionadas
- Como identificar tabelas superprovisionadas do Amazon Keyspaces


## Como recuperar métricas de consumo em suas tabelas do Amazon Keyspaces

Para avaliar a capacidade da tabela, monitore as seguintes CloudWatch métricas e selecione a dimensão apropriada para recuperar as informações da tabela:

| Unidades de capacidade de leitura | Unidades de capacidade de gravação |
| --- | --- |
| `ConsumedReadCapacityUnits` | `ConsumedWriteCapacityUnits` |
| `ProvisionedReadCapacityUnits` | `ProvisionedWriteCapacityUnits` |

| Unidades de capacidade de leitura | Unidades de capacidade de gravação |
| --- | --- |
| `ReadThrottleEvents` | `WriteThrottleEvents` |

Você pode fazer isso por meio do AWS CLI ou do AWS Management Console.

AWS CLI

Antes de recuperar as métricas de consumo da tabela, você precisa começar capturando alguns pontos de dados históricos usando a CloudWatch API.

Comece criando dois arquivos:`write-calc.json` e `read-calc.json`. Esses arquivos representam os cálculos da tabela. Você deverá atualizar alguns dos campos, conforme indicado na tabela abaixo, para corresponder ao seu ambiente.

> **Note**
>
> Se o nome da tabela não for exclusivo em sua conta, você também deverá especificar o nome do espaço de chaves.

| Nome do campo | Definição | Exemplo |
| --- | --- | --- |
| `<table-name>` | Nome da tabela que você analisará | SampleTable |
| `<period>` | O período de tempo que você usará para avaliar a utilização prevista, com base em segundos | Por um período de uma hora, você deve especificar: 3600 |
| `<start-time>` | O início do seu intervalo de avaliação, especificado no formato ISO8601 | 2022-02-21T23:00:00 |
| `<end-time>` | O final do intervalo de avaliação, especificado no formato ISO8601 | 2022-02-22T06:00:00 |

O arquivo de cálculos de gravação recupera o número de WCU provisionado e consumido no período de tempo para o intervalo de datas especificado. Também gera uma porcentagem de utilização que é usada para análise. O conteúdo completo do arquivo `write-calc.json` deve ser como o exemplo a seguir.

```
{
  "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
}
```

O arquivo de cálculos de leitura usa uma métrica semelhante. Esse arquivo recupera quantas RCUs foram provisionadas e consumidas durante o período para o intervalo de datas especificado. O conteúdo do arquivo `read-calc.json` deve ser semelhante a este exemplo.

```
{
  "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
}
```

Depois de criar os arquivos, você poderá começar a recuperar os dados de utilização.

1. Para recuperar dados de utilização de gravação, emita o seguinte comando:

```
aws cloudwatch get-metric-data --cli-input-json file://write-calc.json
```

2. Para recuperar dados de utilização de leitura, emita o seguinte comando:

```
aws cloudwatch get-metric-data --cli-input-json file://read-calc.json
```

O resultado de ambas as consultas é uma série de pontos de dados no formato JSON usados para análise. Seu resultado depende do número de pontos de dados que você especificou, do período e de seus próprios dados específicos da carga de trabalho. Ela se parece com o exemplo a seguir.

```
{
    "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**
>
> Se você especificar um período curto e um longo intervalo de tempo, talvez seja necessário modificar o valor `MaxDatapoints` que, por padrão, está definido como 24 no script. Isso representa um ponto de dados por hora e 24 por dia.

AWS Management Console

1. Faça login no AWS Management Console e navegue até a página de CloudWatch serviço em Introdução ao AWS Management Console. Selecione o apropriado, Região da AWS se necessário.
2. Localize a seção Metrics (Métricas) na barra de navegação à esquerda e selecione All metrics (Todas as métricas).
3. Isso abrirá uma tela com dois painéis. O painel superior mostra o gráfico e o painel inferior tem as métricas que você deseja representar graficamente. Escolha o painel Amazon Keyspaces.
4. Selecione a categoria Table Metrics (Métricas da tabela) nos subpainéis. Isso mostra as tabelas em seu atual Região da AWS.
5. Identifique o nome da sua tabela rolando o menu para baixo e, em seguida, selecione as métricas da operação de gravação: `ConsumedWriteCapacityUnits` e `ProvisionedWriteCapacityUnits`.

   > **Note**
   >
   > Este exemplo fala sobre métricas da operação de gravação, mas você também pode usar essas etapas para representar graficamente as métricas da operação de leitura.

6. Selecione a guia Graphed metrics (2) (Métricas em gráfico (2)) para modificar as fórmulas. Por padrão, CloudWatch escolhe a função estatística Média para os gráficos.
7. Com as duas métricas em gráfico selecionadas (a caixa de seleção à esquerda), selecione o menu Add math (Adicionar matemática), seguido por Common (Comum), e selecione a função Percentage (Porcentagem). Repita o procedimento duas vezes.

   Primeira vez selecionando a função Percentage (Porcentagem):

   Pela segunda vez, selecionando a função Percentage (Porcentagem):

8. Nesse ponto, você deve ter quatro métricas no menu inferior. Vamos trabalhar no cálculo de `ConsumedWriteCapacityUnits`. Para ser consistente, você precisa combinar os nomes com os usados na AWS CLI seção. Clique no ID m1 e altere esse valor para consumedWCU.
9. Altere a estatística de Average (Média) para Sum (Soma). Essa ação cria automaticamente outra métrica chamada ANOMALY_DETECTION_BAND. Para saber o escopo desse procedimento, vamos ignorá-lo removendo a caixa de seleção na métrica ad1 recém-gerada.
10. Repita a etapa 8 para renomear o ID m2 como provisionedWCU. Deixe a estatística definida como Average (Média).
11. Selecione o rótulo Expression1 e atualize o valor para m1 e o rótulo para Consumed WCUs (WCUs consumidas).

> **Note**
>
> Certifique-se de ter selecionado somente m1 (caixa de seleção à esquerda) e provisionedWCU para visualizar corretamente os dados. Atualize a fórmula clicando em Details (Detalhes) e alterando a fórmula para consumedWCU/PERIOD(consumedWCU). Essa etapa também pode gerar outra métrica ANOMALY_DETECTION_BAND, mas, para o escopo desse procedimento, podemos ignorá-la.

12. Agora você deve ter dois gráficos: um que indica suas WCUs provisionadas na tabela e outro que indica as WCUs consumidas.
13. Atualize a fórmula de porcentagem selecionando o gráfico Expression2 (e2). Renomeie os rótulos e IDs para utilizationPercentage. Renomeie a fórmula para corresponder a 100*(m1/provisionedWCU).
14. Remova a caixa de seleção de todas as métricas, exceto utilizationPercentage, para visualizar seus padrões de utilização. O intervalo padrão é definido como 1 minuto, mas fique à vontade para modificá-lo conforme necessário.

Os resultados obtidos dependem dos dados reais de sua carga de trabalho. Intervalos com mais de 100% de utilização estão sujeitos a eventos de erros de capacidade de throughput insuficiente. O Amazon Keyspaces oferece capacidade de expansão, mas assim que a capacidade de throughput se esgota, qualquer coisa acima de 100% experimenta eventos de erro de baixa capacidade de transferência.

## Como identificar tabelas do Amazon Keyspaces subprovisionadas

Para a maioria das workloads, uma tabela é considerada subprovisionada quando consome constantemente mais de 80% de sua capacidade provisionada.

A capacidade de expansão é um atributo do Amazon Keyspaces que permite que os clientes consumam temporariamente mais RCUS/WCUs do que o provisionado originalmente (mais do que o throughput provisionado por segundo definido na tabela). A capacidade de expansão foi criada para absorver aumentos repentinos no tráfego devido a eventos especiais ou picos de uso. Essa capacidade de expansão é limitada; para obter mais informações, consultethe section called "Capacidade de expansão". Assim que as RCUs e WCUs não utilizadas forem esgotadas, você pode experienciar eventos de erros de capacidade de throughput insuficiente se tentar consumir mais capacidade do que a provisionada. Quando o tráfego do seu aplicativo está se aproximando da taxa de utilização de 80%, o risco de passar por erros desnecessários de throughput de baixa capacidade é significativamente maior.

A regra da taxa de utilização de 80% varia com a sazonalidade de seus dados e com o crescimento do tráfego. Considere os seguintes cenários:

- Se o seu tráfego se manteve estável com uma taxa de utilização de ~90% nos últimos 12 meses, sua tabela tem a capacidade certa
- Se o tráfego da sua aplicação estiver crescendo a uma taxa de 8% ao mês em menos de 3 meses, você chegará a 100%
- Se o tráfego da sua aplicação estiver crescendo a uma taxa de 5% em pouco mais de 4 meses, você ainda chegará a 100%

Os resultados das consultas acima fornecem uma imagem da sua taxa de utilização. Use-os como um guia para avaliar melhor outras métricas que podem ajudar você a escolher aumentar a capacidade da tabela conforme necessário (por exemplo: uma taxa de crescimento mensal ou semanal). Trabalhe com sua equipe de operações para definir qual é uma boa porcentagem para sua carga de trabalho e suas tabelas.

Há cenários especiais em que os dados são distorcidos quando os analisamos diariamente ou semanalmente. Por exemplo, com aplicativos sazonais que têm picos de uso durante o horário comercial (mas depois caem para quase zero fora do horário comercial), você pode se beneficiar do Application Auto Scaling programado, em que especifica as horas do dia (e os dias da semana) para aumentar a capacidade provisionada e quando reduzi-la. Em vez de buscar maior capacidade

para cobrir as horas de pico, você também pode se beneficiar das configurações de ajuste de escala automático de tabelas do Amazon Keyspaces se a sua sazonalidade for menos acentuada.

## Como identificar tabelas superprovisionadas do Amazon Keyspaces

Os resultados da consulta obtidos dos scripts acima fornecem os pontos de dados necessários para realizar algumas análises iniciais. Se o seu conjunto de dados apresentar valores inferiores a 20% de utilização em vários intervalos, sua tabela pode estar superprovisionada. Para definir melhor se você precisa reduzir o número de WCUs e RCUS, revise as outras leituras nos intervalos.

Quando suas tabelas contêm vários intervalos de uso baixos, você pode se beneficiar do uso de políticas de Application Auto Scaling, seja programando o Application Auto Scaling ou simplesmente configurando as políticas de Application Auto Scaling padrão para a tabela, com base na utilização.

Se você tem uma carga de trabalho com baixa utilização e alta taxa de aceleração (Máximo (ThrottleEvents) /Min () no intervaloThrottleEvents), isso pode acontecer quando você tem uma carga de trabalho muito alta, em que o tráfego aumenta significativamente em dias específicos (ou horários do dia), mas é consistentemente baixo. Nesses cenários, pode ser benéfico usar o Application Auto Scaling programado.

# Uso do NoSQL Workbench com o Amazon Keyspaces (para Apache Cassandra)

O NoSQL Workbench é um aplicativo do lado do cliente que ajuda você a projetar e visualizar modelos de dados não relacionais para o Amazon Keyspaces com mais facilidade. Os clientes do NoSQL Workbench estão disponíveis para Windows, macOS e Linux.

Como projetar modelos de dados e criar recursos automaticamente

O NoSQL Workbench fornece uma interface de apontar e clicar para projetar e criar modelos de dados do Amazon Keyspaces. Você pode criar facilmente novos modelos de dados do zero definindo espaços de chaves, tabelas e colunas. Você também pode importar modelos de dados existentes e fazer modificações (como adicionar, editar ou remover colunas) para adaptar os modelos de dados a novos aplicativos. Em seguida, o NoSQL Workbench permite que você confirme os modelos de dados no Amazon Keyspaces ou no Apache Cassandra e crie os espaços de chaves e as tabelas automaticamente. Para saber como criar modelos de dados, consulte the section called “Modelador de dados”.

Como visualizar modelos de dados

Usando o NoSQL Workbench, você pode visualizar seus modelos de dados para ajudar a garantir que os modelos de dados possam suportar as consultas e os padrões de acesso do seu aplicativo. Você também pode salvar e exportar seus modelos de dados em vários formatos para colaboração, documentação e apresentações. Para obter mais informações, consulte the section called “Visualizador de dados”.

Tópicos


- Fazer download do NoSQL Workbench
- Noções básicas do NoSQL Workbench
- Como criar modelos de dados
- Como visualizar modelos de dados
- Como confirmar modelos de dados com o Amazon Keyspaces e o Apache Cassandra
- Modelos de dados de amostra no NoSQL Workbench
- Histórico de versões do NoSQL Workbench

# Fazer download do NoSQL Workbench

Siga estas instruções para fazer download do NoSQL Workbench e instalá-lo.

Para fazer download do NoSQL Workbench e instalá-lo

1. Use um dos links a seguir para fazer download do NoSQL Workbench gratuitamente.

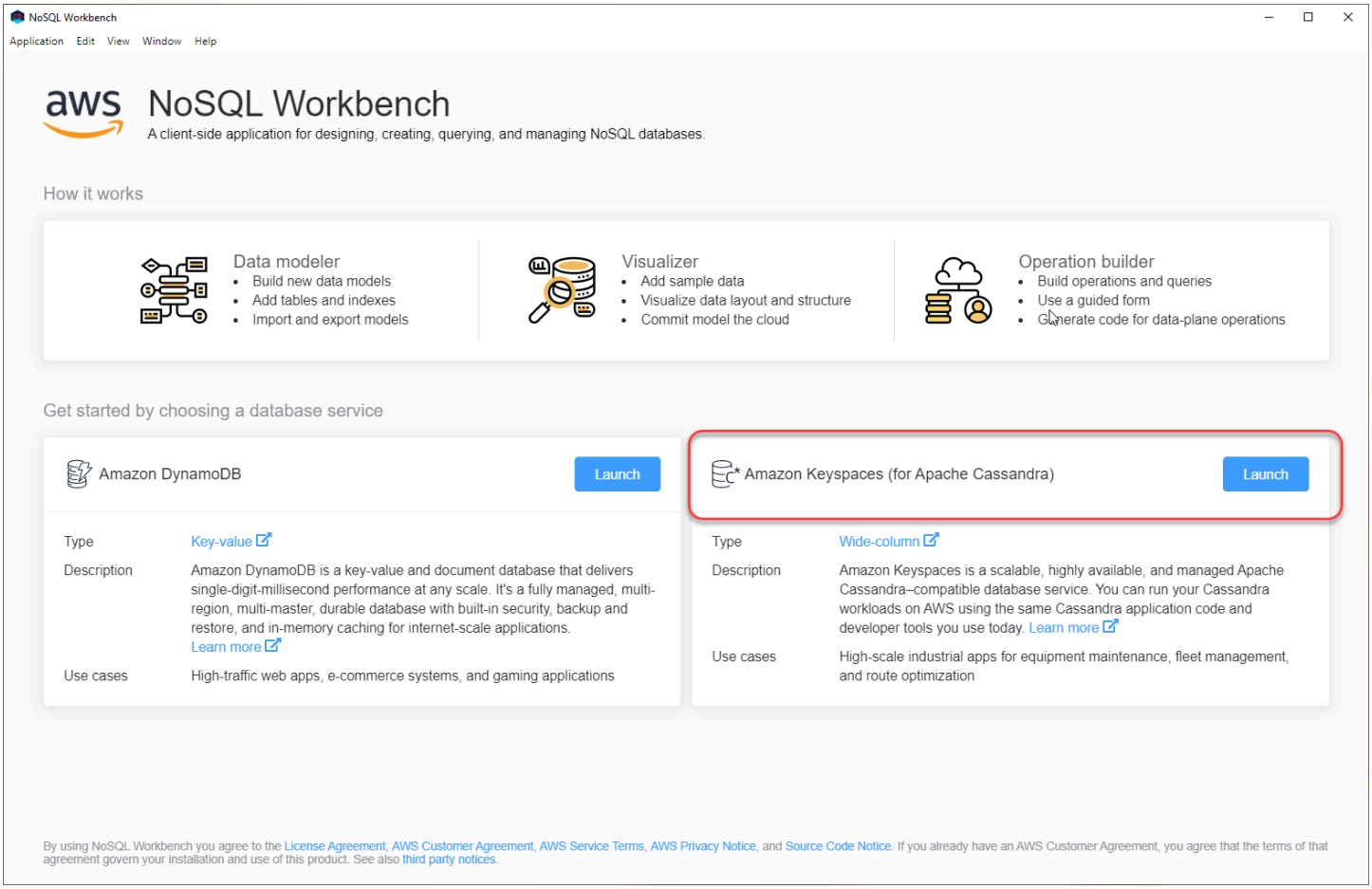
| Sistema operacional | Link para download |
| --- | --- |
| macOS | Download para macOS |
| Linux* | Download para Linux |
| Windows | Download para Windows |

   * O NoSQL Workbench oferece suporte a Ubuntu 12.04, Fedora 21 e Debian 8 ou quaisquer versões mais recentes dessas distribuições do Linux.

2. Depois que o download for concluído, inicie o aplicativo e siga as instruções na tela para concluir a instalação.

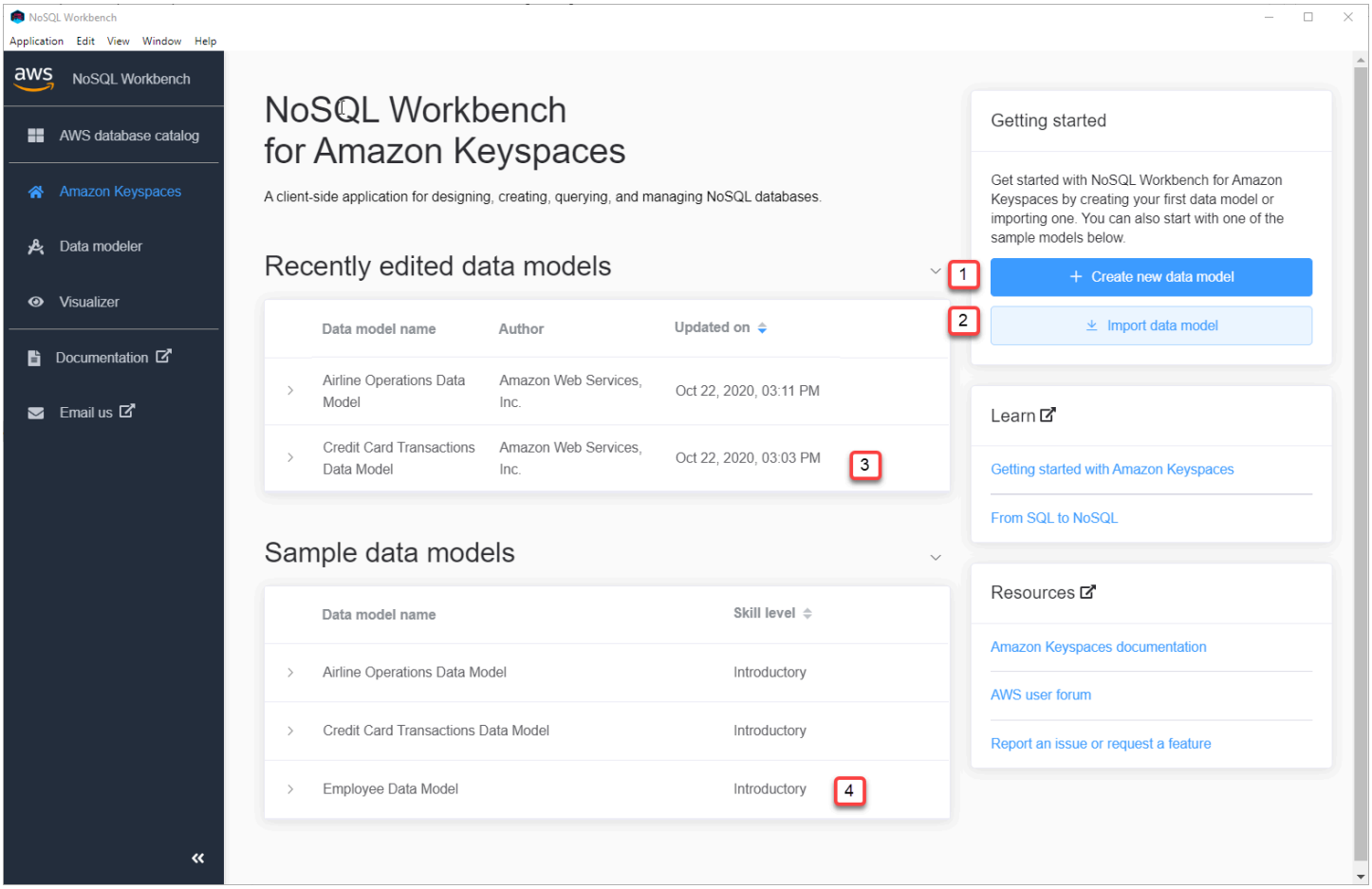
# Noções básicas do NoSQL Workbench

Para começar a usar o NoSQL Workbench, na página de banco de dados do catálogo no NoSQL Workbench, selecione Amazon Keyspaces e, em seguida, selecione Iniciar.




Isso abre a página inicial do NoSQL Workbench para o Amazon Keyspaces, onde você tem as seguintes opções para começar:

1. Criar um novo modelo de dados.
2. Importar um modelo de dados existente no formato JSON.
3. Abrir um modelo de dados editado recentemente.
4. Abrir um dos modelos de amostra disponíveis.




Cada uma das opções abre o modelador de dados NoSQL Workbench. Para continuar criando um novo modelo de dados, consulte the section called “Como criar um modelo de dados”. Para editar um modelo de dados existente, consulte the section called “Como editar um modelo de dados”.

# Como criar modelos de dados

Você pode usar o modelador de dados NoSQL Workbench para projetar novos modelos de dados com base nos padrões de acesso a dados do seu aplicativo. Você pode usar o modelador de dados para criar novos modelos de dados ou importar e modificar modelos de dados existentes criados usando o NoSQL Workbench. O modelador de dados também inclui alguns modelos de dados de amostra para ajudá-lo a começar a modelar dados.

Tópicos

- Como criar modelos de dados com o NoSQL Workbench
- Como editar modelos de dados existentes com o NoSQL Workbench

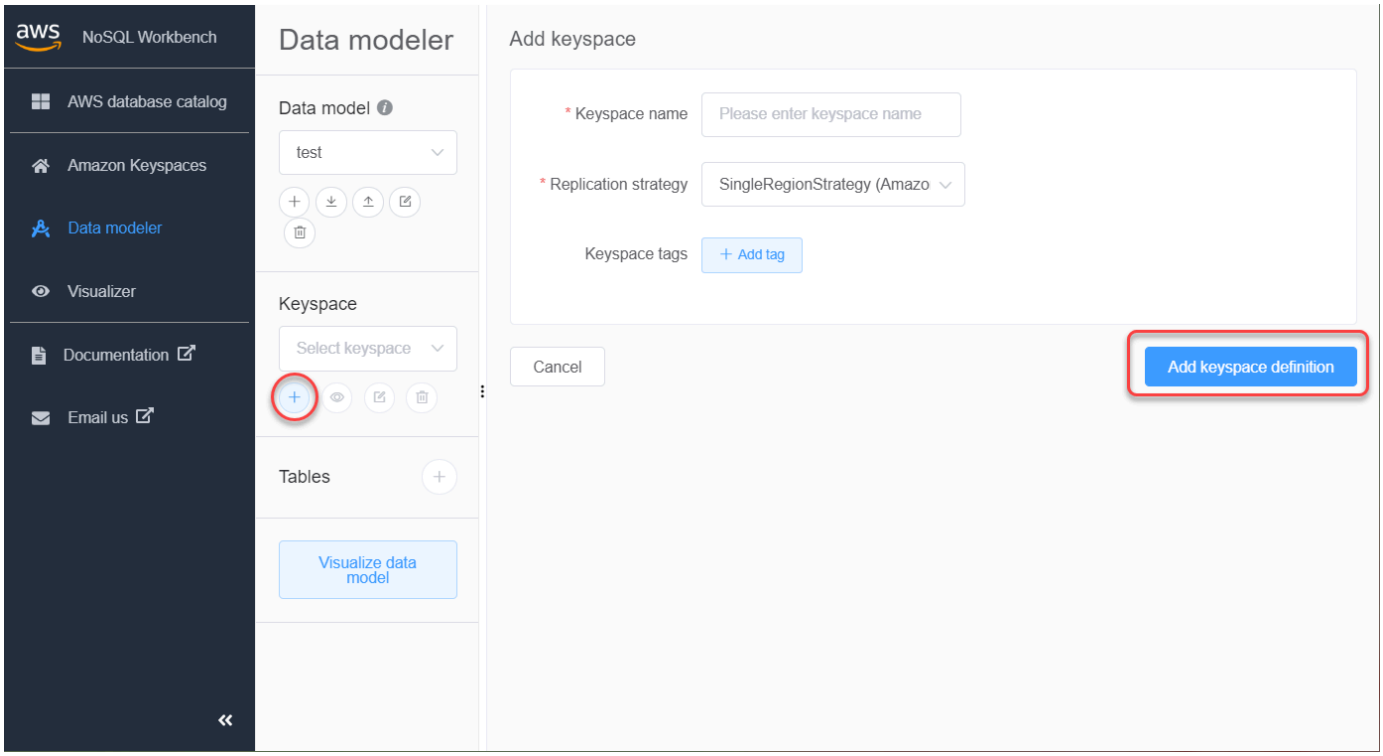
# Como criar modelos de dados com o NoSQL Workbench

Para criar um novo modelo de dados para o Amazon Keyspaces, você pode usar o modelador de dados NoSQL Workbench para criar espaços de chaves, tabelas e colunas. Siga estas etapas para criar um novo modelo de dados.

1. Para criar um novo espaço de chaves, selecione o sinal de mais em Espaço de chaves.

   Nesta etapa, escolha as propriedades e configurações a seguir.

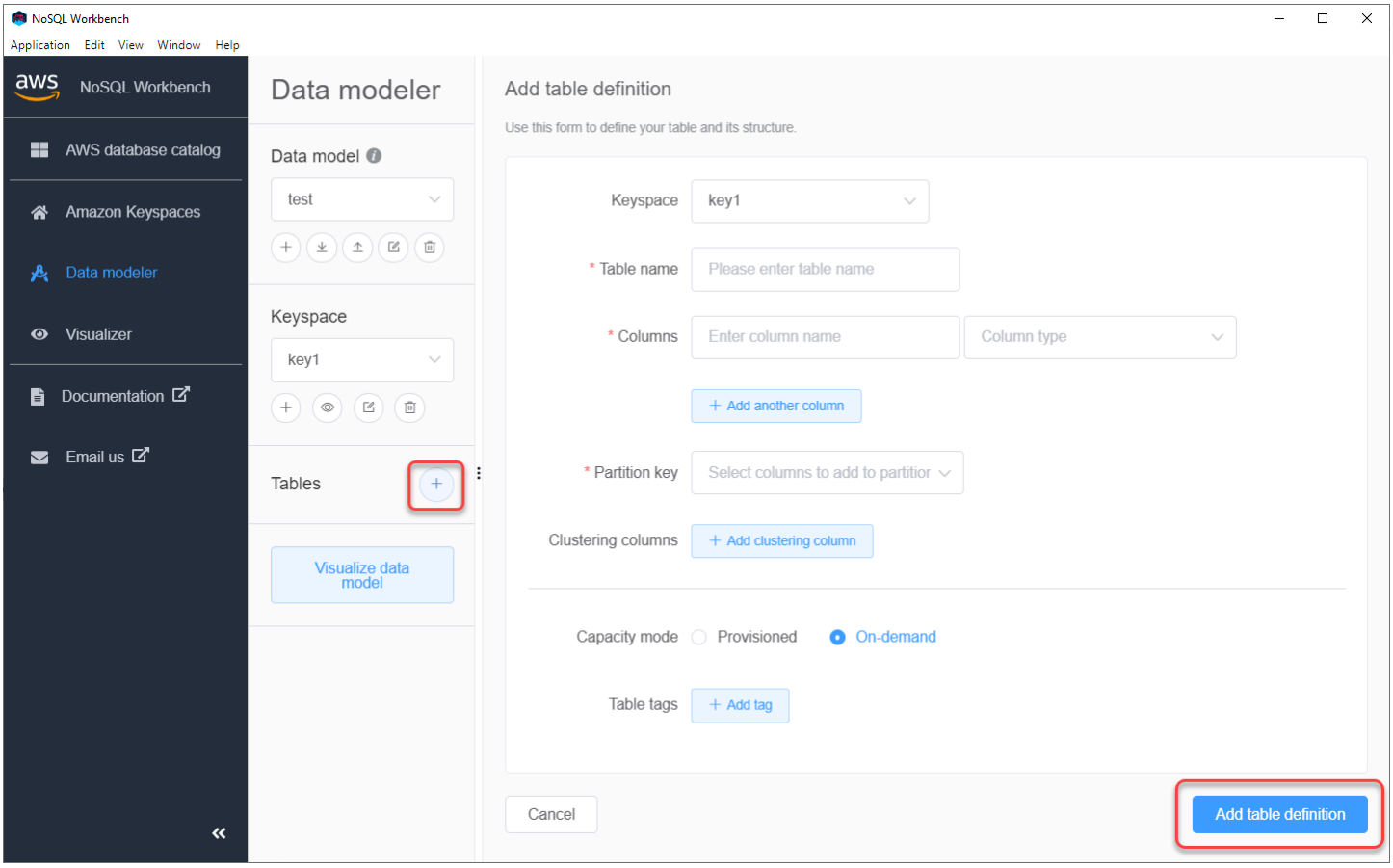
   - Nome do espaço de chaves – Insira o nome do novo espaço de chaves.
   - Estratégia de replicação – Escolha a estratégia de replicação para o espaço de chaves. O Amazon Keyspaces usa a SingleRegionStrategy para replicar dados três vezes automaticamente em várias Zonas de disponibilidade da AWS. Se você planeja confirmar o modelo de dados em um cluster Apache Cassandra, pode escolher SimpleStrategy ou NetworkTopologyStrategy.
   - Tags de espaços de chaves As tags de recursos são opcionais e permitem categorizar recursos de diferentes maneiras, por exemplo, por finalidade, proprietário, ambiente ou outros critérios. Para saber mais sobre tags para recursos do Amazon Keyspaces, consulte. the section called “Trabalhando com tags”
2. Selecione Adicionar definição de espaço de chaves para criar o espaço de chaves.



3. Para criar uma nova tabela, selecione o sinal de adição ao lado de Tabelas. Nesta etapa, você define as propriedades e configurações a seguir.

   - Nome da tabela – O nome da nova tabela.
   - Colunas — Adicione um nome de coluna e escolha o tipo de dados. Repita essas etapas para cada coluna em seu esquema.
   - Chave de partição – Selecione colunas para a chave de partição.
   - Colunas de clustering – Selecione colunas de clustering (opcional).
   - Modo de capacidade – Selecione o modo de capacidade leitura/gravação para a tabela. Você pode escolher capacidade provisionada ou sob demanda. Para saber mais sobre modos de capacidade, consulte the section called “Modos de capacidade de leitura/gravação”.
   - Tags de tabelas – As tags de recursos são opcionais e permitem categorizar recursos de diferentes maneiras, por exemplo, por finalidade, proprietário, ambiente ou outros critérios. Para saber mais sobre tags para recursos do Amazon Keyspaces, consulte. the section called “Trabalhando com tags”

4. Selecione Adicionar definição de tabela para criar a nova tabela.
5. Repita essas etapas para criar tabelas adicionais.
6. Continue para the section called “Visualização de um modelo de dados” para visualizar o modelo de dados que você criou.




## Como editar modelos de dados existentes com o NoSQL Workbench

Com o modelador de dados NoSQL Workbench, você pode editar modelos de dados existentes no Amazon Keyspaces. Eles podem ser modelos de dados importados de um arquivo, os modelos de dados de amostra fornecidos ou os modelos de dados que você criou anteriormente.

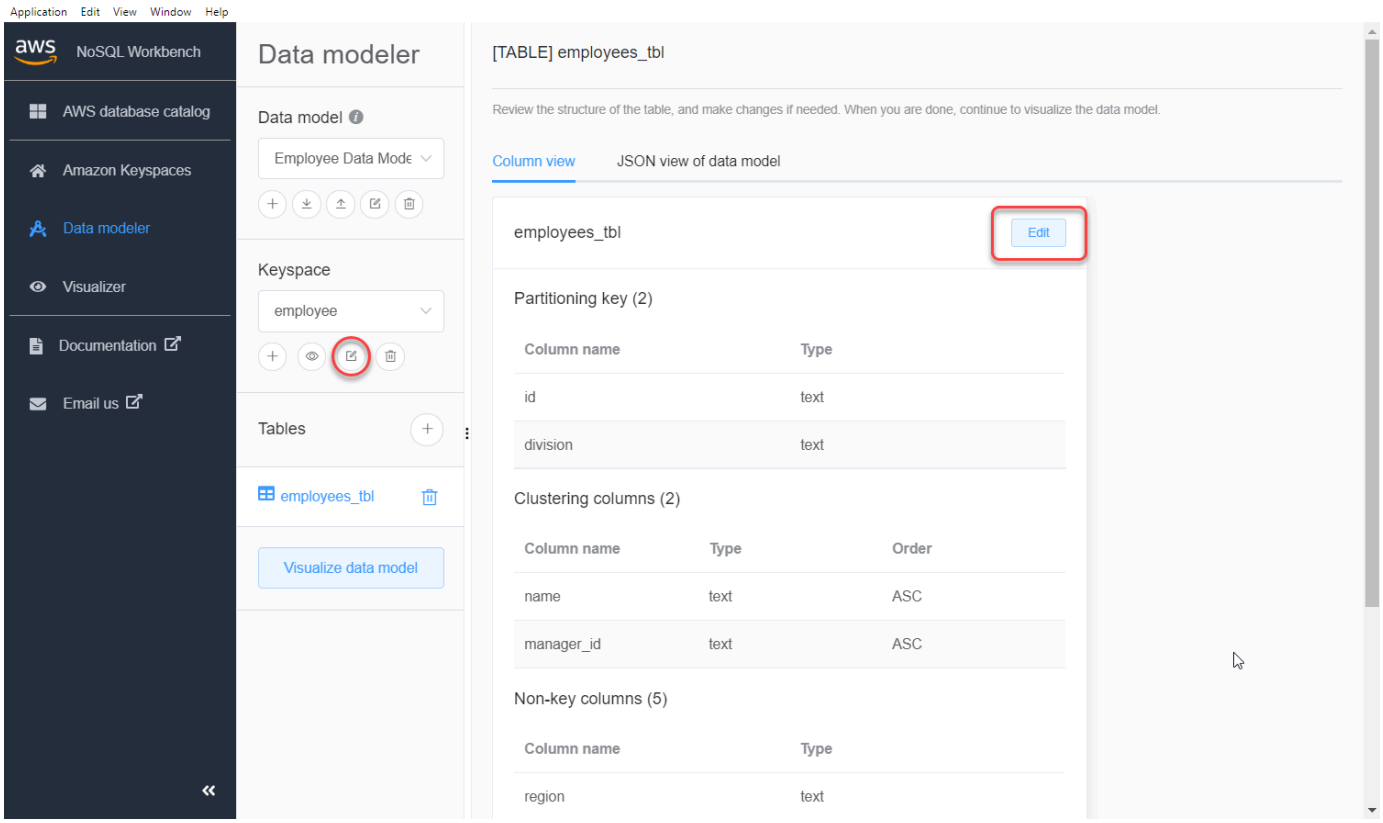
1. Para editar um espaço de chaves, selecione o símbolo de edição em Espaço de chaves.

   Nesta etapa, você pode editar as propriedades e configurações a seguir.

   - Nome do espaço de chaves – Insira o nome do novo espaço de chaves.
   - Estratégia de replicação – Escolha a estratégia de replicação para o espaço de chaves. O Amazon Keyspaces usa a SingleRegionStrategy para replicar dados três vezes automaticamente em várias Zonas de disponibilidade da AWS. Se você planeja confirmar o modelo de dados em um cluster Apache Cassandra, pode escolher SimpleStrategy ou NetworkTopologyStrategy.
   - Tags de espaços de chaves As tags de recursos são opcionais e permitem categorizar recursos de diferentes maneiras, por exemplo, por finalidade, proprietário, ambiente ou outros

critérios. Para saber mais sobre tags para recursos do Amazon Keyspaces, consulte. the section called “Trabalhando com tags”

2. Selecione Salvar edições para atualizar o espaço de chaves.




3. Para editar uma tabela, selecione Editar ao lado do nome da tabela. Nesta etapa, você pode atualizar as propriedades e configurações a seguir.

   - Nome da tabela – O nome da nova tabela.
   - Colunas — Adicione um nome de coluna e escolha o tipo de dados. Repita essas etapas para cada coluna em seu esquema.
   - Chave de partição – Selecione colunas para a chave de partição.
   - Colunas de clustering – Selecione colunas de clustering (opcional).
   - Modo de capacidade – Selecione o modo de capacidade leitura/gravação para a tabela. Você pode escolher capacidade provisionada ou sob demanda. Para saber mais sobre modos de capacidade, consulte the section called “Modos de capacidade de leitura/gravação”.
   - Tags de tabelas – As tags de recursos são opcionais e permitem categorizar recursos de diferentes maneiras, por exemplo, por finalidade, proprietário, ambiente ou outros critérios. Para saber mais sobre tags para recursos do Amazon Keyspaces, consulte. the section called “Trabalhando com tags”

4. Selecione Salvar edições para atualizar a tabela.
5. Continue para the section called “Visualização de um modelo de dados” para visualizar o modelo de dados que você atualizou.

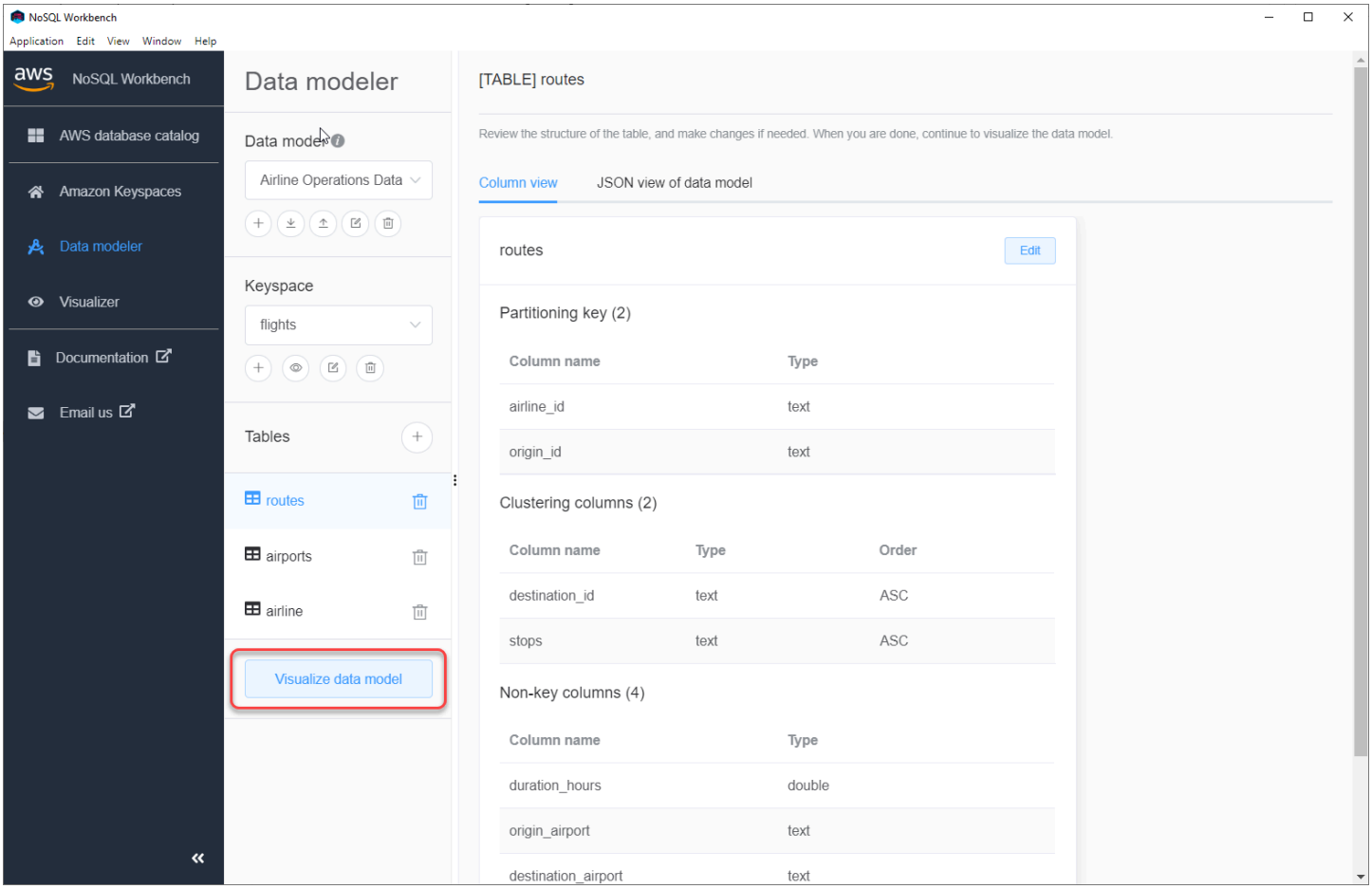
# Como visualizar modelos de dados

Usando o NoSQL Workbench, você pode visualizar seus modelos de dados para ajudar a garantir que os modelos de dados possam suportar as consultas e os padrões de acesso do seu aplicativo. Você também pode salvar e exportar seus modelos de dados em vários formatos para colaboração, documentação e apresentações.

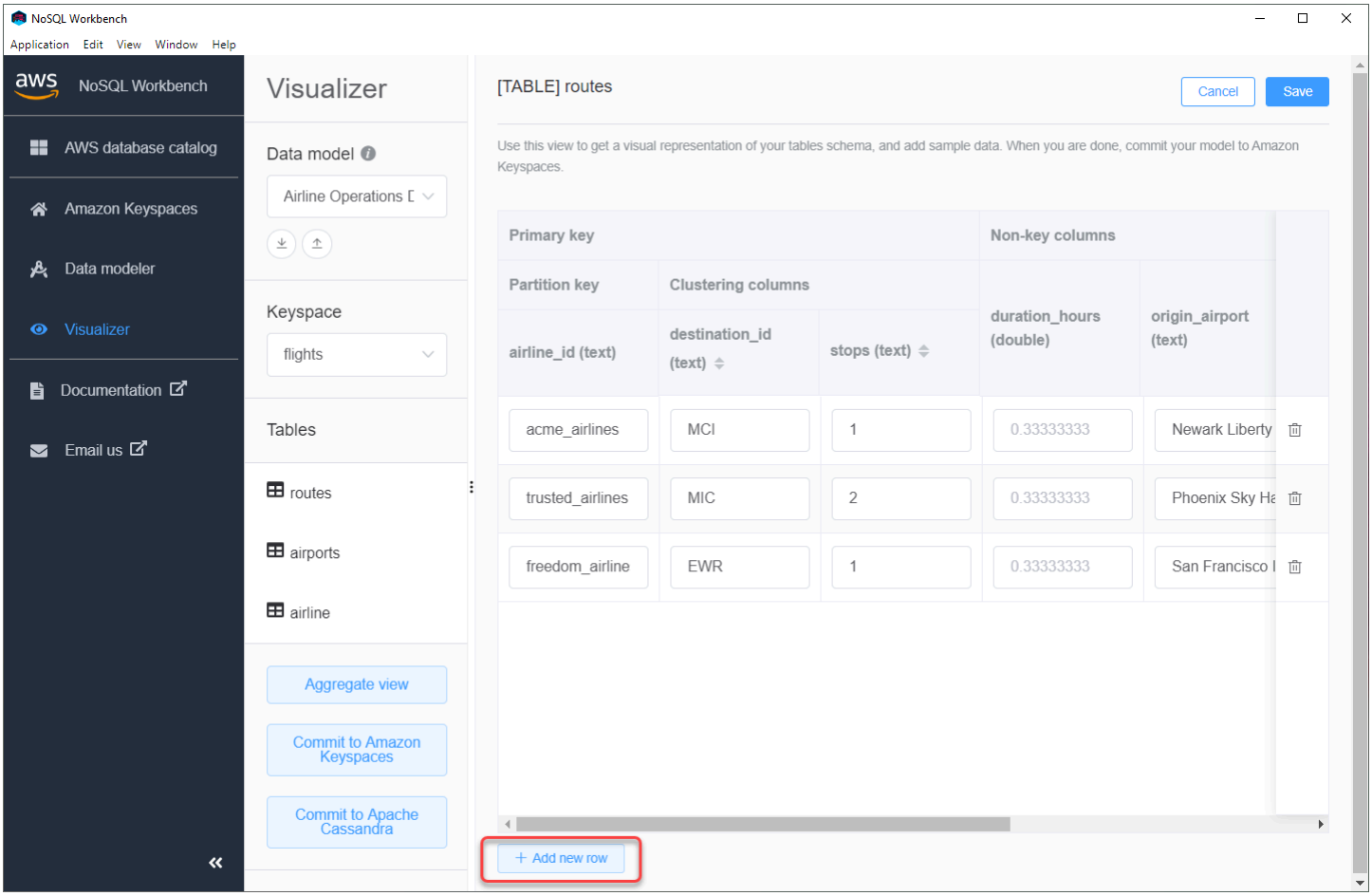
Depois de criar um novo modelo de dados ou editar um modelo de dados existente, você pode visualizar o modelo.

## Visualização de modelos de dados com o NoSQL Workbench

Quando você tiver concluído o modelo de dados no modelador de dados, selecione Visualizar modelo de dados.

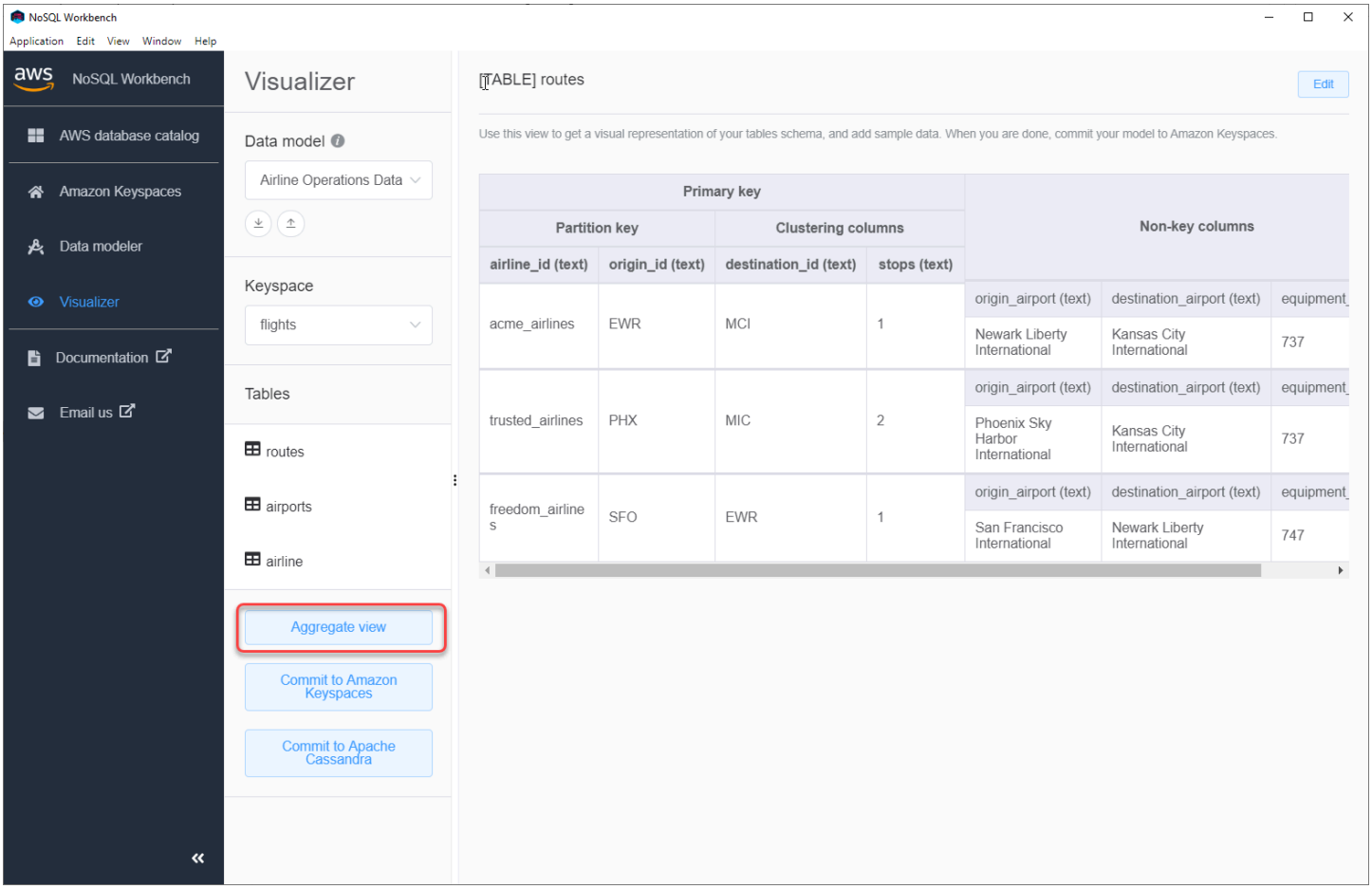



Isso o leva ao visualizador de dados no NoSQL Workbench. O visualizador de dados fornece uma representação visual do esquema da tabela e permite adicionar dados de amostra. Para adicionar dados de amostra a uma tabela, escolha uma tabela do modelo e selecione Editar. Para adicionar uma nova linha de dados, selecione Adicionar nova linha na parte inferior da tela. Selecione Save (Salvar) ao concluir.



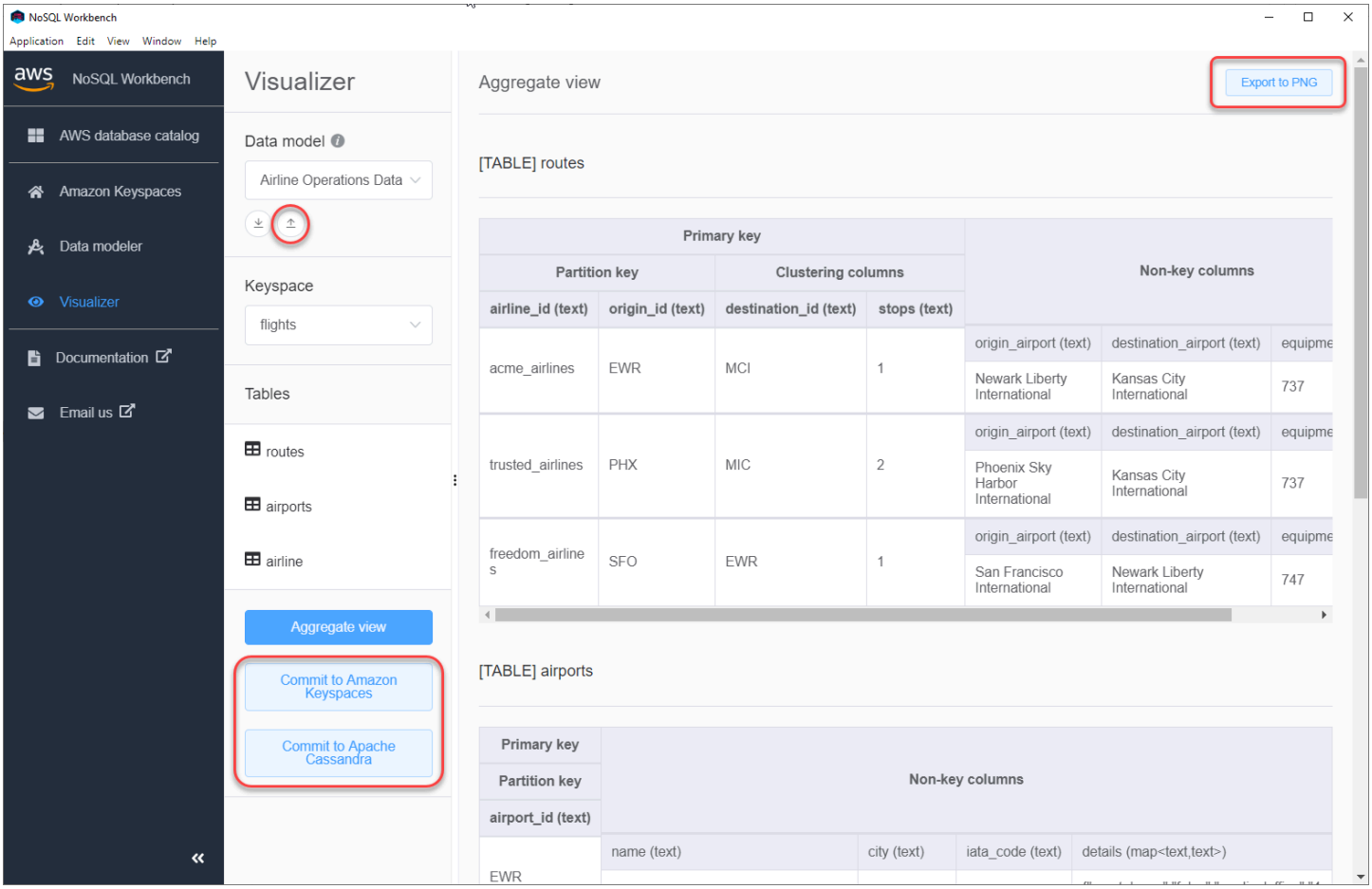

# Exibição agregada

Depois de confirmar o esquema da tabela, você pode agregar exibições do modelo de dados.




Depois de agregar a exibição do modelo de dados, você pode exportá-la para um arquivo PNG. Para exportar o modelo de dados para um arquivo JSON, selecione o sinal de upload abaixo do nome do modelo de dados.

> **Note**
>
> Você pode exportar o modelo de dados no formato JSON a qualquer momento no processo de design.



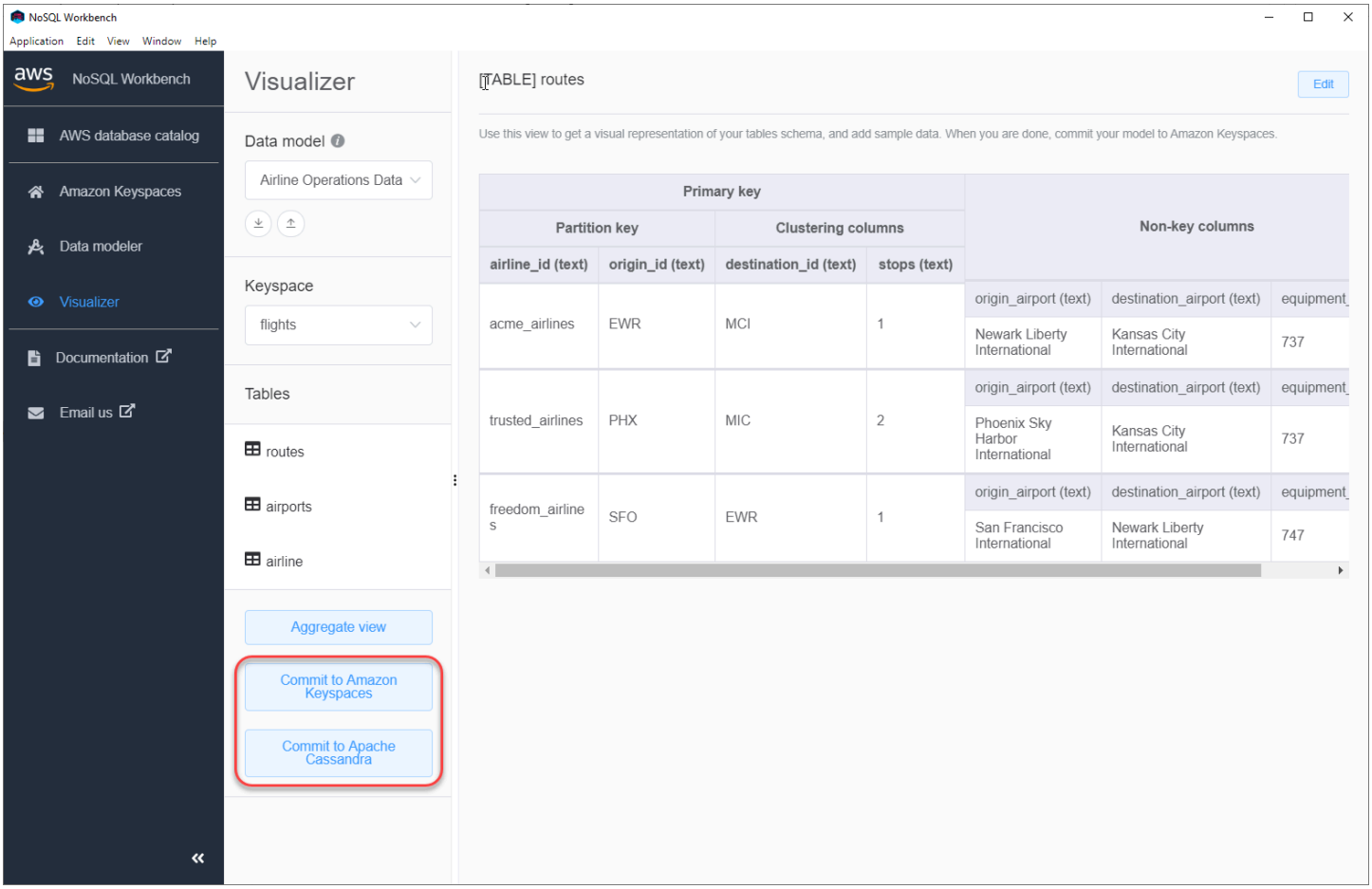

Você tem as seguintes opções para confirmar as alterações:

- Confirmar no Amazon Keyspaces
- Confirmar para um cluster Apache Cassandra

Para saber mais sobre como confirmar alterações, consulte the section called “Confirmar um modelo de dados”.

# Como confirmar modelos de dados com o Amazon Keyspaces e o Apache Cassandra

Esta seção mostra como confirmar modelos de dados concluídos nos clusters do Amazon Keyspaces e do Apache Cassandra. Esse processo cria automaticamente os recursos do lado do servidor para espaços de chaves e tabelas com base nas configurações que você definiu no modelo de dados.




Tópicos

- Antes de começar
- Como conectar-se ao Amazon Keyspaces com credenciais específicas do serviço
- Como conectar-se ao Amazon Keyspaces com credenciais (IAM) AWS Identity and Access Management
- Como usar uma conexão salva
- Como confirmar o Apache Cassandra

## Antes de começar

O Amazon Keyspaces requer o uso do Transport Layer Security (TLS) para ajudar a proteger as conexões com os clientes. Para se conectar ao Amazon Keyspaces usando TLS, você precisa concluir a seguinte tarefa antes de começar.

- Faça o download do certificado digital Starfield usando o comando a seguir e salve `sf-class2-root.crt` localmente ou em seu diretório inicial.

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

> **Note**
>
> Você também pode usar o certificado digital da Amazon para se conectar ao Amazon Keyspaces e continuar fazendo isso se seu cliente estiver se conectando ao Amazon Keyspaces com sucesso. O certificado Starfield fornece compatibilidade adicional com versões anteriores para clientes que usam autoridades de certificação mais antigas.


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


Depois de salvar o arquivo do certificado, você pode se conectar ao Amazon Keyspaces. Uma opção é se conectar usando credenciais específicas do serviço. As credenciais específicas do serviço são um nome de usuário e uma senha associados a um usuário do IAM específico e só podem ser usados com o serviço especificado. A segunda opção é conectar-se às credenciais do IAM que estão usando o processo Signature Version 4 (SigV4) da AWS. Para saber mais sobre essas duas opções, consulte the section called “Criação de credenciais”.

Para conectar-se com credenciais específicas do serviço, consulte the section called “Como conectar-se com credenciais específicas do serviço”.

Para conectar-se com credenciais do IAM, consulte the section called “Como conectar-se com credenciais do IAM”.

## Como conectar-se ao Amazon Keyspaces com credenciais específicas do serviço

Esta seção mostra como usar credenciais específicas do serviço para confirmar o modelo de dados que você criou ou editou com o NoSQL Workbench.

1. Para criar uma nova conexão usando credenciais específicas do serviço, escolha a guia Conectar usando nome de usuário e senha.

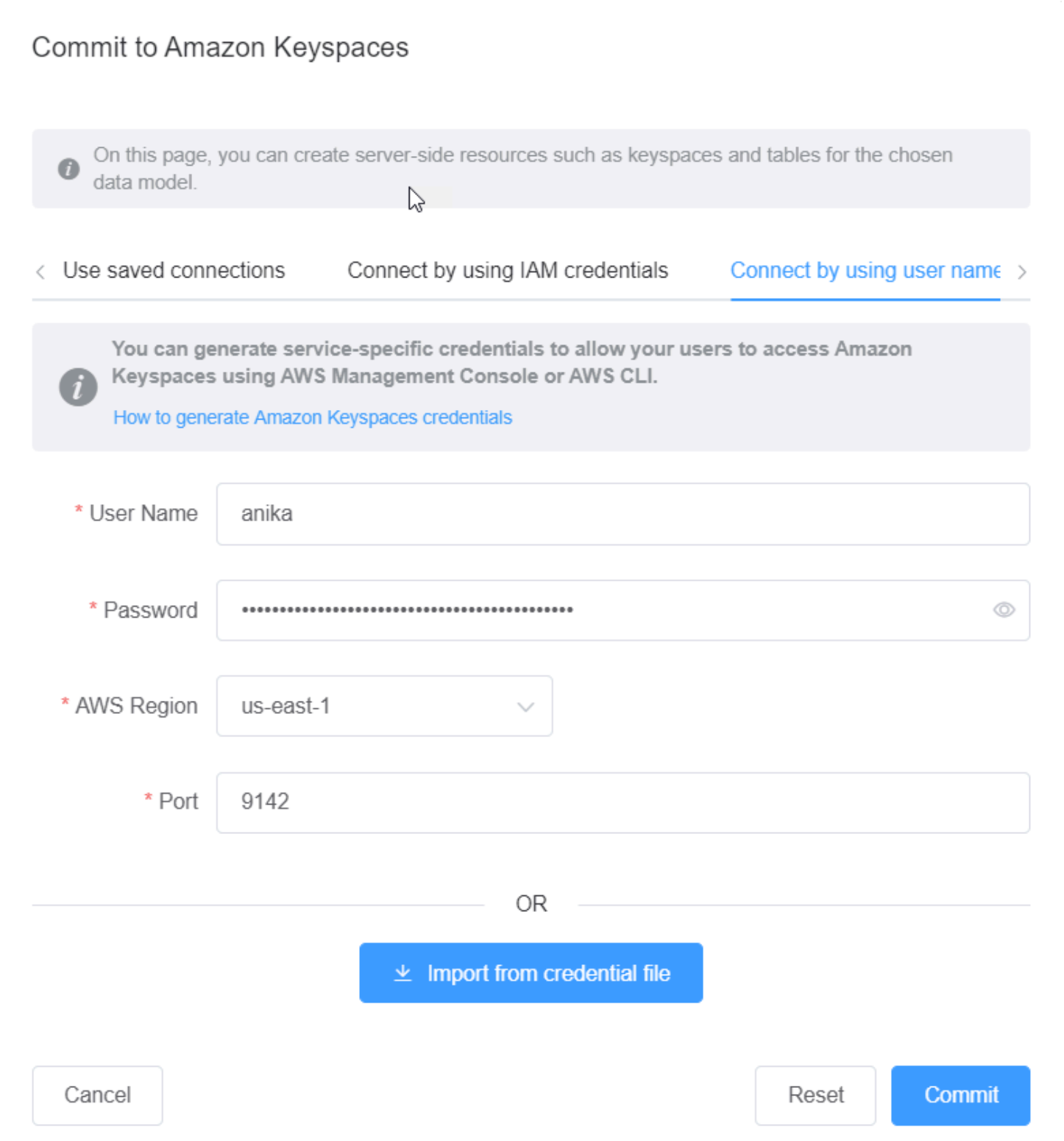
- Antes de começar, é necessário criar credenciais específicas do serviço usando o processo documentado em the section called “Credenciais específicas do serviço”.

Depois de obter as credenciais específicas do serviço, você pode continuar configurando a conexão. Continue com uma das coisas a seguir:

- Nome do usuário – Digite o nome do usuário.
- Senha – Insira a senha.
- Região da AWS – Para ver as Regiões disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.
- Porta – O Amazon Keyspaces usa a porta 9142.

Você também pode importar credenciais salvas de um arquivo.

2. Escolha Confirmar para atualizar o Amazon Keyspaces com o modelo de dados.


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

# Como conectar-se ao Amazon Keyspaces com credenciais (IAM) AWS Identity and Access Management

Esta seção mostra como usar credenciais do IAM para confirmar o modelo de dados criados ou editados com o NoSQL Workbench.

1. Para criar uma nova conexão usando as credenciais do IAM, escolha a guia Conectar usando as credenciais do IAM.

   - Antes de começar, é necessário criar credenciais do IAM usando um dos métodos a seguir.

     - Para acessar o console, use seu nome de usuário e senha do IAM para fazer login no AWS Management Console da página de login do IAM. Para obter informações sobre credenciais de segurança da AWS, incluindo acesso programático e alternativas às credenciais de longo prazo, consulte Credenciais de segurança da AWS no Guia do usuário do IAM. Para obter detalhes sobre como fazer login na Conta da AWS, consulte Como fazer login AWS no Guia do usuário da Início de Sessão da AWS.
     - Para acesso à CLI, você precisa de um ID de chave de acesso e de uma chave de acesso secreta. Use credenciais temporárias em vez de chaves de acesso de longo prazo quando possível. As credenciais temporárias incluem um ID de acesso, uma chave de acesso secreta e um token de segurança que indica quando as credenciais expiram. Para obter mais informações, consulte Usar credenciais temporárias com recursos da AWS no Guia do usuário do IAM.
     - Para acesso à API, é necessário ter um ID de chave de acesso e uma chave de acesso secreta. Use as chaves de acesso do usuário do IAM em vez das chaves de acesso da Usuário raiz da conta da AWS. Para obter mais informações sobre a criação de chaves de acesso, consulte Gerenciar chaves de acesso para usuários do IAM no Guia do usuário do IAM.

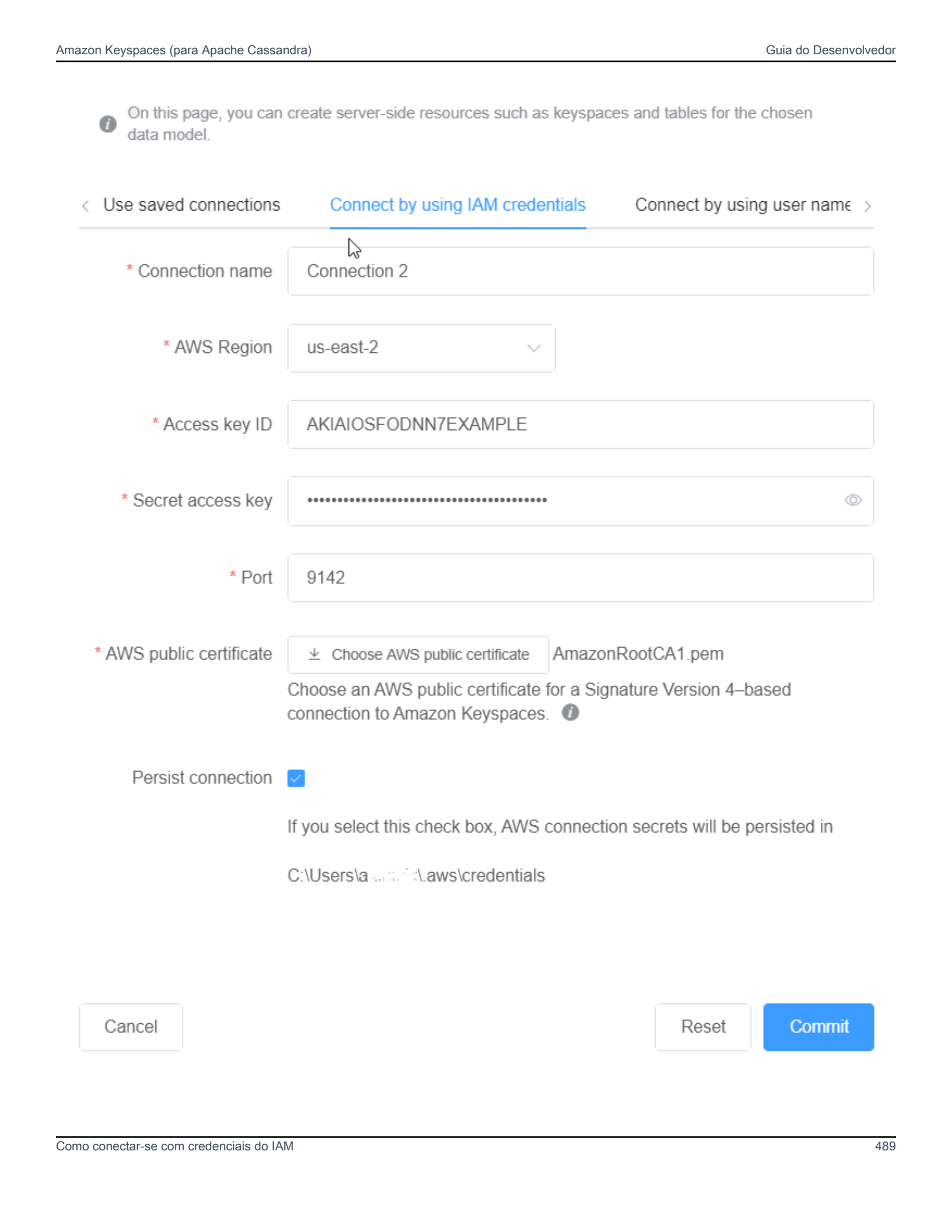
     Para obter mais informações, consulte Gerenciar chaves de acesso para usuários do IAM.

   Depois de obter as credenciais do IAM, você pode continuar configurando a conexão.

   - Nome da conexão – O nome da conexão.
   - Região da AWS – Para ver as Regiões disponíveis, consulte the section called “Service endpoints (Endpoints de serviço)”.

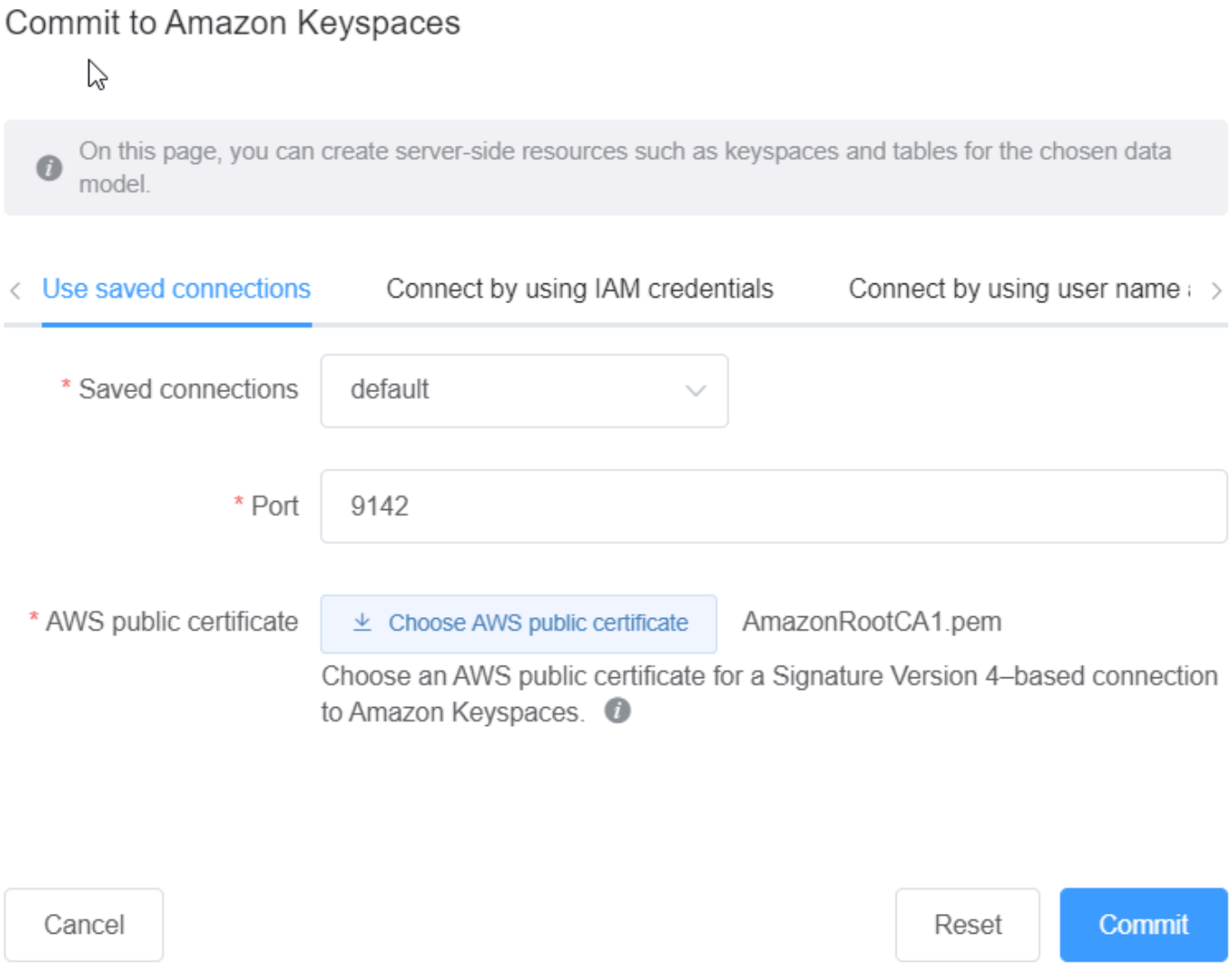
- ID da chave de acesso – Insira o ID da chave de acesso.
- Chave de acesso secreta – Insira a chave de acesso secreta.
- Porta – O Amazon Keyspaces usa a porta 9142.
- AWS certificado público – Aponte para o certificado da AWS que foi baixado na primeira etapa.
- Persistir conexão – Marque essa caixa de seleção se quiser salvar os segredos da conexão da AWS localmente.

2. Escolha Confirmar para atualizar o Amazon Keyspaces com o modelo de dados.


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

## Como usar uma conexão salva

Se você já configurou uma conexão com o Amazon Keyspaces, você pode usá-la como a conexão padrão para confirmar alterações no modelo de dados. Selecione a guia Usar conexões salvas e continue confirmando as atualizações.

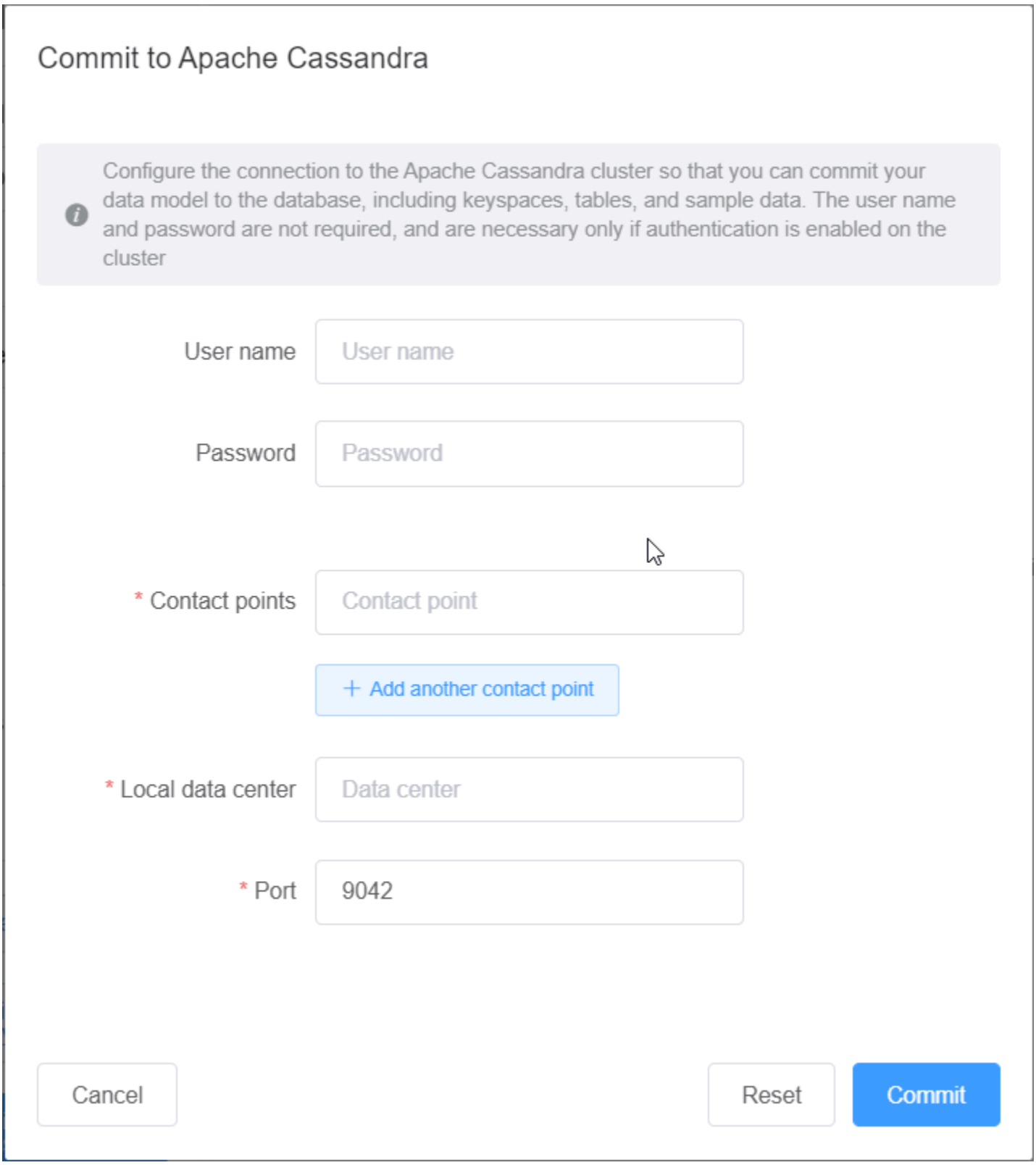



## Como confirmar o Apache Cassandra

Esta seção explica como fazer as conexões com um cluster do Apache Cassandra para confirmar o modelo de dados criado ou editado com o NoSQL Workbench.

> **Note**
>
> Somente modelos de dados que foram criados com `SimpleStrategy` ou `NetworkTopologyStrategy` podem ser confirmados com clusters Apache Cassandra. Para alterar a estratégia de replicação, edite o espaço de chaves no modelador de dados.

1. 
   - Nome de usuário – Insira o nome do usuário se a autenticação estiver habilitada no cluster.
   - Senha – Insira a senha se a autenticação estiver habilitada no cluster.
   - Pontos de contato – Insira os pontos de contato.
   - Datacenter local – Insira o nome do datacenter local.
   - Porta – A conexão usa a porta 9042.
2. Selecione Confirmar para atualizar o cluster Apache Cassandra com o modelo de dados.


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

# Modelos de dados de amostra no NoSQL Workbench

A página inicial do modelador e visualizador exibe uma série de exemplos de modelos fornecidos com o NoSQL Workbench. Esta seção descreve esses modelos e seus usos potenciais.

Tópicos


- Modelo de dados de funcionários
- Modelo de dados de transações com cartão de crédito
- Modelo de dados de operações de companhias aéreas


## Modelo de dados de funcionários

Esse modelo de dados representa um esquema do Amazon Keyspaces para um aplicativo de banco de dados de funcionários.

Os aplicativos que acessam as informações dos funcionários de uma determinada empresa podem usar esse modelo de dados.

Os padrões de acesso suportados por este modelo de dados são:

- Recuperação de um registro de funcionário com um determinado ID.
- Recuperação de um registro de funcionário com um determinado ID e divisão.
- Recuperação de um registro de funcionário com um determinado ID e nome.

## Modelo de dados de transações com cartão de crédito

Esse modelo de dados representa um esquema do Amazon Keyspaces para transações com cartão de crédito em lojas de varejo.

O armazenamento de transações com cartão de crédito não apenas ajuda as lojas na contabilidade, mas também ajuda os gerentes a analisarem as tendências de compra, o que pode ajudá-los na previsão e no planejamento.

Os padrões de acesso suportados por este modelo de dados são:

- Recuperação de transações por número de cartão de crédito, mês, ano e data.

- Recuperação de transações por número de cartão de crédito, categoria e data.
- Recuperação de transações por categoria, localização e número de cartão de crédito.
- Recuperação de transações por número de cartão de crédito e status de disputa.

### Modelo de dados de operações de companhias aéreas

Esse modelo de dados mostra dados sobre voos, incluindo aeroportos, companhias aéreas e rotas de voo.

Os principais componentes da modelagem do Amazon Keyspaces que são demonstrados são pares de valores/chave, armazenamentos de dados de colunas amplas, chaves compostas e tipos de dados complexos, como mapas para demonstrar padrões comuns de acesso a dados NoSQL.

Os padrões de acesso suportados por este modelo de dados são:

- Recuperação de rotas provenientes de uma determinada companhia aérea em um determinado aeroporto.
- Recuperação de rotas com um determinado aeroporto de destino.
- Recuperação de aeroportos com voos diretos.
- Recuperação de detalhes do aeroporto e da companhia aérea.

## Histórico de versões do NoSQL Workbench

A tabela a seguir descreve as alterações importantes em cada versão do aplicativo do lado do cliente NoSQL Workbench.

| Alteração | Descrição | Data |
| --- | --- | --- |
| NoSQL Workbench para Amazon Keyspaces – GA. | O NoSQL Workbench para Amazon Keyspaces está em disponibilidade geral. | 28 de outubro de 2020 |
| A demonstração do NoSQL Workbench foi lançada. | O NoSQL Workbench é um aplicativo do lado do cliente que ajuda você a projetar e visualizar modelos de | 5 de outubro de 2020 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| | dados não relacionais para o Amazon Keyspaces com mais facilidade. Os clientes do NoSQL Workbench estão disponíveis para Windows, macOS e Linux. Para obter mais informações, consulte NoSQL Workbench para Amazon Keyspaces. | |

# Replicação multirregional para o Amazon Keyspaces (para Apache Cassandra

Você pode usar a replicação multirregional do Amazon Keyspaces para replicar seus dados com replicação ativa-ativa automatizada, totalmente gerenciada e em qualquer lugar de sua escolha. Regiões da AWS Com a replicação ativa-ativa, cada Região é capaz de realizar leituras e gravações isoladamente. Você pode melhorar a disponibilidade e a resiliência da degradação regional, além de se beneficiar de leituras e gravações locais de baixa latência para aplicativos globais.

Com a replicação multirregional, o Amazon Keyspaces replica dados de forma assíncrona entre Regiões, e os dados normalmente são propagados entre Regiões em um segundo. Além disso, com a replicação multirregional, você não tem mais o árduo trabalho de resolver conflitos e corrigir problemas de divergência de dados, para que possa se concentrar na aplicação.

Por padrão, o Amazon Keyspaces replica dados em três zonas de disponibilidade dentro da mesma Região da AWS para maior durabilidade e alta disponibilidade. Com a replicação multirregional, você pode criar espaços chave multirregionais que replicam suas tabelas em até seis regiões geográficas diferentes de sua escolha. Regiões da AWS

Tópicos


- Benefícios do uso da replicação multirregional
- Modos de capacidade e preços
- Como a replicação multirregional funciona no Amazon Keyspaces
- Notas de uso da replicação multirregional do Amazon Keyspaces
- Como usar a replicação multirregional


## Benefícios do uso da replicação multirregional

A replicação multirregional oferece os seguintes benefícios.

- Leituras e gravações globais com latência de um dígito em milissegundos — No Amazon Keyspaces, a replicação é ativa-ativa. Você pode fornecer leituras e gravações localmente nas Regiões mais próximas de seus clientes, com latência de um dígito de milissegundos em qualquer escala. Você pode usar tabelas multirregionais do Amazon Keyspaces para aplicações globais que precisam de um tempo de resposta rápido em qualquer lugar do mundo.

- Continuidade de negócios aprimorada e proteção contra a degradação em uma única região — Com a replicação multirregional, você pode se recuperar da degradação em uma única região Região da AWS redirecionando seu aplicativo para uma região diferente em seu keyspace multirregional. Como o Amazon Keyspaces oferece replicação ativa-ativa, não há impacto em suas leituras e gravações.

  O Amazon Keyspaces acompanhará as gravações executadas em seu espaço de chaves multirregionais que não foram propagadas para todas as Regiões da réplica. Depois que a região voltar a ficar on-line, o Amazon Keyspaces sincroniza automaticamente todas as alterações ausentes para que você possa se recuperar sem nenhum impacto no aplicativo.
- Replicação de alta velocidade entre regiões — A replicação multirregional usa replicação física rápida e baseada em armazenamento de dados entre regiões, com um atraso de replicação que normalmente é inferior a 1 segundo.

  A replicação no Amazon Keyspaces tem pouco ou nenhum impacto em suas consultas de banco de dados porque não compartilha recursos computacionais com seu aplicativo. Isso significa que você pode lidar com casos de uso de alta taxa de gravação ou casos de uso com picos ou picos repentinos na taxa de transferência sem nenhum impacto no aplicativo.
- Consistência e resolução de conflitos — Todas as alterações feitas nos dados em qualquer região são replicadas para outras regiões em um espaço chave multirregional. Conflitos poderão ocorrer se as aplicações atualizarem o mesmo item em Regiões diferentes e quase ao mesmo tempo.

  Para ajudar a fornecer consistência eventual, o Amazon Keyspaces usa a data e a hora em nível de célula e um último gravador obtém a reconciliação entre atualizações simultâneas. A resolução de conflitos é totalmente gerenciada e ocorre em segundo plano, sem nenhum impacto no aplicativo.

Para obter mais informações sobre configurações e atributos suportados, consulte the section called “Observações de uso”.

## Modos de capacidade e preços

Para um keyspace multirregional, você pode usar o modo de capacidade sob demanda ou o modo de capacidade provisionada. Para ter mais informações, consulte the section called “Modos de capacidade de leitura/gravação”.

No modo sob demanda, você paga 1,25 unidades de solicitação de gravação (WRUs) para gravar até 1 KB de dados por linha. Você é cobrado pelas gravações em cada região do seu keyspace multirregional. Por exemplo, gravar uma linha de 3 KB de dados em um espaço de chaves multirregional com duas Regiões exige 7,5 WRUs: 3 * 1,25 * 2 = 7,5 WRUs. Além disso, gravações que incluem dados estáticos e não estáticos exigem operações adicionais de gravação.

No modo provisionado, são cobradas 1,25 unidades de capacidade de gravação (WCUs) para gravar até 1 KB de dados por linha. Você é cobrado pelas gravações em cada região do seu keyspace multirregional. Por exemplo, gravar uma linha de 3 KB de dados por segundo em um keyspace multirregional com duas regiões requer 7,5 WCUs: 3 * 1,25 * 2 = 7,5 WCUs. Além disso, gravações que incluem dados estáticos e não estáticos exigem operações adicionais de gravação.

Para obter mais informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

# Como a replicação multirregional funciona no Amazon Keyspaces

Esta seção apresenta uma visão geral de como a replicação multirregional do Amazon Keyspaces funciona. Para obter mais informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como a replicação multirregional funciona no Amazon Keyspaces
- Resolução de conflitos de replicação multirregional
- Recuperação de desastres de replicação multirregional
- Permissões de IAM necessárias para criar e espaços de chaves e tabelas multirregionais.
- Replicação multirregional e integração com point-in-time recuperação (PITR)
- Replicação multirregional e integração com serviços da AWS

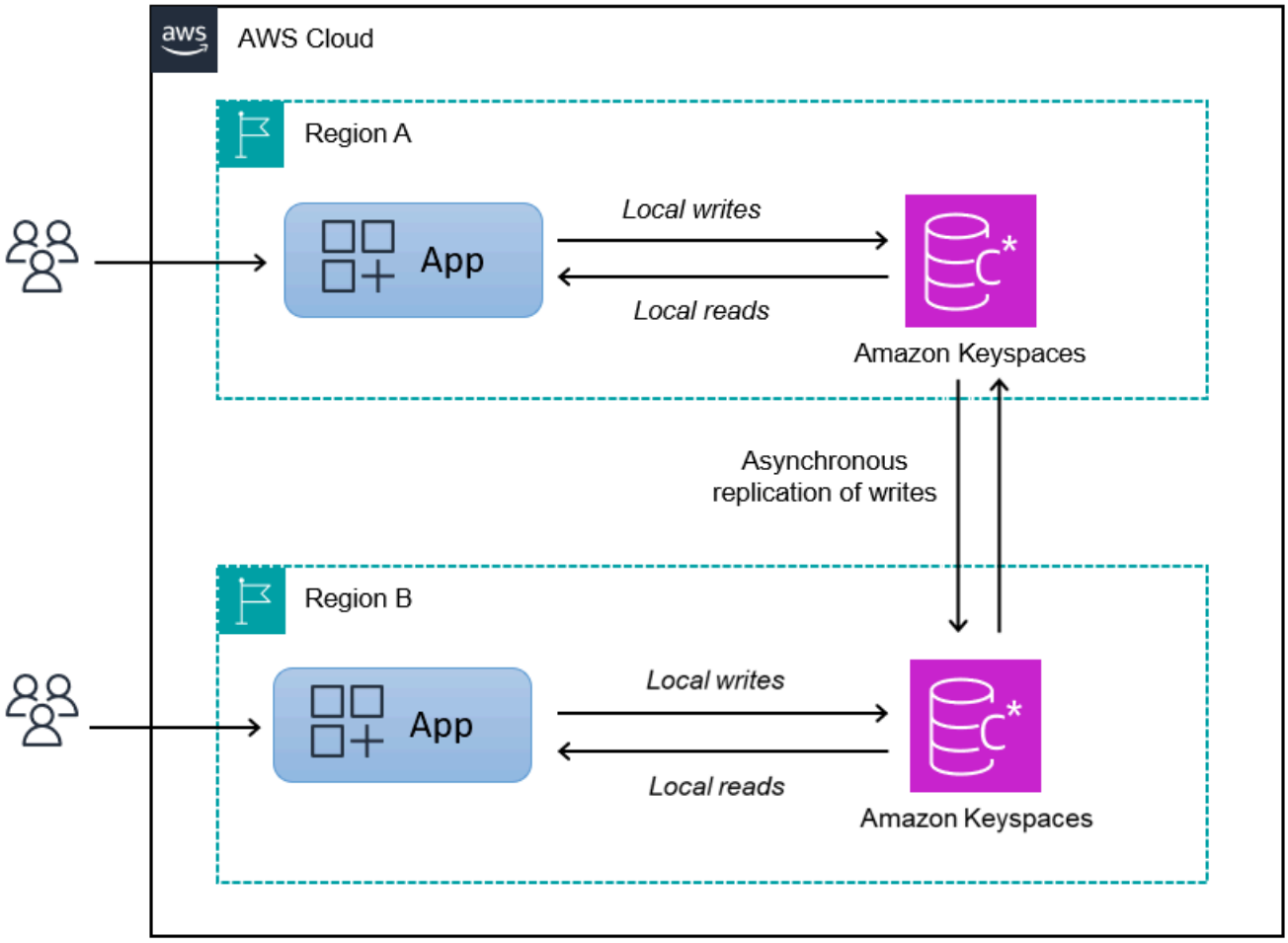

## Como a replicação multirregional funciona no Amazon Keyspaces

A replicação multirregional do Amazon Keyspaces implementa uma arquitetura de resiliência de dados que distribui seus dados de forma independente e distribuída geograficamente nas Regiões da AWS. Ele usa replicação ativa-ativa, que fornece baixa latência local, com cada região sendo capaz de realizar leituras e gravações isoladamente.

Ao criar um espaço de chaves multirregional do Amazon Keyspaces, você pode selecionar até cinco Regiões adicionais para as quais os dados serão replicados. Cada tabela criada em um espaço de chaves multirregional consiste em várias tabelas-réplica (uma por Região) que o Amazon Keyspaces considera como uma única unidade.

Cada réplica possui o mesmo nome de tabela e o mesmo esquema de chave primária. Quando uma aplicação grava dados em uma tabela local em uma Região, os dados são gravados de forma permanente usando o nível de consistência `LOCAL_QUORUM`. O Amazon Keyspaces replica automaticamente os dados de forma assíncrona para as outras Regiões de replicação. O atraso de replicação entre Regiões geralmente é inferior a um segundo e não afeta o desempenho nem o throughput do seu aplicativo.

Depois que os dados forem gravados, você poderá lê-los na tabela multirregional em outra Região de replicação com os níveis de consistência `LOCAL_ONE/LOCAL_QUORUM`. Para obter mais informações sobre configurações e atributos compatíveis, consulte the section called “Observações de uso”.



## Resolução de conflitos de replicação multirregional

A replicação multirregional do Amazon Keyspaces é totalmente gerenciada, o que significa que você não precisa realizar tarefas de replicação, como executar regularmente operações de reparo para resolver problemas de sincronização de dados. O Amazon Keyspaces monitora a consistência de dados entre tabelas em diferentes, Regiões da AWS detectando e reparando conflitos, e sincroniza réplicas automaticamente.

O Amazon Keyspaces usa o método de reconciliação de dados do último gravador vence. Com esse mecanismo de resolução de conflitos, todas as Regiões em um espaço de chaves multirregional concordam com a atualização mais recente e convergem para um estado em que todas possuem dados idênticos. O processo de reconciliação não tem impacto no desempenho do aplicativo. Para apoiar a resolução de conflitos, a data e a hora do lado do cliente são ativadas automaticamente para tabelas multirregionais e não podem ser desativadas. Para ter mais informações, consulte Carimbos de data/hora do lado do cliente.

## Recuperação de desastres de replicação multirregional

Com a replicação multirregional do Amazon Keyspaces, as gravações são replicadas de forma assíncrona em cada região. No caso raro de degradação ou falha em uma única Região, a replicação multirregional ajuda você a se recuperar de um desastre com pouco ou nenhum impacto em seu aplicativo. A recuperação de desastres é normalmente medida usando valores de objetivo de tempo de recuperação (RTO) e objetivo de ponto de recuperação (RPO).

Objetivo de tempo de recuperação: o tempo que um sistema leva para retornar a um estado de trabalho após um desastre. O RTO mede a quantidade de tempo de inatividade sua workload pode tolerar, medida em tempo. Para planos de recuperação de desastres que usam a replicação multirregional para fazer failover para uma Região não afetada, o RTO pode ser quase zero. O RTO é limitado pela rapidez com que seu aplicativo pode detectar a condição de falha e redirecionar o tráfego para outra Região.

Objetivo de ponto de recuperação: a quantidade de dados que podem ser perdidos (medidos no tempo). Para planos de recuperação de desastres que usam a replicação multirregional para fazer failover para uma região não afetada, o RPO geralmente é de um dígito de segundos. O RPO é limitado pela latência de replicação para a réplica de destino do failover.

No caso de uma falha ou degradação Regional, você não precisa promover uma região secundária ou realizar procedimentos de failover do banco de dados porque a replicação no Amazon Keyspaces

é ativa-ativa. Em vez disso, você pode usar o Amazon Route 53 para encaminhar a aplicação para a Região íntegra mais próxima. Para saber mais sobre o Route 53, consulte O que é o Amazon Route 53? .

Se uma única Região da AWS ficar isolada ou degradada, seu aplicativo poderá redirecionar o tráfego para uma região diferente usando o Route 53 para realizar leituras e gravações em uma tabela de réplica diferente. Você também pode aplicar lógica de negócios personalizada para determinar quando redirecionar solicitações para outras regiões. Um exemplo disso é tornar seu aplicativo ciente dos vários endpoints disponíveis.

Quando a Região voltar a ficar online, o Amazon Keyspaces retomará a propagação de todas as gravações pendentes dessa região para as tabelas-réplica nas outras regiões. Ele também retomará a propagação de gravações de outras tabelas-réplica para a região que está online novamente.

## Permissões de IAM necessárias para criar e espaços de chaves e tabelas multirregionais.

Para criar tabelas e espaços de chaves multirregionais bem-sucedidos, a entidade principal do IAM precisa ser capaz de criar uma função vinculada ao serviço. Essa função vinculada ao serviço é um tipo exclusivo de perfil do IAM predefinida pelo Amazon Keyspaces. Isso inclui todas as permissões que o Amazon Keyspaces exige para executar ações em seu nome. Para obter mais informações sobre a função vinculada ao serviço, consulte the section called “Replicação multirregional”.

Para criar a função vinculada ao serviço exigida pela replicação multirregional, a política da entidade principal do IAM exige os seguintes elementos:

- `iam:CreateServiceLinkedRole`: a ação que a entidade principal pode executar.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`: o recurso no qual as ações podem ser executadas.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`— O único AWS serviço ao qual essa função pode ser associada é o Amazon Keyspaces.

Veja a seguir um exemplo da política que concede as permissões mínimas necessárias a uma entidade principal para criar tabelas e espaços de chaves multirregionais.

```
{
```

```
            "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"}}
}
```

Para obter permissões adicionais do IAM para espaços de chaves e tabelas multirregionais, consulte Ações, recursos e chaves de condição do Amazon Keyspaces (para Apache Cassandra) na Referência de autorização do serviço.

## Replicação multirregional e integração com point-in-time recuperação (PITR)

A oint-in-time recuperação P é suportada em tabelas multirregionais. Para restaurar com êxito uma tabela multirregional com PITR, as seguintes condições devem ser atendidas.

- As tabelas de origem e de destino devem ser configuradas como tabelas multirregionais.
- As Regiões de replicação para o espaço de chaves da tabela de origem e para o espaço de chaves da tabela de destino devem ser as mesmas.

Você pode executar a instrução de restauração de qualquer uma das Regiões em que a tabela de origem está disponível. O Amazon Keyspaces restaura automaticamente a tabela de destino em cada Região. Para ter mais informações sobre o PITR, consulte the section called “Como funciona”.

## Replicação multirregional e integração com serviços da AWS

Você pode monitorar o desempenho da replicação entre tabelas diferentes Regiões da AWS usando CloudWatch métricas da Amazon. A métrica a seguir fornece monitoramento contínuo de espaços de chaves multirregionais.

- `ReplicationLatency`: essa métrica mede o tempo necessário para replicar `updates`, `inserts` ou `deletes` de uma tabela de réplica para outra tabela de réplica em um espaço de chaves multirregional.

Para obter mais informações sobre como monitorar CloudWatch métricas, consultethe section called “Monitoramento com CloudWatch”.

# Notas de uso da replicação multirregional do Amazon Keyspaces

Considere o seguinte ao usar a replicação multirregional com o Amazon Keyspaces.

- Você pode selecionar até seis pessoas do público disponível Regiões da AWS. AWS GovCloud (US) Regions, Regiões da China e Regiões da AWS que estão desativadas por padrão não são suportadas.
- Selecione as Regiões de replicação para o espaço de chaves com cuidado, pois você não pode adicioná-las ou removê-las posteriormente.
- Finalize o esquema da tabela antes de criar uma tabela multirregional, pois você não poderá adicionar novas colunas posteriormente.
- Para criptografia em repouso, use uma AWS chave própria. As chaves gerenciadas pelo cliente não são compatíveis com tabelas multirregionais. Para obter mais informações, consulte

  the section called “Como funciona”.
- Ao usar o gerenciamento de capacidade provisionada com o escalonamento automático do Amazon Keyspaces, certifique-se de usar as operações de API do Amazon Keyspaces para criar e configurar suas tabelas multirregionais. As operações subjacentes da API Application Auto Scaling que o Amazon Keyspaces chama em seu nome não têm recursos multirregionais.

  Para ter mais informações, consulte the section called “Como usar a replicação multirregional”. Para obter mais informações sobre como estimar a taxa de transferência da capacidade de gravação de tabelas multirregionais provisionadas, consulte. the section called “Tabelas multirregionais”
- Decida se a tabela precisa de vida útil (TTL). Não será possível ativá-lo mais tarde. Para ter mais informações, consulte Dados expirados com a vida útil.
- Embora os dados sejam replicados automaticamente nas Regiões selecionadas de uma tabela multirregional, quando um cliente se conecta a um endpoint em uma Região e consulta a tabela `system.peers`, a consulta retorna somente informações locais. O resultado da consulta aparece como um único cluster de datacenter para o cliente.
- A replicação multirregional do Amazon Keyspaces é assíncrona e oferece suporte à consistência para gravações. `LOCAL_QUORUM` `LOCAL_QUORUM`a consistência exige que uma atualização em uma linha seja mantida de forma duradoura em duas réplicas na região local antes de retornar o sucesso ao cliente. A propagação das gravações na região (ou regiões) replicada é então executada de forma assíncrona.

A replicação multirregional do Amazon Keyspaces não oferece suporte à replicação síncrona nem à consistência. `QUORUM`

- Quando você cria uma tabela ou espaço de chaves multirregional, todas as tags que você define durante o processo de criação são aplicadas automaticamente a todos os espaços de chaves e tabelas em todas as Regiões. Quando você altera as tags existentes usando `ALTER KEYSPACE` ou`ALTER TABLE`, a atualização é aplicada somente ao espaço de teclas ou à tabela na região em que você está fazendo a alteração.
- A Amazon CloudWatch fornece uma `ReplicationLatency` métrica para cada região replicada. Ele calcula essa métrica rastreando as linhas que chegam, comparando o tempo de chegada com o tempo de gravação inicial e calculando uma média. Os horários são armazenados CloudWatch na região de origem. Para ter mais informações, consulte the section called “Monitoramento com CloudWatch”.

  Pode ser útil visualizar os tempos médio e máximo para determinar o atraso de replicação médio e o pior caso. Não há SLA sobre essa latência.
- Ao usar uma tabela multirregional no modo sob demanda, você pode observar um aumento na latência da replicação assíncrona de gravações se uma réplica de tabela apresentar um novo pico de tráfego. Da mesma forma que o Amazon Keyspaces adapta automaticamente a capacidade de uma tabela sob demanda de uma única região ao tráfego do aplicativo que ela recebe, o Amazon Keyspaces adapta automaticamente a capacidade de uma réplica de tabela sob demanda de várias regiões ao tráfego que ela recebe. O aumento na latência de replicação é transitório porque o Amazon Keyspaces aloca automaticamente mais capacidade à medida que seu volume de tráfego aumenta. Depois que todas as réplicas se adaptarem ao seu volume de tráfego, a latência da replicação deverá voltar ao normal. Para ter mais informações, consulte the section called “Tráfego de pico e propriedades de dimensionamento”.
- Ao usar uma tabela multirregional no modo provisionado, se seu aplicativo exceder sua capacidade de taxa de transferência provisionada, você poderá observar erros de capacidade insuficientes e um aumento na latência de replicação. Para garantir que sempre haja capacidade suficiente de leitura e gravação para todas as réplicas Regiões da AWS de tabelas em toda uma tabela multirregional, recomendamos que você configure o escalonamento automático do Amazon Keyspaces. O auto scaling do Amazon Keyspaces ajuda você a provisionar a capacidade de taxa de transferência de forma eficiente para cargas de trabalho variáveis, ajustando a capacidade de transferência automaticamente em resposta ao tráfego real do aplicativo. Para ter mais informações, consulte the section called “Como o escalonamento automático funciona para tabelas multirregionais”.

# Como usar a replicação multirregional

Você pode criar e gerenciar tabelas e espaços de chave multirregionais usando o console Amazon Keyspaces (para Apache Cassandra), a Cassandra Query Language (CQL), o SDK e o (). AWS AWS Command Line Interface AWS CLI

Esta seção fornece exemplos de como criar tabelas e espaços de chave multirregionais com o console, com o CQL e com o AWS CLI, usando o modo de capacidade provisionada e sob demanda. Todas as tabelas criadas em um espaço de chave multirregional herdam automaticamente as configurações de várias regiões do espaço de chave.

Esta seção também inclui exemplos de como usar o console, o CQL e o AWS CLI para gerenciar as configurações de auto scaling do Amazon Keyspaces de tabelas multirregionais provisionadas. Para obter mais informações sobre as opções gerais de configuração do auto scaling e como elas funcionam, consultethe section called “Gerencie a capacidade de produção com escalabilidade automática”.

Observe que, se você estiver usando o modo de capacidade provisionada para tabelas multirregionais, você deve sempre usar as chamadas de API do Amazon Keyspaces para configurar o auto scaling. Isso ocorre porque as operações subjacentes da API Application Auto Scaling não reconhecem a região.

Para obter mais informações sobre como estimar a taxa de transferência da capacidade de gravação de tabelas multirregionais provisionadas, consulte. the section called “Tabelas multirregionais”

Para obter mais informações sobre a API Amazon Keyspaces, consulte Amazon Keyspaces API Reference.

Para obter mais informações sobre as configurações suportadas e os recursos de replicação multirregional, consulte. the section called “Observações de uso”

Tópicos

- Usando o console para criar e gerenciar tabelas multirregionais
- Usando o CQL para criar e gerenciar tabelas multirregionais
- Usando o AWS CLI para criar e gerenciar tabelas multirregionais

# Usando o console para criar e gerenciar tabelas multirregionais

Esta seção fornece exemplos de como criar tabelas e espaços chave multirregionais no modo de capacidade provisionada e sob demanda usando o console Amazon Keyspaces (para Apache Cassandra). Todas as tabelas que você cria em um espaço de chave multirregional herdam automaticamente as configurações de várias regiões do espaço de chave.

Para exemplos de CQL, consultethe section called “Usar SSL”. Para obter AWS CLI exemplos, consultethe section called “Usando o AWS CLI”.

Tópicos

- Criar um espaço de chaves multirregional (console)
- Criação de uma tabela multirregional com configurações padrão (console)
- Criação de uma tabela multirregional no modo provisionado com o escalonamento automático ativado (console)
- Habilitando o escalonamento automático para uma tabela multirregional existente (console)
- Desativando o escalonamento automático para uma tabela multirregional (console)
- Visualização das atividades de auto scaling do Amazon Keyspaces no console

## Criar um espaço de chaves multirregional (console)

Siga estas etapas para criar um novo keyspace multirregional usando o console do Amazon Keyspaces.

Para criar um espaço de chaves multirregional (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Espaços de chaves e, em seguida, escolha Criar espaço de chaves.
3. Em Nome do espaço de chaves, insira o nome do espaço de chaves.
4. Na seção Replicação multirregional, você pode adicionar até cinco Regiões adicionais que estão disponíveis na lista.
5. Para finalizar, selecione Criar espaço de chaves.

> **Note**
>
> Ao criar um espaço de chaves multirregional, o Amazon Keyspaces cria uma função vinculada ao serviço com o nome `AWSServiceRoleForAmazonKeyspacesReplication` em sua conta. Essa função permite que o Amazon Keyspaces replique gravações em todas as réplicas de uma tabela multirregional em seu nome. Para saber mais, consulte the section called “Replicação multirregional”.

## Criação de uma tabela multirregional com configurações padrão (console)

Siga estas etapas para criar uma tabela multirregional com o console do Amazon Keyspaces.

Para criar uma tabela multirregional (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Selecione um espaço de chaves multirregional.
3. Na guia Tabelas, selecione Criar tabela.
4. Em Nome da tabela insira o nome para a nova tabela. As Regiões da AWS em que essa tabela está sendo replicada são mostradas na caixa de informações.
5. Continue com o esquema da tabela.
6. Em Configurações da tabela, continue com a opção Configurações padrão. Observe as seguintes configurações padrão para tabelas multirregionais.
   - Modo de capacidade — O modo de capacidade padrão é Sob demanda. Para obter mais informações sobre como configurar o modo provisionado, consulte. the section called “Criação de uma tabela multirregional no modo provisionado com o escalonamento automático ativado (console)”
   - Gerenciamento de chaves de criptografia –Apenas a opção Chave pertencente à AWS é suportada.
   - Data e hora do lado do cliente – Esse atributo é necessário para tabelas multirregionais.

- Selecione Personalizar configurações se precisar ativar a vida útil (TTL) para a tabela e todas as suas réplicas.

> **Note**
>
> Você não poderá alterar as configurações de TTL em uma tabela multirregional existente.

7. Para finalizar, selecione Criar tabela.

## Criação de uma tabela multirregional no modo provisionado com o escalonamento automático ativado (console)

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para criar uma nova tabela multirregional com o escalonamento automático ativado

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Selecione um espaço de chaves multirregional.
3. Na guia Tabelas, selecione Criar tabela.
4. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a nova tabela.
5. Na seção Colunas, crie o esquema para sua tabela.
6. Na seção Chave primária, defina a chave primária da tabela e selecione colunas de agrupamento opcionais.
7. Na seção Configurações da tabela, selecione Personalizar configurações.

8. Continue com as configurações de capacidade de leitura/gravação.
9. Para o Modo de capacidade, escolha Provisionado.
10. Na seção Capacidade de leitura, confirme se a opção Escalar automaticamente está selecionada.

    Você pode optar por configurar as mesmas unidades de capacidade de leitura para tudo em Regiões da AWS que a tabela é replicada. Como alternativa, você pode desmarcar a caixa de seleção e configurar a capacidade de leitura para cada região de forma diferente.

    Se você optar por configurar cada região de forma diferente, selecione as unidades de capacidade de leitura mínima e máxima para cada réplica da tabela, bem como a utilização desejada.

    - Unidades de capacidade mínima - insira o valor do nível mínimo de throughput que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).
    - Unidades de capacidade máxima — insira a quantidade máxima de taxa de transferência que você deseja provisionar para a tabela. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).
    - Utilização desejada- insira uma taxa de utilização desejada entre 20% e 90%. Quando o tráfego excede a taxa de utilização desejada definida, a capacidade é automaticamente aumentada. Quando o tráfego fica abaixo da meta definida, ela é automaticamente reduzida novamente.

    - Desmarque a caixa de seleção Dimensionar automaticamente se quiser provisionar a capacidade de leitura da tabela manualmente. Essa configuração se aplica a todas as réplicas da tabela.

      > **Note**
      >
      > Para garantir que haja capacidade de leitura suficiente para todas as réplicas, recomendamos a escalabilidade automática do Amazon Keyspaces para tabelas multirregionais provisionadas.

> **Note**
>
> Para saber mais sobre as cotas padrão da sua conta e como aumentá-las, consulte Cotas.

11. Na seção Capacidade de gravação, confirme se a opção Escalar automaticamente está selecionada. Em seguida, configure as unidades de capacidade da tabela. As unidades de capacidade de gravação permanecem sincronizadas em todas as Regiões da AWS regiões para garantir que haja capacidade suficiente para replicar eventos de gravação em todas as regiões.
    - Limpe a escala automaticamente se quiser provisionar a capacidade de gravação da tabela manualmente. Essa configuração se aplica a todas as réplicas da tabela.

      > **Note**
      >
      > Para garantir que haja capacidade de gravação suficiente para todas as réplicas, recomendamos a escalabilidade automática do Amazon Keyspaces para tabelas multirregionais provisionadas.

12. Escolha Create table. Sua tabela é criada com os parâmetros padrão de ajuste de escala automático.

## Habilitando o escalonamento automático para uma tabela multirregional existente (console)

Siga estas etapas para habilitar o auto scaling para uma tabela multirregional no modo provisionado com o console Amazon Keyspaces.

> **Note**
>
> O ajuste de escala automático do Amazon Keyspaces requer a presença de um perfil vinculado ao serviço (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realize ações de ajuste em seu nome. Esta função é criada automaticamente para você. Para ter mais informações, consulte the section called “Usar perfis vinculados ao serviço”.

Para habilitar a escalabilidade automática do Amazon Keyspaces para uma tabela multirregional existente

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela com a qual você deseja trabalhar e vá até a guia Capacidade.
3. Na seção Configurações de capacidade, escolha Editar.
4. Em Modo de capacidade, verifique se a tabela está usando o modo de capacidade provisionada.
5. Selecione Dimensionar automaticamente e veja a etapa 9 Criação de uma tabela multirregional no modo provisionado com o escalonamento automático ativado (console) para editar a capacidade de leitura e gravação.
6. Quando as configurações de escala automática estiverem definidas, escolha Salvar.

## Desativando o escalonamento automático para uma tabela multirregional (console)

Siga estas etapas para desativar o auto scaling para uma tabela multirregional no modo provisionado com o console do Amazon Keyspaces.

Para desativar a escalabilidade automática do Amazon Keyspaces para uma tabela multirregional existente

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela com a qual você deseja trabalhar e escolha a guia Capacidade.
3. Na seção Configurações de capacidade, escolha Editar.
4. Para desativar a escalabilidade automática do Amazon Keyspaces, desmarque a caixa de seleção Escalar automaticamente. A desativação do escalonamento automático cancela o registro da tabela como um destino escalável com o Application Auto Scaling. Para excluir a função vinculada ao serviço que o Application Auto Scaling usa para acessar sua tabela do Amazon Keyspaces, siga as etapas em. the section called “Exclusão de um perfil vinculado ao serviço do Amazon Keyspaces”

> **Note**
>
> Para excluir a função vinculada ao serviço que o Application Auto Scaling usa, você deve desativar o escalonamento automático em todas as tabelas da conta. Regiões da AWS

5. Quando as configurações de escala automática estiverem definidas, escolha Salvar.

### Visualização das atividades de auto scaling do Amazon Keyspaces no console

Você pode monitorar como a escalabilidade automática do Amazon Keyspaces usa recursos usando a Amazon CloudWatch, que gera métricas sobre seu uso e desempenho. Siga as etapas no Guia do Application Auto Scaling usuário para criar um CloudWatch painel.

## Usando o CQL para criar e gerenciar tabelas multirregionais

Você pode usar a Cassandra Query Language (CQL) para criar e gerenciar tabelas e espaços chave multirregionais no Amazon Keyspaces.

Esta seção fornece exemplos de como criar e gerenciar tabelas multirregionais com CQL. Todas as tabelas que você cria em um espaço de chave multirregional herdam automaticamente as configurações de várias regiões do espaço de chave. Para obter mais informações sobre CQL, consulte a referência de linguagem CQL do Amazon Keyspaces.

Para obter mais informações sobre configurações e atributos suportados, consulte the section called “Observações de uso”.

Tópicos

- Como criar um espaço de chaves multirregional (CQL)
- Criação de uma tabela multirregional com configurações padrão (CQL)
- Criação de uma tabela multirregional com modo de capacidade provisionada e escalabilidade automática (CQL)
- Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CQL)
- Visualizando a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional (CQL)

- Desativando o escalonamento automático para uma tabela multirregional (CQL)
- Configurando manualmente a capacidade provisionada de uma tabela multirregional (CQL)

## Como criar um espaço de chaves multirregional (CQL)

Para criar um espaço de chave multirregional, use `NetworkTopologyStrategy` para especificar o espaço de chave em Regiões da AWS que o espaço de chave será replicado. Você deve incluir sua Região atual e pelo menos uma Região adicional. A seguinte instrução CQL é um exemplo disso.

```
CREATE KEYSPACE mykeyspace
WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3' };
```

Todas as tabelas no espaço de chaves usam a mesma estratégia de replicação do espaço de chaves. Não é possível alterar a estratégia de replicação no nível da tabela.

`NetworkTopologyStrategy`— O fator de replicação para cada região é três, porque o Amazon Keyspaces replica dados em três zonas de disponibilidade dentro da Região da AWS mesma, por padrão.

> **Note**
>
> Ao criar um espaço de chaves multirregional, o Amazon Keyspaces cria uma função vinculada ao serviço com o nome `AWSServiceRoleForAmazonKeyspacesReplication` em sua conta. Essa função permite que o Amazon Keyspaces replique gravações em todas as réplicas de uma tabela multirregional em seu nome. Para saber mais, consulte the section called “Replicação multirregional”.

Você pode usar uma instrução CQL para consultar a `tables` tabela no `system_multiregion_info` keyspace para listar programaticamente as regiões e o status da tabela multirregional que você especificar. O código a seguir é um exemplo disso.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

A saída da declaração tem a seguinte aparência:

```
 keyspace_name  | table_name     | region         | status
----------------+----------------+----------------+--------
 mykeyspace     | mytable        | us-east-1      | ACTIVE
 mykeyspace     | mytable        | ap-southeast-1 | ACTIVE
 mykeyspace     | mytable        | eu-west-1      | ACTIVE
```

## Criação de uma tabela multirregional com configurações padrão (CQL)

Para criar uma tabela multirregional com configurações padrão, você pode usar o exemplo a seguir.

```
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'
 }
};
```

## Criação de uma tabela multirregional com modo de capacidade provisionada e escalabilidade automática (CQL)

Para criar uma tabela multirregional no modo provisionado com escalabilidade automática, você deve primeiro especificar o modo de capacidade `CUSTOM_PROPERTIES` definindo para a tabela. Depois de especificar o modo de capacidade provisionada, você pode definir as configurações de escalonamento automático para a tabela usando. `AUTOSCALING_SETTINGS`

Para obter informações detalhadas sobre as configurações de escalonamento automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consultethe section called “Crie uma nova tabela com escalabilidade automática usando CQL”.

Ao criar uma tabela multirregional, você também pode especificar diferentes configurações de capacidade de leitura e escalabilidade automática de leitura para cada réplica da tabela. As

configurações que você especificar substituem as configurações gerais da tabela pelo especificado Região da AWS. A capacidade de gravação, no entanto, permanece sincronizada entre todas as réplicas para garantir que haja capacidade suficiente para replicar gravações em todas as regiões.

Para definir a capacidade de leitura de uma réplica de tabela em uma região específica, você pode configurar os seguintes parâmetros como parte da `replica_updates` tabela:

- A região
- As unidades de capacidade de leitura provisionadas (opcional)
- Configurações de escalonamento automático para capacidade de leitura (opcional)

O exemplo a seguir mostra uma `CREATE TABLE` declaração para uma tabela multirregional no modo provisionado. As configurações gerais de escalonamento automático da capacidade de gravação e leitura são as mesmas. No entanto, as configurações de escalonamento automático de leitura especificam períodos adicionais de resfriamento de 60 segundos antes de aumentar ou diminuir a capacidade de leitura da tabela. Além disso, as configurações de escalonamento automático da capacidade de leitura para a Região Leste dos EUA (Norte da Virgínia) são maiores do que as de outras réplicas. Além disso, o valor alvo é definido como 70% em vez 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
                    }
                }
            }
        }
    }
};
```

## Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CQL)

Você pode usar `ALTER TABLE` para atualizar o modo de capacidade e as configurações de escalonamento automático de uma tabela existente. Se você estiver atualizando uma tabela que está atualmente no modo de capacidade sob demanda, `capacity_mode` é necessário. Se sua tabela já estiver no modo de capacidade provisionada, esse campo poderá ser omitido.

Para obter informações detalhadas sobre as configurações de escalonamento automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consultethe section called “Crie uma nova tabela com escalabilidade automática usando CQL”.

Na mesma declaração, você também pode atualizar as configurações de capacidade de leitura e escalonamento automático de réplicas de tabela em regiões específicas atualizando a propriedade da `replica_updates` tabela. A instrução a seguir é um exemplo disso.

```
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
                    }
                }
            }
        }
    }
};
```

## Visualizando a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional (CQL)

Para visualizar a configuração de escalonamento automático de uma tabela multirregional, use o comando a seguir.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name = 'mykeyspace'
 AND table_name = 'mytable';
```

A saída desse comando é semelhante à seguinte:

```
 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}}}
```

## Desativando o escalonamento automático para uma tabela multirregional (CQL)

Você pode usar `ALTER TABLE` para desativar o escalonamento automático de uma tabela existente. Observe que você não pode desativar o escalonamento automático para uma réplica de tabela individual.

No exemplo a seguir, o escalonamento automático está desativado para a capacidade de leitura da tabela.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_read_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Para excluir a função vinculada ao serviço usada pelo Application Auto Scaling, você deve desativar o escalonamento automático em todas as tabelas da conta em todas as Regiões da AWS.

## Configurando manualmente a capacidade provisionada de uma tabela multirregional (CQL)

Se você precisar desativar o escalonamento automático de uma tabela multirregional, poderá usá-lo para `ALTER TABLE` provisionar manualmente a capacidade de leitura da tabela para uma tabela de réplica.

```
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**
>
> Recomendamos usar o escalonamento automático para tabelas multirregionais que usam capacidade provisionada. Para ter mais informações, consulte the section called “Tabelas multirregionais”.

## Usando o AWS CLI para criar e gerenciar tabelas multirregionais

Você pode usar o AWS Command Line Interface (AWS CLI) para criar e gerenciar tabelas e espaços de chave multirregionais no Amazon Keyspaces.

Esta seção fornece exemplos de como criar e gerenciar tabelas multirregionais com o. AWS CLI Todas as tabelas que você cria em um espaço de chave multirregional herdam automaticamente as configurações de várias regiões do espaço de chave.

Para obter mais informações sobre os AWS CLI comandos do Amazon Keyspaces descritos neste tópico, consulte a Referência de AWS CLI comandos do Amazon Keyspaces.

Tópicos


- Como criar um novo espaço de chaves multirregional (CLI)
- Criação de uma nova tabela multirregional com configurações padrão (CLI)
- Criação de uma nova tabela multirregional no modo provisionado com escalabilidade automática (CLI)
- Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CLI)
- Visualizando a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional (CLI)
- Desativando o escalonamento automático para uma tabela multirregional (CLI)
- Configurando manualmente a capacidade provisionada de uma tabela multirregional (CLI)

## Como criar um novo espaço de chaves multirregional (CLI)

Para criar um espaço de chaves multirregional, você pode usar a instrução CLI a seguir. Especifique sua Região atual e pelo menos uma Região adicional no `regionList`.

```
aws keyspaces create-keyspace --keyspace-name mykeyspace
             \ --replication-specification
 replicationStrategy=MULTI_REGION,regionList=us-east-1,eu-west-1
```

> **Note**
>
> Ao criar um espaço de chaves multirregional, o Amazon Keyspaces cria uma função vinculada ao serviço com o nome `AWSServiceRoleForAmazonKeyspacesReplication` em sua conta. Essa função permite que o Amazon Keyspaces replique gravações em todas as réplicas de uma tabela multirregional em seu nome. Para saber mais, consulte the section called “Replicação multirregional”.

## Criação de uma nova tabela multirregional com configurações padrão (CLI)

Para criar uma tabela multirregional com configurações padrão, você só precisa especificar o esquema. Você pode usar o exemplo a seguir.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
           \ --schema-definition 'allColumns=[{name=pk,type=int}],partitionKeys={name=
 pk}'
```

A saída do comando é:

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable"
}
```

Para confirmar as configurações da tabela, você pode usar a seguinte declaração.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

A saída mostra todas as configurações padrão de uma tabela multirregional.

```
{
    "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
            }
        }
    ]
}
```

## Criação de uma nova tabela multirregional no modo provisionado com escalabilidade automática (CLI)

Para criar uma tabela multirregional no modo provisionado com configuração de escalonamento automático, você pode usar o. AWS CLI Observe que você deve usar o `create-table` comando CLI do Amazon Keyspaces para definir configurações de auto escalabilidade multirregional. Isso ocorre porque o Application Auto Scaling, o serviço que o Amazon Keyspaces usa para realizar o escalonamento automático em seu nome, não oferece suporte a várias regiões.

Para obter mais informações sobre as configurações de escalonamento automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consultethe section called “Crie uma nova tabela com escala automática usando o AWS CLI”.

Ao criar uma nova tabela multirregional no modo provisionado com configurações de escalonamento automático, você pode especificar as configurações gerais da tabela que são válidas para tudo em Regiões da AWS que a tabela é replicada. Em seguida, você pode sobrescrever as configurações de capacidade de leitura e ler as configurações de escalonamento automático para cada réplica. A capacidade de gravação, no entanto, permanece sincronizada entre todas as réplicas para garantir que haja capacidade suficiente para replicar gravações em todas as regiões.

Para definir a capacidade de leitura de uma réplica de tabela em uma região específica, você pode configurar os seguintes parâmetros como parte da `replicaSpecifications` tabela:

- A região

- As unidades de capacidade de leitura provisionadas (opcional)
- Configurações de escalonamento automático para capacidade de leitura (opcional)

Ao criar tabelas multirregionais provisionadas com configurações complexas de escalonamento automático e configurações diferentes para réplicas de tabela, é útil carregar as configurações de escalonamento automático e de réplica da tabela a partir de arquivos JSON.

Para usar o exemplo de código a seguir, você pode baixar os arquivos JSON de exemplo do auto-scaling.zip `auto-scaling.json` e extrair e. `replication.json` Anote o caminho para os arquivos.

Neste exemplo, os arquivos JSON estão localizados no diretório atual. Para diferentes opções de caminho de arquivo, consulte Como carregar parâmetros de um arquivo.

```
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
```

## Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CLI)

Para atualizar o modo provisionado e a configuração de escalonamento automático de uma tabela existente, você pode usar o comando. AWS CLI `update-table`

Observe que você deve usar os comandos CLI do Amazon Keyspaces para criar ou modificar configurações de auto escalabilidade multirregional. Isso ocorre porque o Application Auto Scaling, o serviço que o Amazon Keyspaces usa para realizar o escalonamento automático da capacidade da tabela em seu nome, não oferece suporte a várias. Regiões da AWS

Ao atualizar o modo provisionado ou as configurações de escalonamento automático de uma tabela multirregional, você pode atualizar as configurações de capacidade de leitura e a configuração de escalabilidade automática de leitura para cada réplica da tabela.

A capacidade de gravação, no entanto, permanece sincronizada entre todas as réplicas para garantir que haja capacidade suficiente para replicar gravações em todas as regiões. Para atualizar a

capacidade de leitura de uma réplica de tabela em uma região específica, você pode alterar um dos seguintes parâmetros opcionais da `replicaSpecifications` tabela:

- As unidades de capacidade de leitura provisionadas (opcional)
- Configurações de escalonamento automático para capacidade de leitura (opcional)

Ao atualizar tabelas multirregionais com configurações complexas de escalonamento automático e configurações diferentes para réplicas de tabela, é útil carregar as configurações de escalonamento automático e as configurações de réplica da tabela a partir de arquivos JSON.

Para usar o exemplo de código a seguir, você pode baixar os arquivos JSON de exemplo do auto-scaling.zip `auto-scaling.json` e extrair e. `replication.json` Anote o caminho para os arquivos.

Neste exemplo, os arquivos JSON estão localizados no diretório atual. Para diferentes opções de caminho de arquivo, consulte Como carregar parâmetros de um arquivo.

```
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
```

## Visualizando a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional (CLI)

Para visualizar a configuração de escalonamento automático de uma tabela multirregional, você pode usar a `get-table-auto-scaling-settings` operação. O comando CLI a seguir é um exemplo disso.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

Você verá a saída a seguir.

```
{
    "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
                        }
                    }
```

```
                }
            }
        }
    ]
}
```

## Desativando o escalonamento automático para uma tabela multirregional (CLI)

Você pode usar o AWS CLI `update-table` comando para desativar o escalonamento automático de uma tabela existente. Observe que você não pode desativar o escalonamento automático para uma réplica de tabela individual.

No exemplo a seguir, o escalonamento automático está desativado para a capacidade de leitura da tabela.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
          \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Para excluir a função vinculada ao serviço usada pelo Application Auto Scaling, você deve desativar o escalonamento automático em todas as tabelas da conta. Regiões da AWS

## Configurando manualmente a capacidade provisionada de uma tabela multirregional (CLI)

Se você precisar desativar o escalonamento automático de uma tabela multirregional, poderá usá-lo para `update-table` provisionar manualmente a capacidade de leitura da tabela para uma tabela de réplica.

```
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**
>
> Recomendamos usar o escalonamento automático para tabelas multirregionais que usam capacidade provisionada. Para ter mais informações, consulte the section called “Tabelas multirregionais”.

# Recuperação em um ponto no tempo para o Amazon Keyspaces (para Apache Cassandra)

A recuperação para um ponto no tempo (PITR) ajuda a proteger as tabelas do Amazon Keyspaces contra operações acidentais de gravação ou exclusão ao fornecer backups contínuos dos dados de tabelas.

Por exemplo, suponhamos que um script de teste seja gravado acidentalmente em uma tabela de produção do Amazon Keyspaces. Com a recuperação para um ponto no tempo, você pode recuperar os dados da tabela para qualquer segundo, já que a PITR foi habilitada nos últimos 35 dias. Se você excluir uma tabela com a recuperação para um ponto no tempo habilitada, poderá consultar os dados da tabela excluída por 35 dias (sem custo adicional) e restaurá-los para o estado em que estavam antes do ponto de exclusão.

Você pode restaurar uma tabela do Amazon Keyspaces em um ponto no tempo usando o console, o SDK da AWS e o AWS Command Line Interface (AWS CLI) ou o Cassandra Query Language (CQL). Para obter mais informações, consulte Como restaurar uma tabela do Amazon Keyspaces para um ponto no tempo.

As operações de um ponto no tempo não afetam o desempenho ou a disponibilidade na tabela base, e a restauração de uma tabela não consome throughput adicional.

Para obter informações sobre cotas PITR, consulte Cotas.

Para obter informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como point-in-time a recuperação funciona no Amazon Keyspaces
- Como restaurar uma tabela do Amazon Keyspaces para um ponto no tempo


## Como point-in-time a recuperação funciona no Amazon Keyspaces

Esta seção fornece uma visão geral de como a point-in-time recuperação do Amazon Keyspaces (PITR) funciona. Para obter mais informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

Tópicos

- Habilitando point-in-time a recuperação (PITR)
- Permissões necessárias para restaurar uma tabela
- Janela de tempo para backups contínuos da PITR
- Configurações de restauração de PITR
- Restauração PITR de tabelas criptografadas
- Restauração PITR de tabelas multirregionais
- Tempo de restauração da tabela com PITR
- PITR do Amazon Keyspaces e integração com serviços da AWS


# Habilitando point-in-time a recuperação (PITR)

Você pode habilitar a PITR usando o console ou de forma programática.

## Como habilitar a PITR com o console

As configurações da PITR para novas tabelas podem ser gerenciadas na opção Configurações personalizadas. Por padrão, a PITR está habilitada em novas tabelas criadas por meio do console.

Para habilitar a PITR para uma tabela existente, conclua as etapas a seguir.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tabelas e selecione a tabela que você deseja editar.
3. Na guia Backups, selecione Editar.
4. Na seção Editar configurações de point-in-time recuperação, selecione Ativar oint-in-time recuperação P.

Você pode desabilitar a PITR em uma tabela a qualquer momento com as etapas a seguir.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tabelas e selecione a tabela que você deseja editar.
3. Na guia Backups, selecione Editar.
4. Na seção Editar configurações point-in-time de recuperação, desmarque a caixa de seleção Ativar oint-in-time recuperação P.

> **Important**
>
> Desabilitar a PITR exclui seu histórico de backup imediatamente, mesmo se você reabilitar a PITR na tabela em 35 dias.

Para saber como restaurar uma tabela usando o console, consulte the section called “Como restaurar uma tabela para um ponto no tempo (console)”.

### Como habilitar PITR usando o AWS CLI

Você pode gerenciar as configurações de PITR para tabelas usando o API `UpdateTable`.

Ao criar uma nova tabela usando o AWS CLI, você deve habilitar explicitamente a PITR ao criar a nova tabela.

Para ativar a PITR ao criar uma nova tabela, você pode usar o comando AWS CLI a seguir como exemplo. O comando foi dividido em linhas separadas para facilitar a leitura.

```
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**
>
> Se nenhum valor point-in-time de recuperação for especificado, a point-in-time recuperação será desativada por padrão.

Para confirmar a configuração point-in-time de recuperação de uma tabela, você pode usar o AWS CLI comando a seguir.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Para habilitar esse recurso para uma tabela existente usando o AWS CLI, execute o comando a seguir.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=ENABLED'
```

Para desabilitar a PITR em uma tabela existente, execute o comando AWS CLI a seguir.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=DISABLED'
```

> **Important**
>
> Desabilitar a PITR exclui seu histórico de backup imediatamente, mesmo se você reabilitar a PITR na tabela em 35 dias.

## Como habilitar a PITR usando CQL

Você pode gerenciar as configurações de PITR para tabelas usando a propriedade customizada `point_in_time_recovery`.

Ao criar uma nova tabela usando CQL, você deve habilitar explicitamente a PITR ao criar a nova tabela.

Para ativar a PITR ao criar uma nova tabela, você pode usar o comando CQL a seguir como exemplo.

```
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**
>
> Se nenhuma propriedade personalizada de point-in-time recuperação for especificada, a point-in-time recuperação será desativada por padrão.

Para habilitar a PITR para uma tabela existente usando CQL, execute o comando CQL a seguir.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'enabled'}}
```

Para desabilitar a PITR em uma tabela existente, execute o comando CQL a seguir.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

> **Important**
>
> Desabilitar a PITR exclui seu histórico de backup imediatamente, mesmo se você reabilitar a PITR na tabela em 35 dias.

Para obter mais informações na Referência de linguagem CQL, consulte the section called “CRIAR TABELA” e the section called “ALTER TABLE”. Para saber como restaurar uma tabela usando CQL, consulte the section called “Como restaurar uma tabela para um ponto no tempo com CQL”.

## Permissões necessárias para restaurar uma tabela

Para restaurar uma tabela com sucesso, o usuário ou perfil do IAM precisa das seguintes permissões mínimas:

- `cassandra:Restore` – A ação de restauração é necessária para que a tabela de destino seja restaurada.
- `cassandra:Select` – A ação de seleção é necessária para ler a tabela de origem.
- `cassandra:TagResource` – A ação da tag é opcional e necessária somente se a operação de restauração adicionar tags.

Veja a seguir um exemplo de política que concede permissões mínimas necessárias a um usuário para restaurar tabelas no espaço de chaves `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*"
         ]
      }
   ]
}
```

Permissões adicionais para restaurar uma tabela podem ser necessárias com base em outros atributos selecionados. Por exemplo, se a tabela de origem for criptografada em repouso com uma chave gerenciada pelo cliente, o Amazon Keyspaces deverá ter permissões para acessar a chave gerenciada pelo cliente da tabela de origem para restaurar a tabela com sucesso. Para ter mais informações, consulte the section called “PITR e tabelas criptografadas”.

Se você estiver usando políticas do IAM com chaves de condição para restringir o tráfego de entrada para fontes específicas, você deve garantir que o Amazon Keyspaces tenha permissão para realizar uma operação de restauração em nome da sua entidade principal. Você deve adicionar uma chave de condição `aws:ViaAWSService` à sua política do IAM se ela restringir o tráfego de entrada a qualquer uma das seguintes opções:

- Endpoints da VPC com `aws:SourceVpce`
- Intervalos de IP com `aws:SourceIp`
- VPCs com `aws:SourceVpc`

A chave de condição `aws:ViaAWSService` permite acesso quando qualquer serviço da AWS faz uma solicitação usando as credenciais da entidade principal. Para obter mais informações, consulte Elementos de política JSON do IAM: chave de condição no Guia do usuário do IAM.

Veja a seguir um exemplo de uma política que restringe o tráfego de origem a um endereço IP específico e permite que o Amazon Keyspaces restaure uma tabela em nome da entidade 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"
            }
         }
      }
   ]
}
```

Para ver um exemplo de política usando a chave de condição global `aws:ViaAWSService`, consulte the section called “Políticas de endpoint de VPC e recuperação do Amazon point-in-time Keyspaces (PITR)”.

## Janela de tempo para backups contínuos da PITR

O PITR do Amazon Keyspaces usa duas datas e horas para manter o período em que os backups restauráveis estão disponíveis para uma tabela.

- Hora de restauração mais antiga – Marca a hora do primeiro backup restaurável. O backup restaurável mais antigo remonta a 35 dias ou quando a PITR foi ativada, o que for mais recente. A janela máxima de backup de 35 dias não pode ser modificada.
- Hora atual – A data e a hora do backup restaurável mais recente é a hora atual. Se nenhum registro de data e hora for fornecido durante uma restauração, a hora atual será usada.

Quando a PITR está habilitada, você pode restaurar para qualquer ponto no tempo entre `EarliestRestorableDateTime` e `CurrentTime`. Você só pode restaurar os dados da tabela em uma hora em que a PITR estava habilitada.

Se você desabilitar a PITR e depois reabilitá-la novamente, redefinirá a hora de início do primeiro backup disponível para quando a PITR foi reabilitada. Isso significa que a desabilitação da PITR apaga seu histórico de backup.

> **Note**
>
> As operações de linguagem de definição de dados (DDL) em tabelas, como alterações de esquema, são executadas de forma assíncrona. Você só pode ver as operações concluídas nos dados da tabela restaurada, mas talvez veja ações adicionais na tabela de origem se elas estavam em andamento no momento da restauração. Para ver uma lista de instruções DDL, consulte the section called “Instruções DDL”.

Uma tabela não precisa estar ativa para ser restaurada. Você também pode restaurar tabelas excluídas se a PITR estiver habilitada na tabela excluída e a exclusão tiver ocorrido na janela de backup (ou nos últimos 35 dias).

> **Note**
>
> Se uma nova tabela for criada com o mesmo nome qualificado (por exemplo, mykeyspace.mytable) de uma tabela excluída anteriormente, a tabela excluída não poderá mais ser restaurada. Se você tentar fazer isso pelo console, um aviso será exibido.

## Configurações de restauração de PITR

Quando você restaura uma tabela usando a PITR, o Amazon Keyspaces restaura o esquema e dados da tabela de origem para o estado com base na data e hora selecionado (`day:hour:minute:second`) em uma nova tabela. A PITR não sobrescreve as tabelas existentes.

Além do esquema e dos dados da tabela, a PITR restaura as `custom_properties` da tabela de origem. Ao contrário dos dados da tabela, que são restaurados com base na data e hora selecionadas entre a hora de restauração mais antiga e a hora atual, as propriedades personalizadas são sempre restauradas com base nas configurações da tabela na hora atual.

As configurações da tabela restaurada correspondem às configurações da tabela de origem com a data e a hora de quando a restauração foi iniciada. Se você desejar sobrescrever essas configurações durante a restauração, poderá fazer isso usando `WITH custom_properties`. As propriedades personalizadas incluem as configurações a seguir.

- Modo de capacidade de leitura/gravação
- Configurações de capacidade de throughput provisionada
- Configurações de PITR

Se a tabela estiver no modo de capacidade provisionada com o escalonamento automático ativado, a operação de restauração também restaurará as configurações de escalabilidade automática da tabela. Você pode sobrescrevê-los usando o `autoscaling_settings` parâmetro na CQL ou `autoScalingSpecification` com a CLI. Para obter mais informações sobre as configurações de escalonamento automático, consultethe section called “Gerencie a capacidade de produção com escalabilidade automática”.

Quando você faz uma restauração completa da tabela, todas as configurações da tabela restaurada são originadas das configurações atuais da tabela de origem no momento da restauração.

Por exemplo, suponha que o throughput provisionado de uma tabela tenha sido reduzido recentemente para 50 unidades de capacidade de leitura e 50 unidades de capacidade de gravação. Em seguida, você restaura o estado da tabela para três semanas atrás. Neste momento, o throughput provisionado foi definido para 100 unidades de capacidade de leitura e 100 unidades de capacidade de gravação. Nesse caso, o Amazon Keyspaces restaura os dados da tabela para esse ponto no tempo, mas usa as configurações de throughput provisionadas atuais (50 unidades de capacidade de leitura e 50 unidades de capacidade de gravação).

As configurações a seguir não são restauradas e você deve configurá-las manualmente para a nova tabela.

- Políticas do AWS Identity and Access Management (IAM)
- CloudWatch Métricas e alarmes da Amazon
- Tags (podem ser adicionadas à instrução `RESTORE` do CQL usando `WITH TAGS`)

## Restauração PITR de tabelas criptografadas

Quando você restaura uma tabela usando a PITR, o Amazon Keyspaces restaura as configurações de criptografia da tabela de origem. Se a tabela foi criptografada com uma Chave pertencente à AWS (padrão), ela será restaurada automaticamente com a mesma configuração. Se a tabela que você deseja restaurar foi criptografada usando uma chave gerenciada pelo cliente, a mesma chave gerenciada pelo cliente precisa estar acessível ao Amazon Keyspaces para restaurar os dados da tabela.

Você pode alterar as configurações de criptografia da tabela no momento da restauração. Para mudar de uma chave Chave pertencente à AWS para uma chave gerenciada pelo cliente, você precisa fornecer uma chave gerenciada pelo cliente válida e acessível no momento da restauração.

Se você quiser mudar de uma chave gerenciada pelo cliente para uma Chave pertencente à AWS, confirme se o Amazon Keyspaces tem acesso à chave gerenciada pelo cliente da tabela de origem para restaurar a tabela com uma Chave pertencente à AWS. Para obter mais informações sobre configurações de criptografia em repouso para tabelas, consulte the section called “Como funciona”.

> **Note**
>
> Se a tabela foi excluída porque o Amazon Keyspaces perdeu o acesso à sua chave gerenciada pelo cliente, você precisa garantir que a chave gerenciada pelo cliente esteja acessível ao Amazon Keyspaces antes de tentar restaurá-la. Uma tabela que foi criptografada com uma chave gerenciada pelo cliente não pode ser restaurada se o Amazon Keyspaces não tiver acesso a essa chave. Para obter mais informações, consulte Solucionar problemas de acesso à chave, no AWS Key Management Service Guia do desenvolvedor.

## Restauração PITR de tabelas multirregionais

Você pode restaurar uma tabela multirregional usando a PITR. Para que a operação de restauração seja bem-sucedida, as tabelas de origem e de destino precisam ser replicadas na mesma Regiões da AWS.

O Amazon Keyspaces restaura as configurações da tabela de origem em cada uma das regiões replicadas que fazem parte do keyspace. Você também pode substituir as configurações durante a operação de restauração. Para obter mais informações sobre as configurações que podem ser alteradas durante a restauração, consulte the section called “Restaurar as configurações”.

Para informações sobre como replicar chaves multirregionais, consulte the section called “Como funciona”.

## Tempo de restauração da tabela com PITR

O tempo necessário para restaurar uma tabela é baseado em vários fatores e nem sempre está correlacionado diretamente com o tamanho da tabela.

Veja a seguir algumas considerações sobre os tempos de restauração.

- Os backups são restaurados para novas tabelas. Pode demorar até 20 minutos (mesmo se a tabela estiver vazia) para executar todas as ações necessárias para criar a nova tabela e iniciar o processo de restauração.
- Os tempos de restauração de tabelas grandes com modelos de dados bem distribuídos podem ser de várias horas ou mais.
- Se a tabela de origem tiver dados com distorção significativa, o tempo de restauração poderá aumentar. Por exemplo, se a chave primária de uma tabela estiver usando o mês do ano como chave de partição e todos os dados forem do mês de dezembro, os dados estarão distorcidos.

Uma prática recomendada ao planejar a recuperação de desastres é documentar regularmente os tempos médios de conclusão da restauração e estabelecer como esses tempos afetam seu objetivo geral de tempo de recuperação.

## PITR do Amazon Keyspaces e integração com serviços da AWS

As seguintes operações de PITR são registradas usando AWS CloudTrail para permitir monitoramento e auditoria contínuos.

- Crie uma nova tabela com a PITR habilitada ou desabilitada.
- Habilite ou desabilite a PITR em uma tabela existente.
- Restaure uma tabela ativa ou excluída.

Para ter mais informações, consulte Registro de chamadas de API do Amazon Keyspaces com AWS CloudTrail.

Você pode realizar as seguintes ações usando AWS CloudFormation.

- Crie uma nova tabela com a PITR habilitada ou desabilitada.

- Habilite ou desabilite a PITR em uma tabela existente.

Para obter mais informações, consulte Referência de tipos de recursos do Cassandra no AWS CloudFormation Guia do usuário.

# Como restaurar uma tabela do Amazon Keyspaces para um ponto no tempo

A recuperação para um ponto no tempo (PITR) do Amazon Keyspaces (para Apache Cassandra) permite restaurar dados da tabela do Amazon Keyspaces para qualquer ponto no tempo nos últimos 35 dias. A primeira parte deste tutorial mostra como restaurar uma tabela em um ponto no tempo usando o console Amazon Keyspaces, o AWS Command Line Interface (AWS CLI) e o Cassandra Query Language (CQL). A segunda parte mostra como restaurar uma tabela excluída usando o AWS CLI e o CQL.

Tópicos


- Antes de começar
- Como restaurar uma tabela para um ponto no tempo (console)
- Como restaurar uma tabela para um ponto no tempo com o AWS CLI
- Como restaurar uma tabela para um ponto no tempo com CQL
- Como restaurar uma tabela excluída com o AWS CLI
- Como restaurar uma tabela excluída com CQL


## Antes de começar

Se você ainda não fez isso, deverá configurar as permissões apropriadas para o usuário restaurar tabelas do Amazon Keyspaces. Em AWS Identity and Access Management (IAM), a política AWS gerenciada pela `AmazonKeyspacesFullAccess` inclui as permissões para restaurar tabelas do Amazon Keyspaces. Para obter etapas detalhadas para implementar uma política com as permissões mínimas necessárias, consulte the section called “Restaurar permissões”.

## Como restaurar uma tabela para um ponto no tempo (console)

O exemplo a seguir demonstra como usar o console do Amazon Keyspaces para restaurar uma tabela existente chamada `mytable` para um ponto no tempo.

> **Note**
>
> Esse procedimento supõe que você habilitou a recuperação em um ponto anterior no tempo. Para habilitar a PITR para a tabela `mytable`, siga as etapas em the section called “Como usar o console”.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, no lado esquerdo do console, selecione Tables (Tabelas).
3. Na lista de tabelas, escolha a tabela `mytable`.
4. Na guia Backups da tabela de `mytable`, na seção Recuperação para um ponto no tempo, selecione Restaurar.
5. Insira **`mytable_restored`** como o novo nome da tabela.
6. Para definir o ponto no tempo da operação de restauração, você pode escolher entre duas opções:
   - Selecione a hora Mais antiga pré-configurada.
   - Selecione Especificar data e hora e insira a data e a hora para as quais você deseja restaurar a nova tabela.

   > **Note**
   >
   > É possível fazer a restauração para qualquer ponto no tempo dentro da hora Mais antiga e da hora atual. O Amazon Keyspaces restaura os dados da tabela para o estado com base na data e na hora selecionadas (dia:hora:minuto:segundo).
7. Selecione Restaurar para iniciar o processo de restauração.

   A tabela que está sendo restaurada é mostrada com o status Restoring (Em restauração). Quando o processo de restauração for concluído, o status da tabela `mytable_restored` mudará para Active (Ativo).

   > **Important**
   >
   > Enquanto uma restauração estiver em andamento, não modifique nem exclua as políticas do AWS Identity and Access Management (IAM) que concedem à entidade do

IAM (por exemplo, usuário, grupo ou função) permissão para realizar a restauração. Do contrário, pode haver um comportamento inesperado. Por exemplo, suponha que você tenha removido as permissões de gravação para uma tabela enquanto essa tabela estava sendo restaurada. Nesse caso, a operação `RestoreTableToPointInTime` subjacente não conseguirá gravar na tabela nenhum dos dados restaurados. Você pode modificar ou excluir permissões somente depois que a operação de restauração é concluída.

## Como restaurar uma tabela para um ponto no tempo com o AWS CLI

O seguinte procedimento mostra como usar a AWS CLI para restaurar uma tabela existente chamada `myTable` para um ponto no tempo.

1. Na primeira etapa, você cria uma tabela simples chamada `myTable` que tem a PITR ativada. O comando foi dividido em linhas separadas para facilitar a leitura.

```
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. Confirme as propriedades da nova tabela e revise `earliestRestorableTimestamp` para a PITR.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

A saída deste comando retorna o seguinte.

```
{
    "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": ""
    }
}
```

Você pode restaurar uma tabela ativa em qualquer ponto no tempo entre `earliestRestorableTimestamp` e a hora atual em intervalos de um segundo. O padrão é a hora atual.

3. Para restaurar uma tabela para um ponto no tempo, especifique um `restore_timestamp` no formato ISO 8601. É possível escolher qualquer ponto no tempo durante os últimos 35

dias em intervalos de um segundo. Por exemplo, o comando a seguir restaura a tabela para o `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"
```

A saída desse comando retornará o ARN da tabela restaurada.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored"
}
```

Para restaurar a tabela para a hora atual, você pode omitir o `restore-timestamp`.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored1'"
```

**Important**

Enquanto uma restauração estiver em andamento, não modifique nem exclua as políticas do AWS Identity and Access Management (IAM) que concedem à entidade do IAM (por exemplo, usuário, grupo ou função) permissão para realizar a restauração. Do contrário, pode haver um comportamento inesperado. Por exemplo, suponha que você tenha removido as permissões de gravação para uma tabela enquanto essa tabela estava sendo restaurada. Nesse caso, a operação `RestoreTableToPointInTime` subjacente não conseguirá gravar na tabela nenhum dos dados restaurados.
Você pode modificar ou excluir permissões somente depois que a operação de restauração é concluída.

## Como restaurar uma tabela para um ponto no tempo com CQL

O procedimento a seguir mostra como usar CQL para restaurar uma tabela existente chamada `mytable` para um ponto no tempo.

> **Note**
>
> Esse procedimento supõe que você habilitou a recuperação em um ponto anterior no tempo. Para habilitar a PITR na tabela , siga as etapas em the section called “CQL”.

1. Você pode restaurar uma tabela ativa para um ponto no tempo entre `earliest_restorable_timestamp` e a hora atual. O padrão é a hora atual.

   Para confirmar se a recuperação de um ponto no tempo está habilitada para a tabela `mytable`, consulte o seguinte `system_schema_mcs.tables`.

   ```
   SELECT custom_properties
   FROM system_schema_mcs.tables
   WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
   ```

   A recuperação para um ponto no tempo está habilitada conforme mostrado no exemplo de saída a seguir.

   ```
   custom_properties
   -----------------
   {
     ...,
       "point_in_time_recovery": {
       "earliest_restorable_timestamp":"2020-06-30T19:19:21.175Z"
       "status":"enabled"
     }
   }
   ```

2. Restaure a tabela para um ponto no tempo, especificada por um `restore_timestamp` no formato ISO 8601. Nesse caso, a tabela `mytable` é restaurada para a hora atual. Você pode omitir a cláusula `WITH restore_timestamp = ...`. Sem a cláusula, a data e a hora atuais são usadas.

   ```
   RESTORE TABLE mykeyspace.mytable_restored
   FROM TABLE mykeyspace.mytable;
   ```

Você também restaurar para um ponto específico. Você pode especificar qualquer ponto durante os últimos 35 dias. Por exemplo, o comando a seguir restaura a tabela para o `EarliestRestorableDateTime`.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable
WITH restore_timestamp = '2020-06-30T19:19:21.175Z';
```

Para obter uma descrição completa da sintaxe, consulte the section called “RESTAURAR TABELA” na referência de linguagem.

Para verificar se a restauração da tabela foi bem-sucedida, consulte o `system_schema_mcs.tables` para confirmar o status da tabela.

```
SELECT status
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable_restored'
```

A consulta mostra o a saída a seguir.

```
status
------
RESTORING
```

A tabela que está sendo restaurada é mostrada com o status Restoring (Em restauração). Quando o processo de restauração for concluído, o status da tabela `mytable_restored` mudará para Active (Ativo).

> **Important**
>
> Enquanto uma restauração estiver em andamento, não modifique nem exclua as políticas do AWS Identity and Access Management (IAM) que concedem à entidade do IAM (por exemplo, usuário, grupo ou função) permissão para realizar a restauração. Do contrário, pode haver um comportamento inesperado. Por exemplo, suponha que você tenha removido as permissões de gravação para uma tabela enquanto essa tabela estava sendo restaurada. Nesse caso, a operação `RestoreTableToPointInTime` subjacente não conseguirá gravar na tabela nenhum dos dados restaurados.

Você pode modificar ou excluir permissões somente depois que a operação de restauração é concluída.

## Como restaurar uma tabela excluída com o AWS CLI

O procedimento a seguir mostra como usar o AWS CLI para restaurar uma tabela excluída chamada `myTable` para o momento da exclusão.

> **Note**
>
> Esse procedimento pressupõe que a PITR foi ativada na tabela excluída.

1. Exclua a tabela criada no tutorial anterior.

```
aws keyspaces delete-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

2. Restaure a tabela excluída até o momento da exclusão com o comando a seguir.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored2'
```

A saída desse comando retornará o ARN da tabela restaurada.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored2"
}
```

## Como restaurar uma tabela excluída com CQL

O procedimento a seguir mostra como usar o CQL para restaurar uma tabela excluída chamada `mytable` para o momento da exclusão.

> **Note**
>
> Esse procedimento pressupõe que a PITR foi ativada na tabela excluída.

1. Para confirmar se a recuperação em um ponto no tempo está habilitada para uma tabela excluída, consulte a tabela do sistema. Somente tabelas com a recuperação em um ponto no tempo habilitada são exibidas.

   ```
   SELECT custom_properties
   FROM system_schema_mcs.tables_history
   WHERE keyspace_name = 'mykeyspace' AND table_name = 'my_table';
   ```

   A consulta mostra o a saída a seguir.

   ```
   custom_properties
   ------------------
   {
       ...,
      "point_in_time_recovery":{
         "restorable_until_time":"2020-08-04T00:48:58.381Z",
         "status":"enabled"
      }
   }
   ```

2. Restaure a tabela até o momento da exclusão com o exemplo de instrução a seguir.

   ```
   RESTORE TABLE mykeyspace.mytable_restored
   FROM TABLE mykeyspace.mytable;
   ```

# Dados expirados usando a vida útil (TTL) do Amazon Keyspaces

A vida útil (TTL) do Amazon Keyspaces (para Apache Cassandra) o ajuda a simplificar a lógica do aplicativo e a otimizar o preço do armazenamento, expirando os dados das tabelas automaticamente. Os dados que você não precisa mais são excluídos automaticamente da sua tabela com base no valor de Vida útil que você definiu. Isso facilita a conformidade com as políticas de retenção de dados com base em requisitos comerciais, setoriais ou regulamentares que definem por quanto tempo os dados precisam ser retidos ou especificam quando os dados devem ser excluídos.

Por exemplo, você pode usar a TTL em um aplicativo da AdTech para programar quando os dados de anúncios específicos expiram e não ficam mais visíveis para os clientes. Você também pode usar a TTL para retirar dados antigos automaticamente e economizar nos custos de armazenamento. Você pode definir um valor de TTL padrão para toda a tabela e substituir esse valor para linhas e colunas individuais. As operações de TTL não afetam o desempenho do seu aplicativo. Além disso, o número de linhas e colunas marcadas para expirar com a TTL não afeta a disponibilidade da tabela.

O Amazon Keyspaces filtra automaticamente os dados expirados para que não sejam retornados nos resultados da consulta ou estejam disponíveis para uso em instruções de linguagem de manipulação de dados (DML). O Amazon Keyspaces normalmente exclui dados expirados do armazenamento dentro de 10 dias da data de expiração. Em casos raros, o Amazon Keyspaces pode não conseguir excluir dados dentro de 10 dias se houver atividade sustentada na partição de armazenamento subjacente para proteger a disponibilidade. Nesses casos, o Amazon Keyspaces continua tentando excluir os dados expirados quando o tráfego na partição diminui. Depois que os dados são excluídos permanentemente do armazenamento, você deixa de incorrer em taxas de armazenamento. Para obter mais informações, consulte the section called “Como funciona”.

Você pode definir, modificar ou desativar as configurações de TTL padrão para tabelas novas e existentes usando o console ou o Cassandra Query Language (CQL). Em tabelas com TTL padrão configurado, você pode usar o Cassandra Query Language (CQL) para substituir as configurações de TTL padrão e aplicar valores de TTL personalizados a linhas e colunas. Para obter mais informações, consulte the section called “Como usar a vida útil”.

O preço da TTL é baseado no tamanho das linhas que estão sendo excluídas ou atualizadas usando a Vida útil. As operações de TTL são medidas em unidades de `TTL deletes`. Uma exclusão de TTL é consumida por KB de dados por linha que é excluída ou atualizada. Por exemplo, para atualizar uma linha que armazena 2,5 KB de dados e excluir uma ou mais colunas dentro da linha ao

mesmo tempo, são necessárias três exclusões de TTL. Ou, para excluir uma linha inteira que contém 3,5 KB de dados, são necessárias quatro exclusões de TTL. Uma exclusão de TTL é consumida por KB de dados excluídos por linha. Para obter mais informações sobre preços, consulte o Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como funciona: vida útil (TTL) do Amazon Keyspaces
- Como usar a vida útil (TTL)


# Como funciona: vida útil (TTL) do Amazon Keyspaces

A vida útil (TTL) do Amazon Keyspaces é totalmente gerenciada. Você não precisa gerenciar configurações de sistema de baixo nível, como estratégias de compactação. Os dados expiram no momento especificado, e o Amazon Keyspaces os remove automaticamente (normalmente em 10 dias), sem afetar o desempenho ou a disponibilidade do seu aplicativo.

Os dados expirados estão marcados para exclusão e não estão disponíveis para instruções de linguagem de manipulação de dados (DML). À medida que você continua realizando leituras e gravações em linhas que contêm dados expirados, eles continuam contabilizando unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs) até serem excluídos do armazenamento.

Tópicos


- Como definir o valor de TTL padrão para uma tabela
- Como configurando valores de TTL personalizados para linhas e colunas
- Como habilitar a TTL em tabelas
- Vida útil do Amazon Keyspaces e integração com serviços da AWS


## Como definir o valor de TTL padrão para uma tabela

No Amazon Keyspaces, você pode definir um valor de TTL padrão para todas as linhas em uma tabela quando a tabela é criada. Você também pode editar uma tabela existente para definir ou alterar o valor de TTL padrão para novas linhas inseridas na tabela. Alterar o valor de TTL padrão de uma tabela não modifica o valor TTL de nenhum dado existente nela. O valor de TTL padrão para uma tabela é zero, o que significa que os dados não expiram automaticamente. Se o valor de TTL

padrão de uma tabela for maior que zero, registros de data e hora de expiração serão adicionados a cada linha.

O Amazon Keyspaces calcula novo registro de data e hora de TTL sempre que os dados são atualizados. Os valores de TTL são definidos em segundos e o valor máximo configurável é 630.720.000 segundos, o que equivale a 20 anos. Para obter mais informações sobre como definir, modificar e desativar o valor de TTL padrão para tabelas usando o AWS Management Console ou CQL, consulte the section called “Como usar a vida útil”.

## Como configurando valores de TTL personalizados para linhas e colunas

> **Note**
>
> Antes de definir valores de TTL personalizados para linhas e colunas, a TTL deve ser ativada primeiro na tabela. Para obter mais informações, consulte the section called “Como habilitar a vida útil (TTL) em tabelas existentes usando propriedades personalizadas”.

Para substituir o valor de TTL padrão de uma tabela ou definir datas de expiração para linhas individuais, você pode usar as seguintes instruções de linguagem de manipulação de dados (DML) CQL:

- `INSERT` – Use para inserir uma nova linha de dados com um conjunto de valores de TTL.
- `UPDATE` – Use para modificar uma linha de dados existente com um novo valor de TTL.

A configuração de valores de TTL para linhas tem precedência sobre a configuração de TTL padrão da tabela.

Para obter exemplos e sintaxe de CQL, consulte the section called “Para usar `INSERT` para editar as configurações de tempo de vida (TTL) personalizadas usando CQL”.

Para substituir ou definir valores de TTL para colunas individuais, você pode atualizar a configuração de TTL para um subconjunto de colunas nas linhas existentes usando a seguinte instrução DML do CQL:

- `UPDATE` – Use para atualizar uma coluna de dados.

A configuração de valores de TTL para colunas tem precedência sobre a configuração de TTL padrão da tabela e qualquer configuração de TTL personalizada para a linha. Para obter exemplos e sintaxe de CQL, consulte the section called “Para usar UPDATE para editar as configurações de tempo de vida (TTL) personalizadas usando CQL”.

## Como habilitar a TTL em tabelas

A TTL é habilitada automaticamente para tabelas quando você especifica um valor `default_time_to_live` maior que 0 em uma das instruções `CREATE TABLE` ou `ALTER TABLE`. Se você não especificar um `default_time_to_live` para a tabela, mas quiser especificar valores de TTL personalizados para linhas ou colunas usando operações `INSERT` ou `UPDATE`, primeiro habilite a TTL para a tabela. Você pode habilitar a TTL para uma tabela usando a propriedade personalizada `ttl`.

Quando você habilita a TTL em uma tabela, o Amazon Keyspaces começa a armazenar metadados adicionais relacionados à TTL para cada linha. Além disso, a TTL usa registros de data e hora de expiração para rastrear quando as linhas ou colunas expiram. Os registros de data e hora são armazenados como metadados da linha e contribuem para o custo de armazenamento da linha.

Depois que o atributo TTL estiver habilitado, você não poderá desabilitá-lo para uma tabela. Definir o `default_time_to_live` da tabela como 0 desabilita os prazos de expiração padrão para novos dados, mas não desabilita o atributo TTL nem reverte a tabela para os metadados de armazenamento ou comportamento de gravação originais do Amazon Keyspaces.

## Vida útil do Amazon Keyspaces e integração com serviços da AWS

A métrica de TTL a seguir está disponível no Amazon CloudWatch para permitir o monitoramento contínuo.

- `TTLDeletes` – As unidades consumidas para excluir ou atualizar dados em uma linha usando a vida útil (TTL).

Para obter mais informações sobre como monitorar as métricas do CloudWatch, consulte the section called “Monitoramento com CloudWatch”.

Ao usarAWS CloudFormation, você pode habilitar a TTL ao criar uma tabela do Amazon Keyspaces. Para obter mais informações, consulte o Guia do usuário da AWS CloudFormation.

# Como usar a vida útil (TTL)

É possível usar o console do Amazon Keyspaces (para Apache Cassandra) ou o CQL para habilitar, atualizar e desabilitar as configurações de vida útil.

Tópicos


- Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas (console)
- Para atualizar as configurações de vida útil (TTL) padrão em tabelas existentes (console)
- Para desabilitar as configurações de vida útil (TTL) padrão em tabelas existentes (console)
- Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas usando CQL
- Para usar ALTER TABLE para editar as configurações de tempo de vida (TTL) padrão usando CQL
- Como habilitar a vida útil (TTL) em novas tabelas usando propriedades personalizadas
- Como habilitar a vida útil (TTL) em tabelas existentes usando propriedades personalizadas
- Para usar INSERT para editar as configurações de tempo de vida (TTL) personalizadas usando CQL
- Para usar UPDATE para editar as configurações de tempo de vida (TTL) personalizadas usando CQL


## Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas (console)

Siga estas etapas para criar uma nova tabela com as configurações de vida útil habilitadas usando o console do Amazon Keyspaces.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas) e Create table (Criar tabela).
3. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a nova tabela.
4. Na seção Esquema, crie o esquema para sua tabela.
5. Na seção Configurações da tabela, selecione Personalizar configurações.

6. Continue para vida útil (TTL).

   Nesta etapa, você seleciona as configurações de TTL padrão para a tabela.

   Para o Período de TTL padrão, insira o tempo de expiração e escolha a unidade de tempo inserida, por exemplo, segundos, dias ou anos. O Amazon Keyspaces armazenará o valor em segundos.
7. Escolha Create table. Sua tabela é criada com o valor de TTL padrão especificado.

> **Note**
>
> Você pode substituir a configuração de TTL padrão da tabela para linhas ou colunas específicas usando a linguagem de manipulação de dados (DML) no editor CQL.

## Para atualizar as configurações de vida útil (TTL) padrão em tabelas existentes (console)

Siga estas etapas para atualizar as configurações de vida útil para tabelas existentes usando o console do Amazon Keyspaces.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela que deseja atualizar e selecione a guia Configurações adicionais.
3. Continue até Vida útil (TTL) e selecione Editar.
4. Para o Período de TTL padrão, insira o tempo de expiração e escolha a unidade de tempo inserida, por exemplo, segundos, dias ou anos. O Amazon Keyspaces armazenará o valor em segundos. Isso não altera o valor de TTL das linhas existentes.
5. Quando as configurações de TTL estiverem definidas, selecione Salvar alterações.

## Para desabilitar as configurações de vida útil (TTL) padrão em tabelas existentes (console)

Siga estas etapas para desabilitar as configurações de vida útil para tabelas existentes usando o AWS Management Console do Amazon Keyspaces.

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela que deseja atualizar e selecione a guia Configurações adicionais.
3. Continue até Vida útil (TTL) e selecione Editar.
4. Selecione Período de TTL padrão e defina o valor como zero. Isso desabilita a TTL para a tabela por padrão para dados futuros. Isso não altera o valor de TTL para as linhas existentes.
5. Quando as configurações de TTL estiverem definidas, selecione Salvar alterações.

## Para criar uma nova tabela com as configurações de vida útil (TTL) padrão habilitadas usando CQL

Habilite a TTL ao criar uma nova tabela com o valor de TTL padrão definido como 3.024.000 segundos, o que representa 35 dias.

```
CREATE TABLE my_table (
                userid uuid,
                time timeuuid,
                subject text,
                body text,
                user inet,
                PRIMARY KEY (userid, time)
                ) WITH default_time_to_live = 3024000;
```

Para confirmar as configurações de TTL para a nova tabela, use a instrução `cqlsh describe` conforme mostrado no exemplo a seguir. A saída mostra a configuração de TTL padrão para a tabela como `default_time_to_live`.

```
describe my_table;
```

## Para usar **`ALTER TABLE`** para editar as configurações de tempo de vida (TTL) padrão usando CQL

Atualize as configurações de TTL da tabela existente para 2.592.000 segundos, o que representa 30 dias.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Para confirmar as configurações de TTL para a tabela atualizada, use a instrução `cqlsh describe` conforme mostrado no exemplo a seguir. A saída mostra a configuração de TTL padrão para a tabela como `default_time_to_live`.

```
describe my_table;
```

## Como habilitar a vida útil (TTL) em novas tabelas usando propriedades personalizadas

Para habilitar as configurações de Vida útil personalizadas que podem ser aplicadas a linhas e colunas sem ativar as configurações padrão de TTL para toda a tabela, você pode usar a seguinte instrução CQL.

```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH CUSTOM_PROPERTIES={'ttl':
{'status': 'enabled'}};
```

Depois de habilitado o `ttl`, você não poderá desabilitá-lo para a tabela.

## Como habilitar a vida útil (TTL) em tabelas existentes usando propriedades personalizadas

Para habilitar as configurações de Vida útil personalizadas que podem ser aplicadas a linhas e colunas sem ativar as configurações padrão de TTL para toda a tabela, você pode usar a seguinte instrução CQL.

```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

Depois de habilitado o `ttl`, você não poderá desabilitá-lo para a tabela.

## Para usar **`INSERT`** para editar as configurações de tempo de vida (TTL) personalizadas usando CQL

A instrução CQL a seguir insere uma linha de dados na tabela e altera a configuração de TTL padrão para 259.200 segundos (o que equivale a 3 dias).

```
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;
```

Para confirmar as configurações de TTL para a linha inserida, use a instrução a seguir.

```
SELECT TTL (subject) from my_table;
```

## Para usar **UPDATE** para editar as configurações de tempo de vida (TTL) personalizadas usando CQL

Para alterar as configurações de TTL da coluna “assunto” inserida anteriormente de 259.200 segundos (3 dias) para 86.400 segundos (um dia), use a seguinte instrução.

```
UPDATE my_table USING TTL 86400 set subject = 'Updated Message' WHERE userid =
 B79CB3BA-745E-5D9A-8903-4A02327A7E09 and time = 96a29100-5e25-11ec-90d7-b5d91eceda0a;
```

Você pode executar uma consulta de seleção simples para ver o registro atualizado antes do prazo de expiração.

```
SELECT * from my_table;
```

A consulta mostra o a saída a seguir.

```
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
```

Para confirmar que a expiração foi bem-sucedida, execute a mesma consulta novamente após o tempo de expiração configurado.

```
SELECT * from my_table;
```

A consulta mostra a seguinte saída após a expiração da coluna “assunto”.

```
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
```

# Como trabalhar com carimbos de data/hora do lado do cliente no Amazon Keyspaces

No Amazon Keyspaces, os carimbos de data/hora do lado do cliente são compatíveis com o Cassandra que persistem para cada célula da sua tabela. Você pode usar carimbos de data/hora do lado do cliente para resolução de conflitos, permitindo que seus aplicativos cliente determinem a ordem das gravações. Por exemplo, quando clientes de um aplicativo distribuído globalmente fazem atualizações nos mesmos dados, os carimbos de data/hora do lado do cliente persistem na ordem em que as atualizações foram feitas nos clientes. O Amazon Keyspaces usa esses carimbos de data/hora para processar as gravações. Para obter mais informações, consulte the section called “Como funciona”.

Depois que os carimbos de data/hora do lado do cliente forem ativados para uma tabela, você poderá especificar um carimbo de data/hora com a cláusula `USING TIMESTAMP` em sua consulta CQL da Data Manipulation Language (DML). Se você não especificar um carimbo de data/hora na sua consulta CQL, o Amazon Keyspaces usará o carimbo de data/hora passado pelo seu driver de cliente. Se o driver do cliente não fornecer carimbos de data/hora, o Amazon Keyspaces atribuirá automaticamente um carimbo no nível da célula. Para consultar carimbos de data/hora, você pode usar a função `WRITETIME` em sua instrução DML. Para obter mais informações, consulte the section called “Como usar carimbos de data/hora no lado do cliente”.

O Amazon Keyspaces não cobra nada a mais para ativar os carimbos de data/hora do lado do cliente. No entanto, com os carimbos de data/hora do lado do cliente, você armazena e grava dados adicionais para cada valor em sua linha. Isso pode levar ao uso adicional do armazenamento e, em alguns casos, ao uso adicional do throughput. Para saber mais sobre como estimar o impacto no tamanho da linha, consulte the section called “Como funciona”. Para obter mais informações sobre os preços do serviço Amazon Keyspaces, consulte os preços do serviço Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Como os carimbos de data/hora do lado do cliente funcionam no Amazon Keyspaces
- Uso de carimbos de data/hora no lado do cliente no Amazon Keyspaces

# Como os carimbos de data/hora do lado do cliente funcionam no Amazon Keyspaces

Os carimbos de data/hora do lado do cliente do Amazon Keyspaces são totalmente gerenciados. Você não precisa gerenciar configurações de sistema de baixo nível, como estratégias de limpeza e compactação.

Ao excluir dados, as linhas são marcadas para exclusão com uma lápide. O Amazon Keyspaces remove dados marcados para exclusão automaticamente (normalmente em 10 dias) sem afetar o desempenho ou a disponibilidade do seu aplicativo. Os dados marcados para exclusão não estão disponíveis para instruções de linguagem de manipulação de dados (DML). À medida que você continua realizando leituras e gravações em linhas que contêm dados marcados com a lápide, esses dados continuam sendo contabilizados para armazenamento, unidades de capacidade de leitura (RCUs) e unidades de capacidade de gravação (WCUs) até serem excluídos do armazenamento.

Tópicos


- Como os carimbos de data/hora do lado do cliente funcionam no Amazon Keyspaces
- Carimbos de data/hora do lado do cliente do Amazon Keyspaces e integração com serviços AWS


## Como os carimbos de data/hora do lado do cliente funcionam no Amazon Keyspaces

Quando os carimbos de data/hora do lado do cliente são ativados no Amazon Keyspaces, cada coluna de cada linha armazena um carimbo de data/hora. Esses carimbos de data/hora ocupam aproximadamente 20 a 40 bytes (dependendo dos seus dados) e contribuem para o custo de armazenamento e throughput da linha. Esses bytes de metadados também contam para sua cota de tamanho de linha de 1 MB. Para determinar o aumento geral no espaço de armazenamento (para garantir que o tamanho da linha permaneça abaixo de 1 MB), considere o número de colunas em sua tabela e o número de elementos de coleção em cada linha. Por exemplo, se uma tabela tiver 20 colunas, com cada coluna armazenando 40 bytes de dados, o tamanho da linha aumentará de 800 bytes para 1200 bytes. Para obter mais informações sobre como estimar o tamanho de uma linha, consulte the section called “Como calcular o tamanho da linha”. Além dos 400 bytes extras para armazenamento, neste exemplo, o número de unidades de capacidade de gravação (WCUs) consumidas por gravação aumenta de 1 WCU para 2 WCUs. Para obter mais informações sobre como calcular a capacidade de leitura e gravação, consulte the section called “Modos de capacidade de leitura/gravação”.

Depois que os carimbos de data/hora do lado do cliente forem ativados para uma tabela, você não poderá desativá-la. Além disso, os carimbos de data/hora não podem ser NULL, portanto, se nenhum carimbo de data/hora do lado do cliente for fornecido pela instrução CQL ou pelo driver do cliente, um carimbo de data/hora gerado pelo Amazon Keyspaces será adicionado automaticamente.

## Carimbos de data/hora do lado do cliente do Amazon Keyspaces e integração com serviços AWS

A seguinte métrica de carimbos de data/hora do lado do cliente está disponível no Amazon CloudWatch para permitir o monitoramento contínuo.

- `SystemReconciliationDeletes` - O número de operações de exclusão necessárias para remover dados marcados com lápides.

Para obter mais informações sobre como usar as métricas do CloudWatch, consulte the section called “Monitoramento com CloudWatch”.

# Uso de carimbos de data/hora no lado do cliente no Amazon Keyspaces

Você pode usar o console do Amazon Keyspaces (para Apache Cassandra), Cassandra Query Language (CQL), o SDK AWS e AWS Command Line Interface (AWS CLI) para ativar carimbos de data/hora do lado do cliente. Esta seção fornece exemplos de como ativar carimbos de data/hora do lado do cliente em tabelas novas e existentes e como usar carimbos de data/hora do lado do cliente em consultas. Para obter mais informações sobre a API, consulte Referência da API do Amazon Keyspaces.

> **Important**
>
> Os carimbos de data/hora do lado do cliente não podem ser desativados. Ativar os carimbos de data/hora do lado do cliente é uma alteração única. O Amazon Keyspaces não oferece a opção de desativá-lo sem excluir a tabela.

Tópicos

- Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (console)

- Como ativar os carimbos de data/hora do lado do cliente em tabelas existentes (console)
- Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CQL)
- Como ativar carimbos de data/hora do lado do cliente para tabelas existentes usando ALTER TABLE (CQL)
- Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CLI)
- Como ativar carimbos de data/hora do lado do cliente em uma tabela existente (CLI)
- Como usar carimbos de data/hora do lado do cliente em instruções de Linguagem de Manipulação de Dados (DML)

## Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (console)

Siga estas etapas para criar uma nova tabela com carimbos de data/hora no lado do cliente ativados no console do Amazon Keyspaces.

Como criar uma nova tabela com carimbos de data/hora do lado do cliente (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas) e Create table (Criar tabela).
3. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a nova tabela.
4. Na seção Esquema, crie o esquema para sua tabela.
5. Na seção Configurações da tabela, selecione Personalizar configurações.
6. Continue com os carimbos de data/hora do lado do cliente.

   Escolha Ativar carimbos de data/hora do lado do cliente para ativar os carimbos de data/hora do lado do cliente para a tabela.
7. Escolha Create table. Sua tabela é criada com os carimbos de data/hora do lado do cliente ativados.

## Como ativar os carimbos de data/hora do lado do cliente em tabelas existentes (console)

Siga estas etapas para ativar os carimbos de data/hora do lado do cliente para tabelas existentes usando o Amazon Keyspaces AWS Management Console.

Para ativar os carimbos de data/hora do lado do cliente para uma tabela existente (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Escolha a tabela que você deseja atualizar e selecione a guia Configurações adicionais.
3. Na guia Configurações adicionais, acesse Modificar carimbos de data/hora do lado do cliente e selecione Ativar carimbos de data/hora do lado do cliente
4. Escolha Salvar alterações para alterar as configurações da tabela.

## Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CQL)

Para ativar os carimbos de data/hora do lado do cliente ao criar uma nova tabela, você pode usar a seguinte instrução CQL.

```
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'}};
```

Para confirmar as configurações de carimbos de data/hora do lado do cliente para a nova tabela, use uma instrução `SELECT` para revisar as `custom_properties` conforme mostrado no exemplo a seguir.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

A saída dessa declaração mostra o status dos carimbos de data/hora do lado do cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

## Como ativar carimbos de data/hora do lado do cliente para tabelas existentes usando **ALTER TABLE** (CQL)

Para ativar os carimbos de data/hora do lado do cliente para uma tabela existente, você pode usar a seguinte instrução CQL.

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps': {'status':
 'enabled'}};;
```

Para confirmar as configurações de carimbos de data/hora do lado do cliente para a nova tabela, use uma instrução `SELECT` para revisar as `custom_properties` conforme mostrado no exemplo a seguir.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

A saída dessa declaração mostra o status dos carimbos de data/hora do lado do cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

## Como criar uma nova tabela com carimbos de data/hora do lado do cliente ativados (CLI)

Para ativar carimbos de data/hora no lado do cliente ao criar uma nova tabela, você pode usar a seguinte instrução de CLI.

```
./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}]'
```

Para confirmar se os carimbos de data/hora do lado do cliente estão ativados para a nova tabela, execute o código a seguir.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

A saída deve ser semelhante a este exemplo:

```
{
    "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": ""
    }
}
```

## Como ativar carimbos de data/hora do lado do cliente em uma tabela existente (CLI)

Para ativar os carimbos de data/hora do lado do cliente para uma tabela existente usando a CLI, você pode usar o código a seguir.

```
./aws keyspaces update-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED'
```

Para confirmar se os carimbos de data/hora do lado do cliente estão ativados para a tabela, execute o código a seguir.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

A saída deve ser semelhante a este exemplo:

```
{
    "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": ""
    }
```

```
}
```

## Como usar carimbos de data/hora do lado do cliente em instruções de Linguagem de Manipulação de Dados (DML)

Depois de ativar os carimbos de data/hora do lado do cliente, você pode passar o carimbo de data/hora em suas declaraçõesINSERT, UPDATE e DELETE com a cláusula USING TIMESTAMP. O valor do carimbo de data/hora bigint representa um número de microssegundos desde a hora base padrão conhecida como epoch: 1º de janeiro de 1970 às 00:00:00 GMT. Um carimbo de data/hora fornecido pelo cliente deve estar entre o intervalo de 2 dias no passado e 5 minutos no futuro a partir do horário atual do relógio. O Amazon Keyspaces mantém metadados de carimbo de data/hora durante a vida útil dos dados. Você pode usar a função WRITETIME para pesquisar registros de data e hora que ocorreram anos atrás. Para obter mais informações sobre sintaxe de CQL, consulte the section called “Instruções DML”.

A instrução CQL a seguir é um exemplo de como usar um carimbo de data/hora como um 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;
```

Se você não especificar um carimbo de data/hora na sua consulta CQL, o Amazon Keyspaces usará o carimbo de data/hora passado pelo seu driver de cliente. Se nenhum carimbo de data/hora for fornecido pelo driver de cliente, o Amazon Keyspaces atribuirá um carimbo de data/hora do lado do servidor para sua operação de gravação.

Para ver o valor do carimbo de data/hora armazenado em uma coluna específica, você pode usar a função WRITETIME em uma instrução SELECT, conforme mostrado no exemplo a seguir.

```
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;
```

# Como criar recursos do Amazon Keyspaces com o AWS CloudFormation

O Amazon Keyspaces (para Apache Cassandra) é integrado ao AWS CloudFormation, um serviço que ajuda você a modelar e configurar seus recursos da AWS, para que você possa passar menos tempo criando e gerenciando seus recursos e sua infraestrutura. Você cria um modelo que descreve todos os recursos da AWS que você deseja (como keyspaces e tabelas) e o AWS CloudFormation cuida do provisionamento e da configuração desses recursos para você.

Quando você usa o AWS CloudFormation, é possível reutilizar o modelo para configurar os recursos do Amazon Keyspaces repetidamente e de forma consistente. Descreva seus recursos uma vez e depois provisione os mesmos recursos repetidamente em várias regiões e Contas da AWS.

## Amazon Keyspaces e modelos AWS CloudFormation

Para provisionar e configurar recursos para o Amazon Keyspaces, é preciso entender os modelos do AWS CloudFormation. Os modelos são arquivos de texto formatados em JSON ou YAML. Esses modelos descrevem os recursos que você deseja provisionar nas suas pilhas do AWS CloudFormation. Se você não estiver familiarizado com JSON ou YAML, poderá usar o AWS CloudFormation Designer para ajudá-lo a começar a usar os modelos do AWS CloudFormation. Para obter mais informações, consulte O que é o Designer de AWS CloudFormation? no Manual do usuário do AWS CloudFormation.

O Amazon Keyspaces oferece suporte à criação de keyspaces e tabelas em AWS CloudFormation. Para as tabelas que você cria usando modelos AWS CloudFormation, você pode especificar o esquema, o modo de leitura/gravação e as configurações de throughput provisionada. Para obter mais informações, incluindo exemplos de modelos JSON e YAML para keyspaces e tabelas, consulte Referência de tipo de recurso do Cassandra no Guia do usuário do AWS CloudFormation.

## Saiba mais sobre o AWS CloudFormation

Para saber mais sobre o AWS CloudFormation, consulte os seguintes recursos:

- AWS CloudFormation
- Manual do usuário do AWS CloudFormation
- Guia do usuário da interface de linha de comando do AWS CloudFormation

# Monitoramento do Amazon Keyspaces (para Apache Cassandra)

O monitoramento é uma parte importante para manter a confiabilidade, a disponibilidade e o desempenho do Amazon Keyspaces e das outras soluções da AWS. A AWS fornece as ferramentas de monitoramento a seguir para observar o Amazon Keyspaces, informar quando algo está errado e realizar ações automaticamente quando apropriado:

- O Amazon Keyspaces oferece um painel pré-configurado no AWS Management Console que mostra a latência e os erros agregados em todas as tabelas da conta.
- O Amazon CloudWatch monitora os recursos da AWS e as aplicações que você executa na AWS em tempo real. Você pode coletar e rastrear métricas com painéis personalizados. Por exemplo, você pode criar uma referência para o desempenho normal do Amazon Keyspaces em seu ambiente medindo o desempenho em vários momentos e sob diferentes condições de carga. À medida que você monitora o Amazon Keyspaces, armazene dados de monitoramento históricos para compará-los com os dados de performance atuais, identificar padrões de performance normais e anomalias de performance e elaborar métodos para resolver problemas. Para estabelecer uma referência, você deve monitorar, no mínimo, os erros do sistema. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch.
- Alarmes do Amazon CloudWatch monitoram uma única métrica ao longo de um período que você especificar e realize uma ou mais ações com base no valor da métrica em relação a um determinado limite ao longo de vários períodos. Por exemplo, se você usa o Amazon Keyspaces no modo provisionado com ajuste de escala automático do aplicativo, a ação é uma notificação enviada pelo Amazon Simple Notification Service (Amazon SNS) para avaliar uma política de ajuste de escala automático do aplicativo.

  Os alarmes do CloudWatch não invocam ações só porque estão em um determinado estado. O estado deve ter sido alterado e mantido por uma quantidade especificada de períodos. Para obter mais informações, consulte Monitorando o Amazon Keyspaces com a Amazon CloudWatch.
- O Amazon CloudWatch Logs permite monitorar, armazenar e acessar os arquivos de log de tabelas do Amazon Keyspaces, do CloudTrail e de outras fontes. O CloudWatch Logs pode monitorar informações nos arquivos de log e notificar você quando determinados limites forem atingidos. Você também pode arquivar seus dados de log em armazenamento resiliente. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch Logs.

- O AWS CloudTrail captura chamadas de API e eventos relacionados feitos por sua conta da Conta da AWS ou em nome dela e entrega os arquivos de log a um bucket do Amazon S3 que você especifica. Você pode identificar quais usuários e contas chamaram a AWS, o endereço IP de origem do qual as chamadas foram feitas e quando elas ocorreram. Para obter mais informações, consulte o Guia do usuário do AWS CloudTrail.

O Amazon EventBridge é um serviço de barramento de eventos sem servidor que facilita a conexão de aplicações a dados de diversas origens. O EventBridge fornece um fluxo de dados em tempo real de suas próprias aplicações, de aplicações de software como serviço (SaaS) e de serviços da AWS e roteia esses dados para destinos como o Lambda. Isso permite monitorar eventos que ocorrem em serviços e crie arquiteturas orientadas a eventos. Para obter mais informações, consulte o Guia do usuário do Amazon EventBridge.

Tópicos


- Monitorando o Amazon Keyspaces com a Amazon CloudWatch
- Registro de chamadas de API do Amazon Keyspaces com AWS CloudTrail


# Monitorando o Amazon Keyspaces com a Amazon CloudWatch

Você pode monitorar o Amazon Keyspaces usando a Amazon CloudWatch, que coleta dados brutos e os processa em métricas legíveis, quase em tempo real. Essas estatísticas são mantidas por 15 meses, de maneira que você possa acessar informações históricas e ter uma perspectiva melhor de como o aplicativo web ou o serviço está se saindo.

Você também pode definir alarmes que observam determinados limites e enviam notificações ou realizam ações quando esses limites são atingidos. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.

> **Note**
>
> Para começar rapidamente com um CloudWatch painel pré-configurado mostrando métricas comuns para o Amazon Keyspaces, você pode usar AWS CloudFormation um modelo disponível em. https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates

Tópicos

- Como usar métricas do Amazon Keyspaces?
- Métricas e dimensões do Amazon Keyspaces
- Criação de CloudWatch alarmes para monitorar o Amazon Keyspaces


## Como usar métricas do Amazon Keyspaces?

As métricas informadas pelo Amazon Keyspaces fornecem informações que você pode analisar de diferentes maneiras. A lista a seguir mostra alguns usos comuns para as métricas. Essas são sugestões para você começar, e não uma lista abrangente. Para obter mais informações sobre a retenção de métricas, consulte Métricas.

| Como? | Métricas relevantes |
| --- | --- |
| Como posso determinar se ocorreu algum erro do sistema? | Você pode monitorar `SystemErrors` para determinar se todas as solicitações resultaram em um código de erro de servidor. Normalmente, essa métrica deve ser igual a zero. Se não for o caso, talvez você deva investigar. |
| Como posso comparar a média de leitura provisionada com a capacidade de leitura consumida? | Para monitorar a capacidade média de leitura provisionada e a capacidade de leitura consumida<br>1. Defina o Período para `ConsumedReadCapacityUnits` e `ProvisionedReadCapacityUnits` para o intervalo que você deseja monitorar.<br>2. Altere a Estatística para `ConsumedReadCapacityUnits` de `Average` a `Sum`.<br>3. Crie uma nova Expressão matemática vazia.<br>4. Na seção Detalhes da nova expressão matemática, insira o ID de **`ConsumedReadCapacityUnits`** e divida a métrica pela função CloudWatch PERÍODO da métrica (metric_id/ (PERIOD (metric_id)).<br>5. Desmarque `ConsumedReadCapacityUnits` .<br><br>Agora você pode comparar sua capacidade média de leitura consumida com sua capacidade provisionada. Para obter mais |

| Como? | Métricas relevantes |
| --- | --- |
| | informações sobre funções aritméticas básicas e como criar uma série temporal, consulte Usar matemática métrica. |
| Como posso comparar a média de gravação provisionada com a capacidade de gravação consumida? | Para monitorar a capacidade média de gravação provisionada e a capacidade de gravação consumida<br>1. Defina o Período para `ConsumedWriteCapacityUnits` e `ProvisionedWriteCapacityUnits` para o intervalo que você deseja monitorar.<br>2. Altere a Estatística para `ConsumedWriteCapacityUnits` de `Average` a `Sum`.<br>3. Crie uma nova Expressão matemática vazia.<br>4. Na seção Detalhes da nova expressão matemática, insira o ID de **`ConsumedWriteCapacityUnits`** e divida a métrica pela função CloudWatch PERÍODO da métrica (metric_id/ (PERIOD (metric_id)).<br>5. Desmarque `ConsumedWriteCapacityUnits` .<br><br>Agora você pode comparar sua capacidade média de gravação consumida com sua capacidade provisionada. Para obter mais informações sobre funções aritméticas básicas e como criar uma série temporal, consulte Usar matemática métrica. |

## Métricas e dimensões do Amazon Keyspaces

Quando você interage com o Amazon Keyspaces, ele envia as seguintes métricas e dimensões para a Amazon. CloudWatch Todas as métricas são agregadas e relatadas a cada minuto. É possível usar os procedimentos a seguir para visualizar as métricas do Amazon Keyspaces.

Para visualizar métricas usando o CloudWatch console

As métricas são agrupadas primeiro pelo namespace do serviço e, em seguida, por várias combinações de dimensão dentro de cada namespace.

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.

2. Se necessário, altere a região da . Na barra de navegação, selecione a Região em que os seus recursos da AWS residem. Para obter mais informações, consulte Endpoints de serviço da AWS.
3. No painel de navegação, selecione Métricas.
4. Na guia Todas as métricas, selecione `AWS/Cassandra`..

Para visualizar métricas usando a AWS CLI

- Em um prompt de comando, use o seguinte comando.

```
aws cloudwatch list-metrics --namespace "AWS/Cassandra"
```

## Métricas e dimensões do Amazon Keyspaces

As métricas e dimensões que o Amazon Keyspaces envia para a Amazon CloudWatch estão listadas aqui.

Métricas do Amazon Keyspaces

A Amazon CloudWatch agrega métricas do Amazon Keyspaces em intervalos de um minuto.

Nem todas as estatísticas, como `Average` ou `Sum`, são aplicáveis a todas as métricas. No entanto, todos esses valores estão disponíveis por meio do console do Amazon Keyspaces, ou usando o CloudWatch console AWS CLI, ou AWS SDKs para todas as métricas. Na tabela a seguir, cada métrica tem uma lista de estatísticas válidas aplicáveis a essa métrica.

| Métrica | Descrição |
| --- | --- |
| `AccountMaxTableLevelReads` | O número máximo de unidades de capacidade de leitura que podem ser usadas por uma tabela de uma conta. Para tabelas sob demanda, esse valor limita o máximo de unidades de solicitação de leitura que uma tabela pode usar.<br><br>Unidades: `Count`<br><br>Estatística válida: |

| Métrica | Descrição |
| --- | --- |
| | • `Maximum` – O número máximo de unidades de capacidade de leitura que podem ser usadas por uma tabela da conta. |
| `AccountMaxTableLevelWrites` | O número máximo de unidades de capacidade de gravação que podem ser usadas por uma tabela de uma conta. Para tabelas sob demanda, esse valor limita o máximo de unidades de solicitação de gravação que uma tabela pode usar.<br><br>Unidades: `Count`<br><br>Estatística válida:<br><br>• `Maximum` – O número máximo de unidades de capacidade de gravação que podem ser usadas por uma tabela da conta. |
| `AccountProvisionedReadCapacityUtilization` | O percentual de unidades de capacidade de leitura provisionada utilizadas por uma conta.<br><br>Unidades: `Percent`<br><br>Estatística válida:<br><br>• `Maximum`: o percentual máximo de unidades de capacidade de leitura provisionada utilizadas pela conta.<br>• `Minimum`: o percentual mínimo de unidades de capacidade de leitura provisionada utilizadas pela conta.<br>• `Average`: o percentual médio de unidades de capacidade de leitura provisionada utilizadas pela conta. A métrica é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de leitura provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `AccountProvisionedWriteCapacityUtilization` | O percentual de unidades de capacidade de gravação provisionada utilizadas por uma conta.<br><br>Unidades: `Percent`<br><br>Estatística válida:<br><br>• `Maximum`: a porcentagem máxima de unidades de capacidade de gravação provisionada utilizadas pela conta.<br>• `Minimum`: a porcentagem mínima de unidades de capacidade de gravação provisionada utilizadas pela conta.<br>• `Average`: a porcentagem média de unidades de capacidade de gravação provisionada utilizadas pela conta. A métrica é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de gravação provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `BillableTableSizeInBytes` | O tamanho faturável da tabela em bytes. É a soma do tamanho codificado de todas as linhas na tabela. Essa métrica o ajuda a monitorar os custos de armazenamento da tabela ao longo do tempo.<br><br>Unidades: `Bytes`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Maximum` – O tamanho máximo de armazenamento da tabela.<br>• `Minimum` – O tamanho mínimo de armazenamento da tabela.<br>• `Average` – O tamanho médio de armazenamento da tabela. Essa métrica é calculada em intervalos de 4 a 6 horas. |

| Métrica | Descrição |
| --- | --- |
| `ConditionalCheckFailedRequests` | O número de solicitações de gravação de transações leves (LWT) com falha. As operações `INSERT`, `UPDATE` e `DELETE` permitem que você forneça uma condição lógica que deve ser avaliada como true antes que a operação possa prosseguir. Se esta condição for avaliada como falsa, `ConditionalCheckFailedRequests` será incrementado em um. As verificações de condição avaliadas como falsas consomem unidades de capacidade de gravação com base no tamanho da linha. Para ter mais informações, consulte the section called “Transações leves”.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Minimum`<br>• `Maximum`<br>• `Average`<br>• `SampleCount`<br>• `Sum` |

| Métrica | Descrição |
| --- | --- |
| `ConsumedReadCapacityUnits` | O número de unidades de capacidade de leitura consumidas ao longo do período especificado. Para obter mais informações, consulte Modo de capacidade de leitura/gravação.<br><br>**Note**<br>Para entender sua utilização média de throughput por segundo, use a estatística `Sum` para calcular o throughput consumido no período de um minuto. Em seguida, divida a soma pelo número de segundos em um minuto (60) para calcular o `ConsumedReadCapacityUnits` médio por segundo (reconhecendo que essa média não destaca picos grandes, mas sim picos breves na atividade de leitura que ocorreram durante esse minuto). Para obter mais informações sobre como comparar a capacidade média de leitura consumida com a capacidade de leitura provisionada, consulte the section called “Uso de métricas do ”<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br>• `Minimum` – O número mínimo de unidades de capacidade de leitura consumidas por qualquer solicitação individual à tabela.<br>• `Maximum` – O número máximo de unidades de capacidade de leitura consumidas por qualquer solicitação individual à tabela. |

| Métrica | Descrição |
| --- | --- |
| | • `Average`: a capacidade de leitura por solicitação média consumida.<br><br>**Note**<br>O valor `Average` é influenciado por períodos de inatividade em que o valor da amostra será zero.<br><br>• `Sum`: o total de unidades de capacidade de leitura consumidas. Essa é a estatística mais útil para a métrica `ConsumedReadCapacityUnits` .<br>• `SampleCount` : O número de solicitações ao Amazon Keyspaces, mesmo que nenhuma capacidade de leitura tenha sido consumida.<br><br>**Note**<br>O valor `SampleCount` é influenciado por períodos de inatividade em que o valor da amostra será zero. |

| Métrica | Descrição |
| --- | --- |
| `ConsumedWriteCapacityUnits` | O número de unidades de capacidade de gravação consumidas ao longo do período especificado. Você pode recuperar a capacidade de gravação total consumida para uma tabela. Para obter mais informações, consulte Modo de capacidade de leitura/gravação.<br><br>**Note**<br>Para entender sua utilização média de throughput por segundo, use a estatística `Sum` para calcular o throughput consumido no período de um minuto. Em seguida, divida a soma pelo número de segundos em um minuto (60) para calcular o `ConsumedWriteCapacityUnits` médio por segundo (reconhecendo que essa média não destaca picos grandes, mas sim picos breves na atividade de gravação que ocorreram durante esse minuto). Para obter mais informações sobre como comparar a capacidade média de gravação consumida com a capacidade de gravação provisionada, consulte the section called “Uso de métricas do ”<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Minimum` – O número mínimo de unidades de capacidade de gravação consumidas por qualquer solicitação individual à tabela.<br>• `Maximum` – O número máximo de unidades de capacidade de gravação consumidas por qualquer solicitação individual à tabela. |

| Métrica | Descrição |
| --- | --- |
| | • `Average`: a capacidade de gravação por solicitação média consumida.<br><br>**Note**<br>O valor `Average` é influenciado por períodos de inatividade em que o valor da amostra será zero.<br><br>• `Sum`: o total de unidades de capacidade de gravação consumidas. Essa é a estatística mais útil para a métrica `ConsumedWriteCapacityUnits` .<br>• `SampleCount` : O número de solicitações ao Amazon Keyspaces, mesmo que nenhuma capacidade de gravação tenha sido consumida.<br><br>**Note**<br>O valor `SampleCount` é influenciado por períodos de inatividade em que o valor da amostra será zero. |

| Métrica | Descrição |
| --- | --- |
| `MaxProvisionedTableReadCapacityUtilization` | O percentual de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta de uma conta.<br><br>Unidades: `Percent`<br><br>Estatística válida:<br><br>• `Maximum`: o percentual máximo de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta da conta.<br>• `Minimum`: o percentual mínimo de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta da conta.<br>• `Average`: o percentual médio de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta da conta. A métrica é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de leitura provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `MaxProvisionedTableWriteCapacityUtilization` | A porcentagem da capacidade de gravação provisionada utilizada pela tabela de gravação provisionada mais alta de uma conta.<br><br>Unidades: `Percent`<br><br>Estatística válida:<br><br>• `Maximum` – A porcentagem máxima de unidades de capacidade de gravação provisionadas utilizadas pela tabela de gravação provisionada mais alta da conta.<br>• `Minimum` – A porcentagem mínima de unidades de capacidade de gravação provisionadas utilizadas pela tabela de gravação provisionada mais alta da conta.<br>• `Average` – A porcentagem média de unidades de capacidade de gravação provisionadas utilizadas pela tabela de gravação provisionada mais alta da conta. A métrica é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de gravação provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `PerConnectionRequestRateExceeded` | Solicitações para o Amazon Keyspaces que excedam a cota de taxa de solicitação por conexão. Cada conexão do cliente com o Amazon Keyspaces pode suportar até 3 mil solicitações de CQL por segundo. Os clientes podem criar várias conexões para aumentar o throughput.<br><br>Quando você usa a replicação multirregional, cada gravação replicada também contribui para essa cota. Como melhor prática, recomendamos aumentar o número de conexões com suas tabelas para evitar erros `PerConnectionRequestRateExceeded`. Não há limite para o número de conexões que você pode ter no Amazon Keyspaces.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `SampleCount`<br>• `Sum` |

| Métrica | Descrição |
| --- | --- |
| `ProvisionedReadCapacityUnits` | O número de unidades de capacidade de leitura provisionada para uma tabela.<br><br>A dimensão `TableName` retorna o `ProvisionedReadCapacityUnits` para a tabela.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Minimum`: a configuração mais baixa para a capacidade de leitura provisionada. Se você usar `ALTER TABLE` para aumentar a capacidade de leitura, esta métrica mostrará o valor mais baixo de `ReadCapacityUnits` provisionado durante esse período.<br>• `Maximum`: a configuração mais alta para a capacidade de leitura provisionada. Se você usar `ALTER TABLE` para diminuir a capacidade de leitura, esta métrica mostrará o valor mais alto de `ReadCapacityUnits` provisionado durante esse período.<br>• `Average`: a capacidade média de leitura provisionada. A métrica `ProvisionedReadCapacityUnits` é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de leitura provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `ProvisionedWriteCapacityUnits` | O número de unidades de capacidade de gravação provisionada para uma tabela.<br><br>A dimensão `TableName` retorna o `ProvisionedWriteCapacityUnits` para a tabela.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Minimum`: a configuração mais baixa para capacidade de gravação provisionada. Se você usar `ALTER TABLE` para aumentar a capacidade de gravação, esta métrica mostrará o valor mais baixo de `WriteCapacityUnits` provisionado durante esse período.<br>• `Maximum`: a configuração mais alta para a capacidade de gravação provisionada. Se você usar `ALTER TABLE` para diminuir a capacidade de gravação, esta métrica mostrará o valor mais alto de `WriteCapacityUnits` provisionado durante esse período.<br>• `Average`: a capacidade média de gravação provisionada. A métrica `ProvisionedWriteCapacityUnits` é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade de gravação provisionada, essa estatística poderá não refletir a média real. |

| Métrica | Descrição |
| --- | --- |
| `ReadThrottleEvents` | Solicitações ao Amazon Keyspaces que excedam a capacidade de leitura provisionada para uma tabela ou cotas em nível de conta, cotas de solicitação por conexão ou cotas em nível de partição.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `SampleCount`<br>• `Sum` |
| `ReplicationLatency` | Essa métrica se aplica somente a espaços de chaves multirregionais e mede o tempo necessário para replicar `updates`, `inserts` ou `deletes` de uma tabela de réplica para outra tabela de réplica em um espaço de chaves multirregional.<br><br>Unidades: `Millisecond`<br><br>Dimensões: `TableName, ReceivingRegion`<br><br>Estatística válida:<br><br>• `Average`<br>• `Maximum`<br>• `Minimum` |

| Métrica | Descrição |
| --- | --- |
| `ReturnedItemCountBySelect` | O número de linhas retornadas por consultas `SELECT` de várias linhas durante o período especificado. As consultas `SELECT` de várias linhas são consultas que não contêm a chave primária totalmente qualificada, como varreduras completas de tabelas e consultas de intervalo.<br><br>O número de itens retornados não é necessariamente o mesmo que o número de itens avaliados. Por exemplo, suponha que você tenha solicitado um `SELECT *` com `ALLOW FILTERING` em uma tabela que tinha 100 linhas, mas especificou um `WHERE` que reduziu os resultados para que apenas 15 linhas fossem retornadas. Nesse caso, a resposta de `SELECT` conteria um `ScanCount` de 100 e um `Count` de 15 linhas retornadas.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `Minimum`<br>• `Maximum`<br>• `Average`<br>• `SampleCount`<br>• `Sum` |

| Métrica | Descrição |
| --- | --- |
| `StoragePartitionThroughputCapacityExceeded` | Solicitações para uma partição de armazenamento do Amazon Keyspaces que excedam a capacidade de throughput da partição. As partições de armazenamento do Amazon Keyspaces podem suportar até 1.000 WCU/WRU por segundo e 3000 RCU/RRU por segundo. Recomendamos revisar seu modelo de dados para distribuir o tráfego de leitura/gravação em mais partições para mitigar essas exceções.<br><br>**Note**<br>As partições lógicas do Amazon Keyspaces podem abranger várias partições de armazenamento e são virtualmente ilimitadas em tamanho.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `SampleCount`<br>• `Sum` |
| `SuccessfulRequestCount` | O número de solicitações bem-sucedidas processadas durante o período especificado.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `SampleCount` |

| Métrica | Descrição |
| --- | --- |
| `SuccessfulRequestLatency` | As solicitações bem-sucedidas para o Amazon Keyspaces Streams durante o período especificado. O `SuccessfulRequestLatency` pode fornecer dois tipos diferentes de informações:<br>• O tempo decorrido para solicitações bem-sucedidas (`Minimum`, `Maximum`, `Sum` ou `Average`).<br>• O número de solicitações bem-sucedidas (`SampleCount` ).<br><br>`SuccessfulRequestLatency` reflete a atividade somente no Amazon Keyspaces e não leva em consideração a latência da rede nem a atividade no lado do cliente.<br><br>Unidades: `Milliseconds`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br>• `Minimum`<br>• `Maximum`<br>• `Average`<br>• `SampleCount` |

| Métrica | Descrição |
| --- | --- |
| SystemErrors | As solicitações para o Amazon Keyspaces que geram um ServerError durante o período de tempo especific ado. Um ServerError geralmente indica um erro de serviço interno.<br><br>Unidades: Count<br><br>Dimensões: Keyspace, TableName, Operation<br><br>Estatística válida:<br><br>• Sum<br>• SampleCount |
| SystemReconciliati onDeletes | As unidades consumidas para excluir dados gravados com marca de exclusão quando a data e a hora do lado do cliente estão habilitadas. Cada SystemRec onciliationDelete fornece capacidade suficient e para excluir ou atualizar até 1 KB de dados por linha. Por exemplo, para atualizar uma linha que armazena 2,5 KB de dados e excluir uma ou mais colunas dentro da linha ao mesmo tempo, são necessárias 3 SystemRec onciliationDeletes . Ou, para excluir uma linha inteira que contém 3,5 KB de dados, são necessárias 4 SystemReconciliationDeletes .<br><br>Unidades: Count<br><br>Dimensões: Keyspace, TableName<br><br>Estatística válida:<br><br>• Sum – O número total de SystemReconciliati onDeletes consumidos em um período. |

| Métrica | Descrição |
| --- | --- |
| `TTLDeletes` | As unidades consumidas para excluir ou atualizar dados em uma linha usando vidas úteis (TTL). Cada `TTLDelete` fornece capacidade suficiente para excluir ou atualizar até 1 KB de dados por linha. Por exemplo, para atualizar uma linha que armazena 2,5 KB de dados e excluir uma ou mais colunas dentro da linha ao mesmo tempo, são necessárias 3 exclusões de TTL. Ou, para excluir uma linha inteira que contém 3,5 KB de dados, são necessárias 4 exclusões de TTL.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName`<br><br>Estatística válida:<br><br>• `Sum` – O número total de `TTLDeletes` consumidos em um período. |

| Métrica | Descrição |
| --- | --- |
| `UserErrors` | Solicitações para o Amazon Keyspaces que geram um erro `InvalidRequest` durante o período de tempo especificado. Um código `InvalidRequest` geralmente indica um erro no lado do cliente, como uma combinação inválida de parâmetros, uma tentativa de atualizar uma tabela inexistente ou uma assinatura de solicitação incorreta.<br><br>`UserErrors` representa o agregado de solicitações inválidas para a atual Região da AWS e a atual. Conta da AWS<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `Sum`<br>• `SampleCount` |
| `WriteThrottleEvents` | Solicitações ao Amazon Keyspaces que excedam a capacidade de gravação provisionada para uma tabela ou cotas em nível de conta, cotas de solicitação por conexão ou cotas em nível de partição.<br><br>Unidades: `Count`<br><br>Dimensões: `Keyspace, TableName, Operation`<br><br>Estatística válida:<br><br>• `SampleCount`<br>• `Sum` |

Dimensões para métricas do Amazon Keyspaces

As métricas do Amazon Keyspaces são qualificadas de acordo com os valores da conta, do nome da tabela ou da operação. Você pode usar o CloudWatch console para recuperar dados do Amazon Keyspaces em qualquer uma das dimensões na tabela a seguir.

| Dimensão | Descrição |
| --- | --- |
| `Keyspace` | Esta dimensão limita os dados a um espaço de chaves específico. Este valor pode ser qualquer espaço de chaves na Região atual e na Conta da AWS atual. |
| `Operation` | Esta dimensão limita os dados a uma das operações CQL do Amazon Keyspaces, como as operações `INSERT` ou `SELECT`. |
| `TableName` | Esta dimensão limita os dados a uma tabela específica. Este valor pode ser qualquer nome de tabela na Região atual e na Conta da AWS atual. Se o nome da tabela não for exclusivo na conta, você também deverá especificar o `Keyspace`. |

## Criação de CloudWatch alarmes para monitorar o Amazon Keyspaces

Você pode criar um CloudWatch alarme da Amazon para o Amazon Keyspaces que envia uma mensagem do Amazon Simple Notification Service (Amazon SNS) quando o alarme muda de estado. Um alarme observa uma única métrica por um período tempo que você especifica. Ele executa uma ou mais ações com base no valor da métrica em relação a um limite especificado ao longo de vários períodos. A ação é uma notificação enviada a um tópico do Amazon SNS ou a uma política de ajuste de escala automático do aplicativo.

Quando você usa o Amazon Keyspaces no modo provisionado com o Application Auto Scaling, o serviço cria dois pares de alarmes em seu nome. CloudWatch Cada par representa seus limites superiores e inferiores para configurações de throughput provisionado e consumido. Esses CloudWatch alarmes são acionados quando a utilização real da tabela se desvia da sua meta de utilização por um longo período de tempo. Para saber mais sobre CloudWatch os alarmes criados pelo Application Auto Scaling, consulte. the section called “Como funciona o ajuste de escala automático do Amazon Keyspaces”

Os alarmes invocam ações somente para mudanças de estado sustentadas. CloudWatch os alarmes não invocam ações simplesmente porque estão em um estado específico. O estado deve ter sido alterado e mantido por uma quantidade especificada de períodos.

Para obter mais informações sobre a criação de CloudWatch alarmes, consulte Usando CloudWatch alarmes da Amazon no Guia CloudWatch do usuário da Amazon.

# Registro de chamadas de API do Amazon Keyspaces com AWS CloudTrail

O Amazon Keyspaces é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon Keyspaces. CloudTrail captura chamadas da API Data Definition Language (DDL) e chamadas da API Data Manipulation Language (DML) para Amazon Keyspaces como eventos. As chamadas capturadas incluem as do console do Amazon Keyspaces e as chamadas programáticas para as operações de API do Amazon Keyspaces.

Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon Simple Storage Service (Amazon S3), incluindo eventos para o Amazon Keyspaces.

Se você não configurar uma trilha, ainda poderá ver os eventos suportados mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita ao Amazon Keyspaces, o endereço IP a partir do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

Tópicos

- Configurando entradas do arquivo de log do Amazon Keyspaces no CloudTrail
- Informações sobre Amazon Keyspaces Data Definition Language (DDL) em CloudTrail
- Informações sobre Amazon Keyspaces Data Manipulation Language (DML) em CloudTrail
- Noções básicas sobre entradas de arquivos de log do Amazon Keyspaces

# Configurando entradas do arquivo de log do Amazon Keyspaces no CloudTrail

Cada ação da API Amazon Keyspaces registrada CloudTrail inclui parâmetros de solicitação que são expressos na linguagem de consulta CQL. Para obter mais informações, consulte Referência da linguagem CQL.

Você pode exibir, pesquisar e baixar eventos recentes em sua Conta da AWS. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para obter um registro contínuo dos eventos em seu Conta da AWS, incluindo eventos do Amazon Keyspaces, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, a trilha se aplica a todas as AWS regiões. A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros.

Para obter mais informações, consulte os seguintes tópicos no Guia do usuário do AWS CloudTrail :

- Visão geral da criação de uma trilha
- CloudTrail serviços e integrações suportados
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões
- Recebendo arquivos de CloudTrail log de várias contas

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário root ou AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte o elemento CloudTrail userIdentity.

## Informações sobre Amazon Keyspaces Data Definition Language (DDL) em CloudTrail

CloudTrail é ativado no seu Conta da AWS quando você cria a conta. Quando uma atividade de DDL ocorre no Amazon Keyspaces, essa atividade é automaticamente registrada como CloudTrail um evento junto com AWS outros eventos de serviço no histórico de eventos. A tabela a seguir mostra as instruções DDL registradas no Amazon Keyspaces.

| CloudTrail **`eventName`** | Statement | Ação de CQL | AWS Ação do 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` |

## Informações sobre Amazon Keyspaces Data Manipulation Language (DML) em CloudTrail

Para habilitar o registro de declarações DML do Amazon Keyspaces com CloudTrail, você deve primeiro habilitar o registro da atividade da API do plano de dados em. CloudTrail Você pode começar a registrar eventos DML do Amazon Keyspaces em trilhas novas ou existentes escolhendo registrar a atividade da tabela Cassandra do tipo de evento de dados usando o CloudTrail console ou definindo o valor `resources.type` para usar a CLI AWS ou `AWS::Cassandra::Table` as operações de API. CloudTrail Para obter mais informações, consulte Registrar eventos de dados.

A tabela a seguir mostra os eventos de dados registrados CloudTrail por for`Cassandra table`.

| CloudTrail **eventName** | Statement | Ação de CQL | AWS Ação do SDK |
|---|---|---|---|
| Select | DML | `SELECT` | `GetKeyspace` , `GetTable`, `ListKeyspaces` , `ListTables` `ListTagsForResource` |
| Inserir | DML | `INSERT` | nenhuma ação AWS do SDK disponível |
| Atualizar | DML | `UPDATE` | nenhuma ação AWS do SDK disponível |
| Delete | DML | `DELETE` | nenhuma ação AWS do SDK disponível |

## Noções básicas sobre entradas de arquivos de log do Amazon Keyspaces

CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra as `DropTable` ações `CreateKeyspaceDropKeyspace`,`CreateTable`, e:

```
{
  "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"
    }
  ]
}
```

O arquivo de log a seguir mostra um exemplo de uma instrução SELECT.

```
{
    "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"
    }
}
```

O arquivo de log a seguir mostra um exemplo de uma instrução `INSERT`.

```
{
    "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"
    }
}
```


O arquivo de log a seguir mostra um exemplo de uma instrução UPDATE.


```
{
    "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"
        }
}
```

O arquivo de log a seguir mostra um exemplo de uma instrução DELETE.

```
{
    "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"
```

```
    }
}
```

# Segurança no Amazon Keyspaces (para Apache Cassandra)

A segurança da nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se contará com um datacenter e uma arquitetura de rede criados para atender aos requisitos das organizações com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre a AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa produtos da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. A eficácia da nossa segurança é regularmente testada e verificada por auditores de terceiros como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao Amazon Keyspaces, consulte Serviços da AWS no escopo por programa de conformidade.
- Segurança da nuvem: sua responsabilidade é determinada pelo serviço da AWS que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da sua organização e as leis e regulamentos aplicáveis.

Esta documentação o ajudará a entender como aplicar o modelo de responsabilidade compartilhada ao usar o Amazon Keyspaces. Os tópicos a seguir mostram como configurar o Amazon Keyspaces para atender aos seus objetivos de segurança e de conformidade. Você também aprenderá como usar outros serviços da AWS que podem ajudar a monitorar e proteger seus recursos do Amazon Keyspaces.

Tópicos


- Proteção de dados no Amazon Keyspaces
- AWS Identity and Access Management para Amazon Keyspaces
- Validação de conformidade do Amazon Keyspaces (para Apache Cassandra)
- Resiliência e recuperação de desastres no Amazon Keyspaces
- Segurança da infraestrutura no Amazon Keyspaces
- Análise de configuração e vulnerabilidade no Amazon Keyspaces
- Práticas recomendadas de segurança para o Amazon Keyspaces

# Proteção de dados no Amazon Keyspaces

O AWS modelo de responsabilidade compartilhada se aplica à proteção de dados no Amazon Keyspaces (para Apache Cassandra). Conforme descrito nesse modelo, a AWS é responsável por proteger a infraestrutura global que executa toda a Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados. Para mais informações sobre a proteção de dados na Europa, consulte o artigo AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja asConta da AWS credenciais da e configure as contas de usuário individuais com o AWS IAM Identity Center ou o AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA [multi-factor authentication]) com cada conta.
- Use SSL/TLS para se comunicar com os atributos da AWS. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure o registro em log das atividades da API e do usuário com o .AWS CloudTrail
- Use AWS as soluções de criptografia da , juntamente com todos os controles de segurança padrão dos Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar a AWS por meio de uma interface de linha de comandos ou uma API, use um endpoint do FIPS. Para ter mais informações sobre endpoints do FIPS, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Name (Nome). Isso também vale para o uso do Amazon Keyspaces ou de outros Serviços da AWS com o console, a API, a AWS CLI ou os SDKs AWS. Quaisquer dados inseridos em tags ou campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou

de diagnóstico. Se você fornecer um URL para um servidor externo, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse servidor.

Tópicos


- Criptografia em repouso no Amazon Keyspaces
- Criptografia em trânsito no Amazon Keyspaces
- Privacidade do tráfego entre redes no Amazon Keyspaces


## Criptografia em repouso no Amazon Keyspaces

A criptografia em repouso do Amazon Keyspaces (para Apache Cassandra) fornece segurança aprimorada ao criptografar seus dados em repouso usando chaves de criptografia armazenadas no AWS Key Management Service (AWS KMS). Essa funcionalidade ajuda a reduzir a carga e complexidade operacionais necessárias para proteger dados confidenciais. Com a criptografia de dados em repouso, você pode criar aplicativos confidenciais que atendem a requisitos rigorosos de conformidade e regulamentação de criptografia para proteção de dados.

A criptografia em repouso do Amazon Keyspaces usa o Advanced Encryption Standard de 256 bits (AES-256). Isso ajuda a proteger seus dados contra o acesso não autorizado ao armazenamento subjacente.

O Amazon Keyspaces criptografa e o descriptografa os dados da tabela de forma transparente. O Amazon Keyspaces usa criptografia envelopada e uma hierarquia de chaves para proteger as chaves de criptografia de dados. Ele se integra ao AWS KMS para armazenar e gerenciar a chave de criptografia raiz. Para obter mais informações sobre a hierarquia de chaves da criptografia, consulte o the section called “Como funciona”. Para obter mais informações sobre conceitos do AWS KMS como criptografia envelopada, consulte Conceitos do serviço de gerenciamento do AWS KMS no Guia do desenvolvedor do AWS Key Management Service.

Ao criar uma nova tabela, você pode escolher uma das seguintes chaves AWS KMS (chaves KMS):

- Chave pertencente à AWS: Esta é a opção de criptografia padrão. A chave é de propriedade do Amazon Keyspaces (sem custo adicional).
- Chave gerenciada pelo cliente: a chave é armazenada na sua conta e é você que a cria, detém e gerencia. Você tem controle total sobre a chave gerenciada pelo cliente (cobranças do AWS KMS são aplicáveis).

É possível alternar entre a Chave pertencente à AWS e a chave gerenciada pelo cliente a qualquer momento. É possível especificar uma chave gerenciada pelo cliente quando você cria uma nova tabela ou alterou a chave KMS de uma tabela existente usando o console ou programaticamente usando as instruções CQL. Para saber como, consulte Criptografia em repouso: como usar chaves gerenciadas pelo cliente para criptografar tabelas no Amazon Keyspaces.

A criptografia em repouso que usa a Chaves pertencentes à AWS é oferecida sem custo adicional. No entanto, as cobranças do AWS KMS se aplicam a chaves gerenciadas pelo cliente. Para obter mais informações sobre a definição de preço, consulte Definição de preço do AWS KMS.

A criptografia em repouso do Amazon Keyspaces está disponível em todas as regiões da Regiões da AWS, incluindo as regiões AWS China (Pequim) e AWS China (Ningxia). Para obter mais informações, consulte Criptografia em repouso: como funciona no Amazon Keyspaces.

Tópicos


- Criptografia em repouso: como funciona no Amazon Keyspaces
- Criptografia em repouso: como usar chaves gerenciadas pelo cliente para criptografar tabelas no Amazon Keyspaces


## Criptografia em repouso: como funciona no Amazon Keyspaces

A criptografia em repouso do Amazon Keyspaces (para Apache Cassandra) criptografa seus dados usando Advanced Encryption Standard (AES-256) de 256 bits. Isso ajuda a proteger seus dados contra o acesso não autorizado ao armazenamento subjacente. Todos os dados do cliente nas tabelas do Amazon Keyspaces são criptografados em repouso por padrão, e a criptografia do lado do servidor é transparente, o que significa que não são necessárias alterações nos aplicativos.

A criptografia em repouso integra-se com AWS Key Management Service (AWS KMS) para gerenciamento da chave de criptografia usada para criptografar suas tabelas. Ao criar uma nova tabela ou atualizar uma tabela existente, você pode escolher uma das seguintes opções de chave AWS KMS:

- Chave pertencente à AWS: Esta é a opção de criptografia padrão. A chave é de propriedade do Amazon Keyspaces (sem custo adicional).
- Chave gerenciada pelo cliente: a chave é armazenada na sua conta e é você que a cria, detém e gerencia. Você tem controle total sobre a chave gerenciada pelo cliente (cobranças do AWS KMS são aplicáveis).

chave AWS KMS (chave KMS)

A criptografia em repouso protege todos os seus dados do Amazon Keyspaces com uma chave AWS KMS. Por padrão, o Amazon Keyspaces usa uma Chave pertencente à AWS, uma chave de criptografia multilocatário que é criada e gerenciada em uma conta de serviço do Amazon Keyspaces.

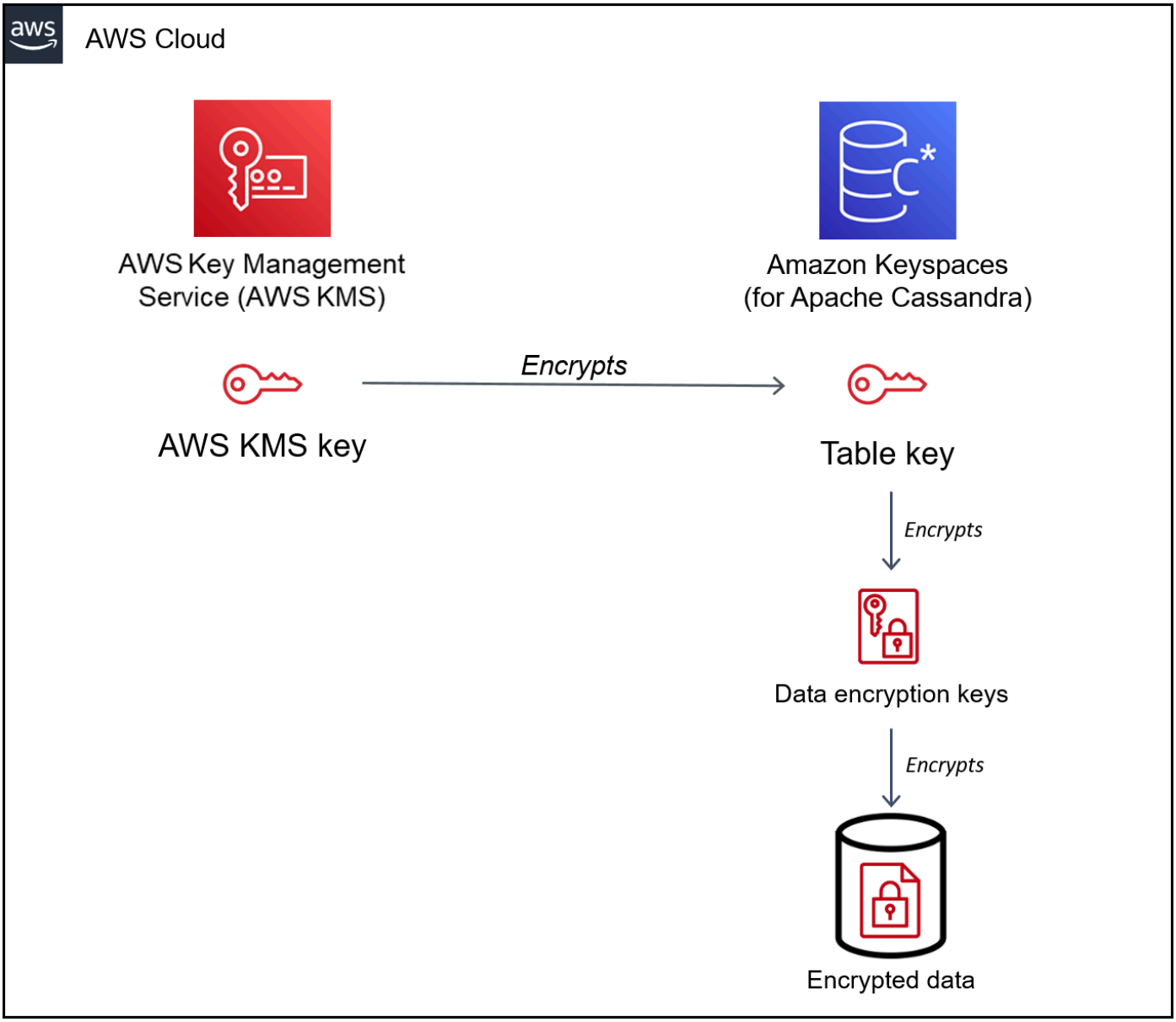
No entanto, você pode criptografar suas tabelas do Amazon Keyspaces usando uma chave gerenciada pelo cliente na sua Conta da AWS. Você pode selecionar uma chave do KMS diferente para cada tabela em uma keyspace. A chave do KMS selecionada para uma tabela também é usada para criptografar todos os metadados e backups restauráveis.

Você seleciona a chave do KMS para uma tabela ao criar ou atualizar essa tabela. É possível alterar a chave do KMS de uma tabela a qualquer momento, seja no console do Amazon Keyspaces ou usando a instrução ALTER TABLE. O processo de alternar chaves KMS é facilitado e não exige tempo de inatividade ou causa serviço de degradação.

Hierarquia de chaves

O Amazon Keyspaces usa uma hierarquia de chaves para criptografar dados. Nessa hierarquia de chaves, a chave KMS é a chave raiz. É usado para criptografar e descriptografar a chave de criptografia de tabela do Amazon Keyspaces. A chave de criptografia da tabela é usada para criptografar as chaves de criptografia usadas internamente pelo Amazon Keyspaces para criptografar e descriptografar dados ao realizar operações de leitura e gravação.

Com a hierarquia de chaves de criptografia, você pode fazer alterações na chave KMS sem precisar recriptografar os dados ou afetar os aplicativos e as operações de dados em andamento.




Chave de tabela

A chave de tabela do Amazon Keyspaces é usada como uma chave de criptografia de chaves. O Amazon Keyspaces usa a chave de tabela para proteger as chaves de criptografia de dados internas usadas para criptografar os dados armazenados em tabelas, arquivos de log e backups restauráveis. O Amazon Keyspaces gera uma chave de criptografia de dados exclusiva para cada estrutura subjacente em uma tabela. No entanto, várias linhas da tabela podem ser protegidas pela mesma chave de criptografia de dados.

Quando você define pela primeira vez a chave KMS como uma chave gerenciada pelo cliente, AWS KMS gera uma chave de dados. A chave AWS KMS de dados se refere à chave de tabela no Amazon Keyspaces.

Quando você acessa uma tabela criptografada, o Amazon Keyspaces envia uma solicitação para o AWS KMS usar a chave do KMS para descriptografar a chave de tabela. Ele usa a chave de tabela de texto simples para descriptografar as chaves de criptografia de dados do Amazon Keyspaces e usa as chaves de criptografia de dados de texto simples para descriptografar os dados da tabela.

O Amazon Keyspaces armazena e usa a chave de tabela e as chaves de criptografia de dados fora do AWS KMS. Ele protege todas as chaves com a criptografia Advanced Encryption Standard (AES) e chaves de criptografia de 256 bits. Depois, armazena as chaves criptografadas com os dados criptografados para que estejam disponíveis para descriptografar os dados da tabela sob demanda.

Armazenamento em cache de chave de tabela

Para evitar chamar o AWS KMS para cada operação do Amazon Keyspaces, o Amazon Keyspaces armazena chaves de tabela de texto simples para cada conexão na memória. Quando o Amazon Keyspaces recebe uma solicitação para a chave de tabela armazenada em cache após cinco minutos de inatividade, ele envia uma nova solicitação ao AWS KMS para descriptografar a chave de tabela. Essa chamada captura todas as alterações feitas nas políticas de acesso da chave do KMS no AWS KMS ou no AWS Identity and Access Management (IAM) desde a última solicitação para descriptografar a chave de tabela.

criptografia envelopada

Se você alterar a chave da tabela gerenciada pelo cliente, o Amazon Keyspaces gera outra chave de tabela. Depois, ele usa a nova chave de tabela para criptografar novamente as chaves de criptografia de dados. Ele também usa a nova chave de tabela para criptografar chaves de tabela anteriores que são usadas para proteger backups restauráveis. Esse processo é chamado de criptografia envelopada. Isso garante que você possa acessar backups restauráveis mesmo se você alternar a chave gerenciada pelo cliente. Para obter mais informações sobre a criptografia envelopada, consulte Criptografia de envelope no Guia do desenvolvedor do AWS Key Management Service.

Tópicos

- chaves de propriedade da AWS
- Chaves gerenciadas pelo cliente
- Notas de uso de criptografia em repouso

chaves de propriedade da AWS

Chaves pertencentes à AWS não estão armazenadas na sua Conta da AWS. Elas fazem parte de um conjunto de chaves do KMS que a AWS detém e gerencia para serem usadas em várias contas das Contas da AWS. Os serviços da AWS podem usar Chaves pertencentes à AWS para proteger seus dados.

Você não pode visualizar, gerenciar ou usar Chaves pertencentes à AWS, nem auditar seu uso. No entanto, você não precisa fazer nenhum trabalho nem alterar nenhum programa para proteger as chaves que criptografam seus dados.

Não é cobrada taxa mensal nem taxa de uso para usar Chaves pertencentes à AWS, e elas não são contabilizadas com base nas cotas do AWS KMS para a sua conta.

Chaves gerenciadas pelo cliente

Chaves gerenciadas pelo cliente são chaves em sua Conta da AWS que você cria, detém e gerencia. Você tem controle total sobre essas chaves KMS.

Use uma chave gerenciada pelo cliente para obter os seguintes recursos:

- Você cria e gerencia a chave gerenciada pelo cliente, incluindo a configuração e a manutenção de políticas de chaves, políticas do IAM e concessões para controlar o acesso à chave gerenciada pelo cliente. Você pode habilitar e desabilitar a chave gerenciada pelo cliente, habilitar e desabilitar a alternância automática de chaves e programar a chave gerenciada pelo cliente para ser deletada quando ela não estiver mais em uso. Você pode criar tags e aliases para as chaves gerenciadas pelo cliente que você gerencia.
- Você pode usar uma chave gerenciada pelo cliente com material de chave importado ou uma chave gerenciada pelo cliente em um armazenamento de chaves personalizado que você possui e gerencia.
- Você pode usar o AWS CloudTrail e o Amazon CloudWatch Logs para rastrear as solicitações que o Amazon Keyspaces envia ao AWS KMS por você. Para obter mais informações, consulte the section called “Etapa 6: Configurar o monitoramento com AWS CloudTrail”.

As chaves gerenciadas pelo cliente geram uma cobrança para cada chamada de API, e as cotas do AWS KMS são aplicáveis a essas chaves do KMS. Para obter mais informações, consulte Cotas AWS KMS de recurso ou de solicitação.

Quando você especifica uma chave gerenciada pelo cliente como a chave de criptografia raiz de uma tabela, os backups restauráveis são criptografados com a mesma chave de criptografia especificada para a tabela no momento em que o backup é criado. Se a chave KMS da tabela for alternada, o envelopamento da chave garante que a chave KMS mais recente tenha acesso a todos os backups restauráveis.

O Amazon Keyspaces deve ter acesso à sua chave gerenciada pelo cliente para fornecer acesso aos dados da sua tabela. Se o estado da chave de criptografia estiver definido como desativado ou programado para exclusão, o Amazon Keyspaces não conseguirá criptografar ou descriptografar dados. Como resultado, você não consegue realizar operações de leitura e gravação na tabela. Assim que o serviço detectar que a chave de criptografia está inacessível, o Amazon Keyspaces enviará uma notificação por e-mail para alertar você.

Você deve restaurar a acesso à sua chave de criptografia dentro de sete dias ou o Amazon Keyspaces excluirá sua tabela automaticamente. Como precaução, o Amazon Keyspaces cria um backup restaurável dos dados da tabela antes de excluí-la. O Amazon Keyspaces mantém o backup restaurável por 35 dias. Depois de 35 dias, você não poderá mais restaurar os dados da tabela. Você não é cobrado pelo backup restaurável, mas cobranças de restauração padrão se aplicam.

É possível usar esse backup restaurável e restaurar seus dados para uma nova tabela. Para iniciar a restauração, a última chave gerenciada pelo cliente usada na tabela deve estar habilitada e o Amazon Keyspaces deve ter acesso a ela.

> **Note**
>
> Quando vovê cria uma tabela criptografada usando uma chave gerenciada pelo cliente que está inacessível ou programada para ser excluída antes da conclusão do processo de criação, ocorre um erro. A operação de criação de tabela falha e você recebe uma notificação por e-mail.

Notas de uso de criptografia em repouso

Considere o seguinte ao usar criptografia em repouso no Amazon Keyspaces.

- A criptografia do lado do servidor em repouso está habilitada em todas as tabelas do Amazon Keyspaces e não pode ser desabilitada. A tabela inteira é criptografada em repouso, você não pode selecionar colunas ou linhas específicas para criptografia.

- Por padrão, o Amazon Keyspaces usa uma chave padrão de serviço único (Chave pertencente à AWS) para criptografar todas as suas tabelas. Se essa chave não existir, ela é criada para você. As chaves padrão do serviço não podem ser desabilitadas.
- A criptografia em repouso só criptografa dados enquanto eles estão estáticos (em repouso) em uma mídia de armazenamento persistente. Se a segurança dos dados for motivo de preocupação para dados em trânsito ou dados em uso, será necessário tomar outras medidas:
  - Dados em trânsito: todos os dados no Amazon Keyspaces são criptografados em trânsito. Por padrão, as comunicações com o Amazon Keyspaces são protegidas com a criptografia Secure Sockets Layer (SSL) /Transport Layer Security (TLS).
  - Dados em uso: proteja seus dados antes de enviá-los ao Amazon Keyspaces usando a criptografia do lado do cliente.
  - Chaves gerenciadas pelo cliente: os dados em repouso em suas tabelas são sempre criptografados usando suas chaves gerenciadas pelo cliente. No entanto, as operações que realizam atualizações atômicas de várias linhas criptografam os dados temporariamente usando Chaves pertencentes à AWS durante o processamento. Isso inclui operações de exclusão de intervalos e operações que acessam simultaneamente dados estáticos e não estáticos.
- Uma única chave gerenciada pelo cliente pode ter até 50.000 concessões. Cada tabela do Amazon Keyspaces associada a uma chave gerenciada pelo cliente consome 2 concessões. Uma concessão é liberada quando a tabela é excluída. A segunda concessão é usada para criar um snapshot automático da tabela para proteger contra perda de dados caso o Amazon Keyspaces perca o acesso à chave gerenciada pelo cliente acidentalmente. Essa concessão é liberada 42 dias após a exclusão da tabela.

## Criptografia em repouso: como usar chaves gerenciadas pelo cliente para criptografar tabelas no Amazon Keyspaces

É possível usar o console ou a instruções CQL para especificar a AWS KMS key em novas tabelas e atualizar as chaves de criptografia em tabelas existentes no Amazon Keyspaces. O tópico a seguir descreve como implementar chaves gerenciadas pelo cliente para tabelas novas e existentes.

Tópicos

- Pré-requisitos: Crie uma chave gerenciada pelo cliente usando AWS KMS e conceda permissões ao Amazon Keyspaces
- Etapa 3: Especificar uma chave gerenciada pelo cliente para uma nova tabela
- Etapa 4: Atualizar a chave de criptografia de uma tabela existente

- Etapa 5: Usar o contexto de criptografia do Amazon Keyspaces nos logs
- Etapa 6: Configurar o monitoramento com AWS CloudTrail

Pré-requisitos: Crie uma chave gerenciada pelo cliente usando AWS KMS e conceda permissões ao Amazon Keyspaces

Antes de proteger uma tabela do Amazon Keyspaces com uma chave gerenciada pelo cliente, você deve primeiro criar a chave em AWS Key Management Service (AWS KMS) e depois autorizar o Amazon Keyspaces a usar essa chave.

Etapa 1: Criar uma chave gerenciada pelo cliente do AWS KMS

Para criar uma chave gerenciada pelo cliente para ser usada para proteger uma tabela do Amazon Keyspaces, você pode seguir as etapas em Criação de chaves KMS de criptografia simétrica usando o console ou a API AWS.

Etapa 2: Autorizar o uso da chave gerenciada pelo cliente

Antes de escolher uma chave gerenciada pelo cliente para proteger uma tabela do Amazon Keyspaces, as políticas nessa chave gerenciada pelo cliente devem conceder ao Amazon Keyspaces permissão para usá-la em seu nome. Você tem controle total sobre as políticas e concessões em uma chave gerenciada pelo cliente. É possível fornecer essas permissões em uma política de chaves, em uma política do IAM ou em uma concessão.

O Amazon Keyspaces não precisa de autorização adicional para usar o padrão Chave pertencente à AWS para proteger as tabelas do Amazon Keyspaces em sua conta AWS.

Os tópicos a seguir mostram como configurar as permissões necessárias usando concessões e políticas do IAM que permitem que as tabelas do Amazon Keyspaces usem uma chave gerenciada pelo cliente.

Tópicos

- Política de chaves para chaves gerenciadas pelo cliente
- Política de chaves de exemplo
- Como usar concessões para autorizar o Amazon Keyspaces

### Política de chaves para chaves gerenciadas pelo cliente

Ao escolher uma chave gerenciada pelo cliente para proteger uma tabela do Amazon Keyspaces, o Amazon Keyspaces obtém permissão para usar a chave gerenciada pelo cliente em nome da entidade principal que faz a seleção. Essa entidade principal, um usuário ou um perfil deve ter as permissões na chave gerenciada pelo cliente exigida pelo Amazon Keyspaces.

No mínimo, o Amazon Keyspaces exige as seguintes permissões em uma chave gerenciada pelo cliente:

- kms:Encrypt
- kms:Decrypt
- kms:ReEncrypt* (para kms:ReEncryptFrom e kms:ReEncryptTo)
- kms:GenerateDataKey* (para kms:GenerateDataKey e kms:GenerateDataKeyWithoutPlaintext)
- kms:DescribeKey
- kms:CreateGrant

### Política de chaves de exemplo

Por exemplo, a política de chaves de exemplo a seguir fornece somente as permissões necessárias. A política tem os seguintes efeitos:

- Permite que o Amazon Keyspaces use a chave gerenciada pelo cliente em operações criptográficas e cria concessões, mas somente quando está atuando em nome de entidades principais na conta que tem permissão para usar o Amazon Keyspaces. Se as entidades principais especificados na instrução da política não tiverem permissão para usar o Amazon Keyspaces, a chamada falhará, mesmo se vier do serviço do Amazon Keyspaces.
- A chave de condição kms:ViaService concede as permissões somente quando a solicitação é proveniente do Amazon Keyspaces em nome das entidades principais listadas na instrução da política. Essas entidades principais não podem chamar essas operações diretamente. Observe que o valor de `kms:ViaService`, `cassandra.*.amazonaws.com`, tem um asterisco (*) na posição da região. O Amazon Keyspaces exige a permissão para ser independente de qualquer Região da AWS específica.
- Concede aos administradores da chave gerenciada pelo cliente (usuários que podem assumir o perfil `db-team`) acesso somente leitura à chave gerenciada pelo cliente e permissão para revogar concessões, incluindo as concessões exigidas pelo Amazon Keyspaces para proteger a tabela.

- Concede ao Amazon Keyspaces acesso somente leitura à chave gerenciada pelo cliente. Nesse caso, o Amazon Keyspaces pode chamar essas operações diretamente. Ele não precisa atuar em nome da entidade principal de uma conta.

Antes de usar uma política de chaves de exemplo, substitua o exemplo de entidades principais por entidades principais reais da sua conta da Conta da 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": "*"
    }
  ]
}
```

Como usar concessões para autorizar o Amazon Keyspaces

Além de políticas de chaves, o Amazon Keyspaces usa concessões para definir permissões em uma chave gerenciada pelo cliente. Para visualizar as concessões em uma chave gerenciada pelo cliente na sua conta, use a operação ListGrants. O Amazon Keyspaces não precisa de concessões ou permissões adicionais para usar a Chave pertencente à AWS para proteger sua tabela.

O Amazon Keyspaces usa as permissões de concessão ao executar manutenção do sistema e tarefas de proteção de dados contínua em segundo plano. Usa também concessões para gerar chaves de tabela.

Cada concessão é específica a uma tabela. Se a conta inclui várias tabelas criptografadas na mesma chave gerenciada pelo cliente, há uma concessão de cada tipo para cada tabela. A concessão é restrita pelo contexto da criptografia do Amazon Keyspaces, que inclui o nome da tabela e o ID da Conta da AWS. A concessão inclui permissão para retirar a concessão se ela não for mais necessária.

Para criar as concessões, o Amazon Keyspaces deve ter permissão para chamar `CreateGrant` em nome do usuário que criou a tabela criptografada.

A política de chaves também pode permitir que a conta revogue a concessão na chave gerenciada pelo cliente. No entanto, se você revogar a concessão em uma tabela criptografada ativa, o Amazon Keyspaces não poderá proteger e manter a tabela.

Etapa 3: Especificar uma chave gerenciada pelo cliente para uma nova tabela

Siga estas etapas para especificar a chave gerenciada pelo cliente em uma nova tabela usando o console do Amazon Keyspaces ou o CQL.

Crie uma tabela criptografada usando uma chave gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas) e Create table (Criar tabela).

3. Na página Criar tabela, na seção Detalhes da tabela, selecione um espaço de chaves e forneça um nome para a nova tabela.
4. Na seção Esquema, crie o esquema para sua tabela.
5. Na seção Configurações da tabela, selecione Personalizar configurações.
6. Continue com as configurações de criptografia.

   Nesta etapa, você seleciona as configurações de criptografia para a tabela.

   Na seção Criptografia em repouso, em Escolha uma AWS KMS key, escolha a opção Escolha uma chave KMS diferente (avançado) e, no campo de pesquisa, escolha AWS KMS key ou insira um Nome do recurso da Amazon (ARN).

   > **Note**
   >
   > Se a chave selecionada não estiver acessível ou não tiver as permissões necessárias, consulte Solução de problemas de acesso à chave no Guia do desenvolvedor AWS Key Management Service.

7. Selecione Create (Criar) para criar a tabela criptografada.

Crie uma nova tabela usando uma chave gerenciada pelo cliente para criptografia em repouso (CQL)

Para criar uma nova tabela que usa uma chave gerenciada pelo cliente para criptografia em repouso, você pode usar a instrução `CREATE TABLE` como no exemplo a seguir. Certifique-se de substituir o ARN da chave por um ARN para uma chave válida com permissões concedidas ao 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'
            }
    };
```

Se você receber uma `Invalid Request Exception`, precisará confirmar que a chave gerenciada pelo cliente é válida e que o Amazon Keyspaces tem as permissões necessárias. Para confirmar se

a chave foi configurada corretamente, consulte Solução de problemas de acesso à chave no Guia do desenvolvedor AWS Key Management Service.

Etapa 4: Atualizar a chave de criptografia de uma tabela existente

Também é possível usar o console do Amazon Keyspaces ou CQL para atualizar as chaves de criptografia de uma tabela existente entre uma Chave pertencente à AWS e uma chave gerenciada pelo cliente a qualquer momento.

Atualizar uma tabela existente com a nova chave gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. No painel de navegação, selecione Tables (Tabelas).
3. Escolha a tabela com a qual você deseja trabalhar e selecione a guia Configurações adicionais.
4. Na seção Criptografia em repouso, escolha Gerenciar criptografia para editar as configurações de criptografia da tabela.

   Em Escolher uma AWS KMS key, escolha a opção Escolha uma chave KMS diferente (avançado) e, no campo de pesquisa, escolha uma AWS KMS key ou insira um Nome do recurso da Amazon (ARN).

   > **Note**
   >
   > Se a chave selecionada não for válida, consulte Solução de problemas de acesso à chave no Guia do desenvolvedor AWS Key Management Service.

   Como alternativa, você pode escolher uma Chave pertencente à AWS para uma tabela criptografada com uma chave gerenciada pelo cliente.
5. Selecione Save (Salvar) para salvar as alterações.

Atualizar a chave de criptografia usada para uma tabela existente

Para alterar a chave de criptografia de uma tabela existente, use a instrução `ALTER TABLE` para especificar uma chave gerenciada pelo cliente para criptografia em repouso. Certifique-se de substituir o ARN da chave por um ARN para uma chave válida com permissões concedidas ao 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'
                   }
         };
```

Se você receber uma `Invalid Request Exception`, precisará confirmar que a chave gerenciada pelo cliente é válida e que o Amazon Keyspaces tem as permissões necessárias. Para confirmar se a chave foi configurada corretamente, consulte Solução de problemas de acesso à chave no Guia do desenvolvedor AWS Key Management Service.

Para alterar a chave de criptografia de volta para a opção padrão de criptografia em repouso com Chaves pertencentes à AWS, você pode usar a instrução `ALTER TABLE` conforme mostrado no exemplo a seguir.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
                'encryption_specification':{
                     'encryption_type' : 'AWS_OWNED_KMS_KEY'
                   }
         };
```

Etapa 5: Usar o contexto de criptografia do Amazon Keyspaces nos logs

Um contexto de criptografia é um conjunto de pares de chave-valor que contêm dados arbitrários não secretos. Quando você inclui um contexto de criptografia em uma solicitação para criptografar dados, o AWS KMS vincula de forma criptográfica o contexto de criptografia aos dados criptografados. Para descriptografar os dados, você deve passar o mesmo contexto de criptografia.

O Amazon Keyspaces usa o mesmo contexto de criptografia em todas as operações de criptografia do AWS KMS. Se você usar uma chave gerenciada pelo cliente para proteger a tabela do Amazon Keyspaces, é possível usar o contexto de criptografia para identificar o uso da chave gerenciada pelo cliente em logs e registros de auditoria. Ele também aparece em texto simples em logs, como aqueles para AWS CloudTrail e Amazon CloudWatch Logs.

Em suas solicitações para o AWS KMS, o Amazon Keyspaces usa um contexto de criptografia com dois pares de chave/valor.

```
"encryptionContextSubset": {
```

```
    "aws:cassandra:keyspaceName": "my_keyspace",
    "aws:cassandra:tableName": "mytable"
    "aws:cassandra:subscriberId": "111122223333"
}
```

- Espaço de chaves: O primeiro par de chave-valor identifica o espaço de chaves que inclui a tabela que o Amazon Keyspaces está criptografando. A chave é `aws:cassandra:keyspaceName`. O valor é o nome do espaço de chaves.

  ```
  "aws:cassandra:keyspaceName": "<keyspace-name>"
  ```

  Por exemplo:

  ```
  "aws:cassandra:keyspaceName": "my_keyspace"
  ```

- Tabela: O segundo par de chave/valor identifica a tabela que o Amazon Keyspaces está criptografando. A chave é `aws:cassandra:tableName`. O valor é o nome da tabela.

  ```
  "aws:cassandra:tableName": "<table-name>"
  ```

  Por exemplo:

  ```
  "aws:cassandra:tableName": "my_table"
  ```

- Conta: O terceiro par de chave-valor identifica a Conta da AWS. A chave é `aws:cassandra:subscriberId`. O valor é o ID de conta.

  ```
  "aws:cassandra:subscriberId": "<account-id>"
  ```

  Por exemplo:

  ```
  "aws:cassandra:subscriberId": "111122223333"
  ```

### Etapa 6: Configurar o monitoramento com AWS CloudTrail

Se você usa uma chave gerenciada pelo cliente para proteger suas tabelas do Amazon Keyspaces, você poderá usar logs do AWS CloudTrail para rastrear as solicitações que o Amazon Keyspaces envia ao AWS KMS por você.

As solicitações `GenerateDataKey`, `DescribeKey`, `Decrypt`, e `CreateGrant` são discutidas nesta seção. Além disso, o Amazon Keyspaces usa uma operação RetireGrant para remover uma concessão quando você exclui uma tabela.

GenerateDataKey

O Amazon Keyspaces cria uma chave de tabela exclusiva para criptografar dados em repouso. Ele envia uma solicitação GenerateDataKey ao AWS KMS que especifica a chave do KMS para a tabela.

O evento que registra a operação `GenerateDataKey` é semelhante ao evento de exemplo a seguir. O usuário é a conta de serviço do Amazon Keyspaces. Os parâmetros incluem o Nome do recurso da Amazon (ARN) da chave gerenciada pelo cliente, um especificador de chave que requer uma chave de 256 bits e o contexto de criptografia que identifica a tabela e a conta da Conta da 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

Além disso, o Amazon Keyspaces usa uma operação DescribeKey para determinar se a chave do KMS escolhida existe na conta e na região.

O evento que registra a operação `DescribeKey` é semelhante ao evento de exemplo a seguir. O usuário é a conta de serviço do Amazon Keyspaces. Os parâmetros incluem o ARN da chave gerenciada pelo cliente e um especificador de chaves que requer uma chave 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

Quando você acessa uma tabela criptografada do Amazon Keyspaces, o Amazon Keyspaces precisa descriptografar a chave da tabela para que possa descriptografar as chaves abaixo dela na hierarquia. Descriptografa os dados na tabela. Para descriptografar a chave de tabela. O Amazon Keyspaces envia uma solicitação Decrypt para o AWS KMS que especifica a chave KMS para a tabela.

O evento que registra a operação `Decrypt` é semelhante ao evento de exemplo a seguir. O usuário é a entidade principal na sua Conta da AWS que está acessando a tabela. Os parâmetros incluem a chave de tabela criptografada (como um blob de texto cifrado) e o contexto de criptografia que identifica a tabela e a Conta da AWS. O AWS KMS deriva o ID da chave gerenciada pelo cliente do texto cifrado.

```
{
    "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

Quando você usa uma chave gerenciada pelo cliente para proteger sua tabela do Amazon Keyspaces, o Amazon Keyspaces usa concessões para permitir que o serviço execute a proteção

de dados e tarefas de manutenção e durabilidade contínuas. Essas concessões não são necessárias em Chaves pertencentes à AWS.

As concessões que o Amazon Keyspaces cria são específicas a uma tabela. A entidade principal na solicitação CreateGrant é o usuário que criou a tabela.

O evento que registra a operação `CreateGrant` é semelhante ao evento de exemplo a seguir. Os parâmetros incluem o nome de recurso da Amazon (ARN) da chave gerenciada pelo cliente para a tabela, a entidade principal favorecida e a entidade principal que está sendo retirada (o serviço do Amazon Keyspaces) e as operações que a concessão abrange. Inclui também uma restrição que requer que todas as operações de criptografia usem o contexto de criptografia.

```
{
    "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"
}
```

## Criptografia em trânsito no Amazon Keyspaces

O Amazon Keyspaces só aceita conexões seguras usando Transport Layer Security (TLS). A criptografia em trânsito fornece uma camada adicional de proteção de dados ao criptografar

seus dados à medida que eles viajam de e para o Amazon Keyspaces. Políticas organizacionais, regulamentações setoriais ou governamentais e exigências de conformidade geralmente demandam o uso de criptografia em trânsito para aumentar a segurança de dados de seus aplicativos quando transmitem dados pela rede.

Para saber como criptografar conexões `cqlsh` com o Amazon Keyspaces usando TLS, consulte the section called “Como configurar manualmente as conexões `cqlsh` para o TLS”. Para aprender a usar a criptografia TLS com drivers de cliente, consulte the section called “Uso de um driver de cliente Cassandra”.

## Privacidade do tráfego entre redes no Amazon Keyspaces

Descreve como o Amazon Keyspaces (para Apache Cassandra) protege conexões de aplicativos on-premises com o Amazon Keyspaces e entre o Amazon Keyspaces e outros recursos da AWS dentro da mesma Região da AWS.

### Tráfego entre clientes de serviço e on-premises e as aplicações

Você tem duas opções de conectividade entre sua rede privada e a AWS:

- Uma conexão do AWS Site-to-Site VPN. Para obter mais informações, consulte O que é o AWS Site-to-Site VPN? no  Guia do usuário do AWS Site-to-Site VPN.
- Uma conexão do AWS Direct Connect. Para obter mais informações, consulte O que é o AWS Direct Connect? no  Guia do usuário do AWS Direct Connect.

Como um serviço gerenciado, o Amazon Keyspaces (para Apache Cassandra) é protegido pela segurança da rede global da AWS. Para obter informações sobre serviços de segurança da AWS e como a AWS protege a infraestrutura, consulte Segurança na Nuvem AWS. Para projetar seu ambiente da AWS usando as práticas recomendadas de segurança de infraestrutura, consulte Proteção de infraestrutura em Pilar segurança: AWS Well-Architected Framework.

Você usa as chamadas de API publicadas da AWS para acessar o Amazon Keyspaces por meio da rede. Os clientes devem oferecer suporte para:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

O Amazon Keyspaces oferece suporte a dois métodos de autenticação de solicitações de clientes. O primeiro método usa credenciais específicas do serviço, que são credenciais baseadas em senha geradas para um usuário do IAM específico. Você pode criar e gerenciar a senha usando o console do IAM, o AWS CLI ou o API da AWS. Para obter mais informações, consulte Como usar o IAM com o Amazon Keyspaces.

O segundo método usa um complemento de autenticação para o driver Java DataStax de código aberto para o Cassandra. Esse plug-in permite que usuários, perfis e identidades federadas do IAM adicionem informações de autenticação às solicitações de API do Amazon Keyspaces (para Apache Cassandra) usando o processo AWSProcesso do Signature Version 4 (SigV4). Para obter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

### Tráfego entre recursos da AWS na mesma região

Os endpoints da VPC de interface permitem a comunicação privada entre a nuvem privada virtual (VPC) em execução no Amazon VPC e no Amazon Keyspaces. Os endpoints da VPC de interface são fornecidos pelo AWS PrivateLink, um serviço da AWS permite a comunicação privada entre os serviços da VPC e os serviços da AWS. O AWS PrivateLink permite isso usando uma interface de rede elástica com IPs privados em sua VPC para que o tráfego de rede não saia da rede Amazon. Os endpoints da VPC de interface não exigem um gateway da Internet, dispositivo NAT, conexão VPN ou conexão do AWS Direct Connect. Para obter mais informações, consulte Amazon Virtual Private Cloud e Endpoints da VPC de interface (AWS PrivateLink). Para obter exemplos de políticas, consulte the section called “Como usar o endpoint da VPC para o Amazon Keyspaces”.

## AWS Identity and Access Management para Amazon Keyspaces

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (ter permissões) para usar os recursos do Amazon Keyspaces. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

- Público
- Autenticando com identidades
- Gerenciando acesso usando políticas
- Como o Amazon Keyspaces funciona com o IAM
- Exemplos de políticas baseadas em identidade do Amazon Keyspaces
- Políticas gerenciadas pela AWS para o Amazon Keyspaces
- Solução de problemas de identidade e acesso do Amazon Keyspaces
- Uso de perfis vinculados ao serviço para o Amazon Keyspaces


## Público

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz no Amazon Keyspaces.

Usuário do serviço – se você usar o serviço do Amazon Keyspaces para fazer seu trabalho, o administrador fornecerá as credenciais e as permissões necessárias. À medida que mais atributos do Amazon Keyspaces forem usados para realizar o trabalho, talvez sejam necessárias permissões adicionais. Entender como o acesso é gerenciado pode ajudar você a solicitar as permissões corretas ao seu administrador. Se você não puder acessar um atributo no Amazon Keyspaces, consulte Solução de problemas de identidade e acesso do Amazon Keyspaces.

Administrador do serviço: se você for o responsável pelos recursos do Amazon Keyspaces em sua empresa, provavelmente terá acesso total a ele. Cabe a você determinar quais funcionalidades e atributos do Amazon Keyspaces os usuários do seu serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais sobre como a empresa pode usar o IAM com o Amazon Keyspaces, consulte Como o Amazon Keyspaces funciona com o IAM.

Administrador do IAM: se você for um administrador do IAM, talvez deseje saber detalhes sobre como escrever políticas para gerenciar o acesso ao Amazon Keyspaces. Para visualizar exemplos de políticas baseadas em identidade do Amazon Keyspaces que podem ser usadas no IAM, consulte Exemplos de políticas baseadas em identidade do Amazon Keyspaces.

# Autenticando com identidades

A autenticação é como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login em AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação multifator no Guia AWS IAM Identity Center do usuário e Utilizar a autenticação multifator (MFA) na AWS no Guia do usuário do IAM.

## Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do Usuário do IAM.

## Usuários e grupos do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do Usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de um perfil) no Guia do usuário do IAM.

## Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para o uso de perfis, consulte Utilizar perfis do IAM no Guia do usuário do IAM.

Funções do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do Usuário do IAM. Se você usar o Centro de identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a

autenticação, o Centro de identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de permissões no Guia do usuário AWS IAM Identity Center .

- Permissões temporárias para usuários do IAM –- um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas –- é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre funções e políticas baseadas em recursos para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado a serviço.
  - Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
  - Função de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
  - Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para funções vinculadas ao serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Utilizar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar perfis do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) no Guia do usuário do IAM.

## Gerenciando acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do Usuário do IAM.

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissões para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem presumir os perfis.

As políticas do IAM definem permissões para uma ação independente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

### Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas políticas

controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criando políticas do IAM no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda adicionalmente como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do Usuário do IAM.

### Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

### Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

### Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um atributo avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade com seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do Usuário do IAM.
- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em. AWS Organizations AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para obter mais informações sobre o Organizações e SCPs, consulte How SCPs work (Como os SCPs funcionam) no Guia do usuário do AWS Organizations .
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em atributo. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Guia do Usuário do IAM.

### Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

## Como o Amazon Keyspaces funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon Keyspaces, você deve entender quais recursos do IAM estão disponíveis para ser usado com ele. Para ter uma visão de alto nível de como

o Amazon Keyspaces e AWS outros serviços funcionam com o IAM, AWS consulte os serviços que funcionam com o IAM no Guia do usuário do IAM.

Tópicos

- Políticas baseadas em identidade do Amazon Keyspaces
- Políticas baseadas em recursos do Amazon Keyspaces
- Autorização baseada em tags do Amazon Keyspaces
- Perfis do IAM no Amazon Keyspaces

## Políticas baseadas em identidade do Amazon Keyspaces

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. O Amazon Keyspaces oferece suporte a ações e recursos específicos e chaves de condição. Para saber mais sobre todos os elementos usados em uma política JSON, consulte Referência de elementos de política JSON do IAM no Guia do usuário do IAM.

Para ver os recursos e as ações específicas do serviço Amazon Keyspaces e as chaves de contexto condicionais que podem ser usadas para políticas de permissões do IAM, consulte as Ações, recursos e chaves de condição do Amazon Keyspaces (para Apache Cassandra) na Referência de autorização de serviço.

Ações

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

As ações de política no Amazon Keyspaces usam o seguinte prefixo antes da ação: `cassandra:`. Por exemplo, para conceder a uma pessoa permissão para criar um espaço de chaves do

Amazon Keyspaces com a instrução CREATE CQL do Amazon Keyspaces, inclua a ação `cassandra:Create` na política. As instruções de política devem incluir um elemento `Action` ou `NotAction`. O Amazon Keyspaces define seu próprio conjunto de ações que descrevem as tarefas que você pode executar com esse serviço.

Para especificar várias ações em uma única instrução, separe-as com vírgulas, como segue:

```
"Action": [
      "cassandra:CREATE",
      "cassandra:MODIFY"
          ]
```

Para ver uma lista de ações do Amazon Keyspaces, consulte Ações definidas pelo Amazon Keyspaces (para Apache Cassandra) na Referência de autorização do serviço.

Recursos

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

No Amazon Keyspaces, os espaços de chaves e as tabelas podem ser usados no elemento `Resource` de permissões do IAM.

O recurso de espaço de chaves do Amazon Keyspaces tem o seguinte ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/
```

O recurso de tabela do Amazon Keyspaces tem o seguinte ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/table/
${tableName}
```

Para obter mais informações sobre o formato dos ARNs, consulte Amazon Resource Names (ARNs) e namespaces AWS de serviços.

Por exemplo, para especificar o espaço de chaves `mykeyspace` em sua instrução, use o seguinte ARN:

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/mykeyspace/"
```

Para especificar todos os espaços de chaves que pertencem a uma conta específica, use o curinga (*):

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/*"
```

Algumas ações do Amazon Keyspaces, como as de criação de recursos, não podem ser executadas em um recurso específico. Nesses casos, você deve utilizar o caractere curinga (*).

```
"Resource": "*"
```

Para se conectar programaticamente ao Amazon Keyspaces com um driver padrão, a entidade principal deve ter acesso SELECT às tabelas do sistema, pois a maioria dos drivers lê os espaços de chaves/tabelas do sistema na conexão. Por exemplo, para conceder permissões `SELECT` a um usuário do IAM para `mytable` em `mykeyspace`, a entidade principal deve ter permissões para ler ambos, `mytable` e `system keyspace`. Para especificar vários recursos em uma única instrução, separe os ARNs com vírgulas.

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
```

Para ver uma lista de tipos de recursos do Amazon Keyspaces e seus ARNs, consulte Recursos definidos pelo Amazon Keyspaces (para Apache Cassandra) na Referência de autorização do serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo Amazon Keyspaces (para Apache Cassandra).

Chaves de condição

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único `Condition` elemento, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um atributo somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos da política do IAM: variáveis e tags no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

O Amazon Keyspaces define seu próprio conjunto de chaves de condição e também oferece suporte ao uso de algumas chaves de condição globais. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

Todas as ações do Amazon Keyspaces oferecem suporte às chaves de condição `aws:RequestTag/${TagKey}`, `aws:ResourceTag/${TagKey}` e `aws:TagKeys`. Para ter mais informações, consulte the section called “ Acesso a recursos do Amazon Keyspaces com base em tags”.

Para ver uma lista de chaves de condição do Amazon Keyspaces, consulte Chaves de condição do Amazon Keyspaces (para Apache Cassandra) na Referência de autorização do serviço. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte Ações definidas pelo Amazon Keyspaces (para Apache Cassandra).

Exemplos

Para visualizar exemplos de políticas baseadas em identidade do Amazon Keyspaces, consulte Exemplos de políticas baseadas em identidade do Amazon Keyspaces.

## Políticas baseadas em recursos do Amazon Keyspaces

O Amazon Keyspaces não oferece suporte a políticas baseadas em recursos. Para visualizar um exemplo de uma política baseada em recurso detalhada, consulte https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

## Autorização baseada em tags do Amazon Keyspaces

Você pode gerenciar o acesso a seus recursos do Amazon Keyspaces usando tags. Para gerenciar o acesso a recursos baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as chaves de condição `cassandra:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Para obter mais informações sobre recursos de marcação do Amazon Keyspaces, consulte the section called “Trabalhando com tags”.

Para visualizar exemplos de políticas baseadas em identidade para limitar o acesso a um recurso baseado em tags desse recurso, consulte Acesso a recursos do Amazon Keyspaces com base em tags.

## Perfis do IAM no Amazon Keyspaces

Uma função do IAM é uma entidade dentro da sua Conta da AWS que tem permissões específicas.

Uso de credenciais temporárias com o Amazon Keyspaces

É possível usar credenciais temporárias para fazer login com federação, assumir uma função do IAM ou assumir uma função entre contas. Você obtém credenciais de segurança temporárias chamando operações de AWS STS API, como AssumeRoleou GetFederationToken.

O Amazon Keyspaces oferece suporte ao uso de credenciais temporárias com o plug-in de autenticação AWS Signature Version 4 (SigV4) disponível no repositório Github para os seguintes idiomas:

- 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.
- Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Para exemplos e tutoriais que implementam o plug-in de autenticação para acessar o Amazon Keyspaces programaticamente, consulte. the section called “Uso de um driver de cliente Cassandra”

Funções vinculadas a serviço

As funções vinculadas ao serviço permitem que AWS os serviços acessem recursos em outros serviços para concluir uma ação em seu nome. Os perfis vinculados a serviço aparecem em sua conta do IAM e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados a serviço.

Para obter detalhes sobre como criar ou gerenciar perfis vinculados ao serviço do Amazon Keyspaces, consulte the section called “Usar perfis vinculados ao serviço”.

Perfis de serviço

O Amazon Keyspaces não é compatível com perfis de serviço.

## Exemplos de políticas baseadas em identidade do Amazon Keyspaces

Por padrão, os usuários e os perfis do IAM não têm permissão para criar ou modificar recursos do Amazon Keyspaces. Eles também não podem realizar tarefas usando o console, o CQLSH ou AWS a AWS CLI API. Um administrador do IAM deve criar políticas do IAM que concedam aos usuários e perfis permissão para executarem operações de API específicas nos recursos especificados de que precisam. O administrador deve anexar essas políticas aos usuários ou grupos do IAM que exigem essas permissões.

Para saber como criar uma política baseada em identidade do IAM utilizando esses exemplos de documentos de política JSON, consulte Criar políticas na guia JSON no Guia do usuário do IAM.

Tópicos

- Melhores práticas de política
- Uso do console do Amazon Keyspaces
- Permitir que usuários visualizem suas próprias permissões
- Como acessar as tabelas do Amazon Keyspaces

- Acesso a recursos do Amazon Keyspaces com base em tags

## Melhores práticas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do Amazon Keyspaces em sua conta. Essas ações podem incorrer em custos para seus Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do Usuário do IAM.
- Aplique permissões de privilégio mínimo –- ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em atributos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do Usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso –- você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode gravar uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: Condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais –- o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudá-lo a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando

as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas Recomendadas de Segurança no IAM no Guia do Usuário do IAM.

### Uso do console do Amazon Keyspaces

O Amazon Keyspaces não exige permissões específicas para acessar o console do Amazon Keyspaces. Você precisa de pelo menos permissões de somente leitura para listar e visualizar detalhes sobre os recursos do Amazon Keyspaces em seu. Conta da AWS Se você criar uma política baseada em identidade que seja mais restritiva que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis do IAM) com essa política.

Duas políticas AWS gerenciadas estão disponíveis para as entidades acessarem o console do Amazon Keyspaces.

- AmazonKeyspacesReadOnlyAccess_v2 — Essa política concede acesso somente de leitura ao Amazon Keyspaces.
- AmazonKeyspacesFullAccess— Essa política concede permissões para usar o Amazon Keyspaces com acesso total a todos os recursos.

Para obter mais informações sobre políticas gerenciadas no Amazon Keyspaces, consulte the section called “Políticas gerenciadas pela AWS”.

### Permitir que usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou 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": "*"
        }
    ]
}
```

## Como acessar as tabelas do Amazon Keyspaces

Veja a seguir um exemplo de política que concede acesso somente de leitura (`SELECT`) às tabelas do sistema Amazon Keyspaces. Para todas as amostras, substitua o ID da região e da conta no Amazon Resource Name (ARN) pelo seu.

> **Note**
>
> Para se conectar com um driver padrão, um usuário deve ter pelo menos acesso `SELECT` às tabelas do sistema, porque a maioria dos drivers lê os espaços de chaves/as tabelas do sistema na conexão.

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

```
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

O exemplo de política a seguir adiciona acesso somente leitura à tabela do usuário `mytable` no 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*"
         ]
      }
   ]
}
```

O exemplo de política a seguir atribui acesso de leitura/gravação a uma tabela de usuários e acesso de leitura às tabelas do sistema.

> **Note**
>
> As tabelas do sistema são sempre somente para leitura.

```
{
```

```
   "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*"
         ]
      }
   ]
}
```

O exemplo de política a seguir permite que um usuário crie tabelas no espaço de chaves `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*"
         ]
      }
   ]
}
```

## Acesso a recursos do Amazon Keyspaces com base em tags

Você pode usar condições em sua política baseada em identidade para controlar o acesso aos recursos do Amazon Keyspaces com base em etiquetas. Essas políticas controlam a visibilidade dos espaços de chaves e das tabelas na conta. Observe que as permissões baseadas em tags para

tabelas do sistema se comportam de forma diferente quando as solicitações são feitas usando o AWS SDK em comparação com as chamadas da API Cassandra Query Language (CQL) por meio de drivers e ferramentas de desenvolvedor do Cassandra.

- Para fazer solicitações de recursos `List` e `Get` com o SDK da AWS ao usar o acesso baseado em tags, o chamador precisa ter acesso de leitura às tabelas do sistema. Por exemplo, as permissões de ação `Select` são necessárias para ler dados das tabelas do sistema por meio da operação `GetTable`. Se o chamador tiver apenas acesso baseado em tags a uma tabela específica, uma operação que exija acesso adicional a uma tabela do sistema falhará.
- Para compatibilidade com o comportamento estabelecido do driver Cassandra, as políticas de autorização baseadas em tags não são aplicadas ao realizar operações em tabelas do sistema usando chamadas de API do Cassandra Query Language (CQL) por meio de drivers e ferramentas de desenvolvedor do Cassandra.

O exemplo a seguir mostra como você pode criar uma política que conceda permissões a um usuário para visualizar uma tabela se o `Owner` da tabela contiver o valor do nome do usuário em questão. Neste exemplo, você também concede acesso de leitura às tabelas do sistema.

```
{
   "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}"
            }
         }
      }
   ]
}
```

É possível anexar essa política aos usuários do IAM na sua conta. Se um usuário chamado `richard-roe` tentar visualizar uma tabela do Amazon Keyspaces, a tabela deverá estar marcada como `0wner=richard-roe` ou `owner=richard-roe`. Caso contrário, ele terá o acesso negado. A chave da tag de condição `0wner` corresponde a `0wner` e a `owner` porque os nomes das chaves de condição não fazem distinção entre maiúsculas e minúsculas. Para obter mais informações, consulte Elementos de política JSON do IAM: condição no Guia do usuário do IAM.

A política a seguir concede permissões a um usuário para criar tabelas com tags se o `0wner` da tabela contiver o valor do nome de usuário desse usuário.

```
{
    "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}"
            }
         }
      }
   ]
}
```

## Políticas gerenciadas pela AWS para o Amazon Keyspaces

Uma política gerenciada pela AWS é uma política independente criada e administrada pela AWS. As políticas gerenciadas pela AWS são criadas para fornecer permissões a vários casos de uso comuns a fim de que você possa começar a atribuir permissões a usuários, grupos e perfis.

Lembre-se de que as políticas gerenciadas pela AWS podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para todos os clientes da

AWS usarem. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da  específicas para seus casos de uso.

Você não pode alterar as permissões definidas em políticas gerenciadas pela AWS. Se a AWS atualiza as permissões definidas em um política gerenciada pela AWS, a atualização afeta todas as identidades de entidades principais (usuários, grupos e perfis) às quais a política está vinculada. É mais provável que a AWS atualize uma política gerenciada pela AWS quando um novo AWS service (Serviço da AWS) é lançado ou novas operações de API são disponibilizadas para os serviços existentes.

Para obter mais informações, consulte Políticas gerenciadas pela AWS no Manual do usuário do IAM.

## política gerenciada pela AWS: AmazonKeyspacesReadOnlyAccess_v2

É possível anexar a política `AmazonKeyspacesReadOnlyAccess_v2` a suas identidades do IAM.

Esta política concede acesso somente leitura ao Amazon Keyspaces e inclui as permissões necessárias ao se conectar por meio de endpoints da VPC privados.

Detalhes da permissão

Esta política inclui as seguintes permissões.

- `Amazon Keyspaces`: fornece acesso somente leitura ao Amazon Keyspaces.
- `Application Auto Scaling` – Isso permite que as entidades principais visualizem configurações do Aplicativo de ajuste de escala automático. Isso é necessário para que os usuários possam visualizar as políticas de escalabilidade automática anexadas a uma tabela.
- `CloudWatch` – Permite que as entidades principais visualizem dados métricos e alarmes configurados no CloudWatch. Isso é necessário para que os usuários possam visualizar o tamanho da tabela faturável e os alarmes do CloudWatch que foram configurados para uma tabela.
- `AWS KMS` – Permite que as entidades principais visualizem as chaves configuradas no AWS KMS. Isso é necessário para que os usuários possam visualizar as chaves AWS KMS que eles criam

e gerenciam em suas contas para confirmar que a chave atribuída ao Amazon Keyspaces é uma chave de criptografia simétrica que está habilitada.

- `Amazon EC2`: Permite que as entidades principais que se conectam ao Amazon Keyspaces por meio de endpoints da VPC consultem a VPC em sua instância do Amazon EC2 para obter informações sobre endpoints e interfaces de rede. Esse acesso somente leitura à instância do Amazon EC2 é necessário para que o Amazon Keyspaces possa pesquisar e armazenar endpoints da VPC de interface disponíveis na tabela `system.peers` usada para balanceamento de carga de conexão.

Para revisar a política no formato `JSON`, consulte AmazonKeyspacesReadOnlyAccess_v2.

## Política gerenciada pela AWS: AmazonKeyspacesReadOnlyAccess

É possível anexar a política `AmazonKeyspacesReadOnlyAccess` a suas identidades do IAM.

Esta política concede acesso somente leitura ao Amazon Keyspaces.

Detalhes da permissão

Esta política inclui as seguintes permissões.

- `Amazon Keyspaces`: fornece acesso somente leitura ao Amazon Keyspaces.
- `Application Auto Scaling` – Isso permite que as entidades principais visualizem configurações do Aplicativo de ajuste de escala automático. Isso é necessário para que os usuários possam visualizar as políticas de escalabilidade automática anexadas a uma tabela.
- `CloudWatch` – Permite que as entidades principais visualizem dados métricos e alarmes configurados no CloudWatch. Isso é necessário para que os usuários possam visualizar o tamanho da tabela faturável e os alarmes do CloudWatch que foram configurados para uma tabela.
- `AWS KMS` – Permite que as entidades principais visualizem as chaves configuradas no AWS KMS. Isso é necessário para que os usuários possam visualizar as chaves AWS KMS que eles criam e gerenciam em suas contas para confirmar que a chave atribuída ao Amazon Keyspaces é uma chave de criptografia simétrica que está habilitada.

Para revisar a política no formato JSON, consulte AmazonKeyspacesReadOnlyAccess

## Política gerenciada pela AWS: AmazonKeyspacesFullAccess

É possível anexar a política AmazonKeyspacesFullAccess a suas identidades do IAM.

Esta política concede permissões administrativas que permitem aos administradores acesso irrestrito ao Amazon Keyspaces.

Detalhes da permissão

Esta política inclui as seguintes permissões.

- Amazon Keyspaces: Permite que as entidades entidade principais acessem qualquer recurso do Amazon Keyspaces e executem todas as ações.
- Application Auto Scaling: Permite que as entidades principais criem, visualizem e excluam políticas de escalabilidade automática para tabelas do Amazon Keyspaces. Isso é necessário para que os administradores possam gerenciar políticas de escalabilidade automática para tabelas do Amazon Keyspaces.
- CloudWatch: Permite que as entidades principais vejam o tamanho da tabela faturável, bem como criem, visualizem e excluam alarmes do CloudWatch para as políticas de escalabilidade automática do Amazon Keyspaces. Isso é necessário para que os administradores possam visualizar o tamanho da tabela faturável e criar um painel do CloudWatch.
- IAM: Permite que o Amazon Keyspaces crie perfis vinculados a serviços com o IAM automaticamente quando os seguintes atributos são ativados:
  - Application Auto Scaling: Quando um administrador habilita o Aplicativo de ajuste de escala automático para uma tabela, o Amazon Keyspaces cria um perfil vinculado ao serviço para realizar ações automáticas de escalabilidade em seu nome.
  - Amazon Keyspaces Multi-Region Replication – Quando um administrador cria um espaço de chaves de várias regiões, um perfil vinculado ao serviço é criado automaticamente para realizar a replicação de dados para as Regiões da AWS selecionadas em seu nome.

  Para obter mais informações sobre funções vinculadas ao serviço, consulte the section called “Usar perfis vinculados ao serviço”.

- `AWS KMS` – Permite que as entidades principais visualizem as chaves configuradas no AWS KMS. Isso é necessário para que os usuários possam visualizar as chaves AWS KMS que eles criam e gerenciam em suas contas para confirmar que a chave atribuída ao Amazon Keyspaces é uma chave de criptografia simétrica que está habilitada.
- `Amazon EC2`: Permite que as entidades principais que se conectam ao Amazon Keyspaces por meio de endpoints da VPC consultem a VPC em sua instância do Amazon EC2 para obter informações sobre endpoints e interfaces de rede. Esse acesso somente leitura à instância do Amazon EC2 é necessário para que o Amazon Keyspaces possa pesquisar e armazenar endpoints da VPC de interface disponíveis na tabela `system.peers` usada para balanceamento de carga de conexão.

Para revisar a política no formato `JSON`, consulte AmazonKeyspacesFullAccess.

## Atualizações do Amazon Keyspaces para políticas gerenciadas pela AWS

Visualize detalhes sobre atualizações em políticas gerenciadas pela AWS para o Amazon Keyspaces desde que esse serviço começou a rastrear essas alterações. Para obter alertas automáticos sobre alterações feitas nesta página, inscreva-se no feed RSS na página Histórico do documento.

| Alteração | Descrição | Data |
| --- | --- | --- |
| AmazonKeySpacesFullAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões somente leitura para clientes que se conectam a ele por meio de endpoints da VPC de interface para acessar a instância do Amazon EC2 e pesquisar informações de rede.<br><br>O Amazon Keyspaces armazena os endpoints da VPC de interface disponíveis | 3 de outubro de 2023 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| | na tabela `system.peers` para balanceamento de carga de conexão. Para obter mais informações, consulte the section called “Usar VPC endpoints da interface do ”. | |
| AmazonKeyspacesReadOnlyAccess_v2 – Nova política | O Amazon Keyspaces criou uma nova política para adicionar permissões somente leitura para clientes que se conectam a ele por meio de endpoints da VPC de interface para acessar a instância do Amazon EC2 e pesquisar informações de rede.<br><br>O Amazon Keyspaces armazena os endpoints da VPC de interface disponíveis na tabela `system.peers` para balanceamento de carga de conexão. Para obter mais informações, consulte the section called “Usar VPC endpoints da interface do ”. | 12 de setembro de 2023 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| AmazonKeySpacesFullAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões para permitir que o Amazon Keyspaces crie um perfil vinculado ao serviço quando um administrador cria um espaço de chaves de várias regiões.<br><br>O Amazon Keyspaces usa o perfil vinculado ao serviço para executar tarefas de replicação de dados em seu nome. Para obter mais informações, consulte the section called “Replicação multirregional”. | 5 de junho de 2023 |
| AmazonKeyspacesReadOnlyAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões para permitir que os usuários visualizem o tamanho faturável de uma tabela usando o CloudWatch.<br><br>O Amazon Keyspaces se integra ao Amazon CloudWatch para permitir que você monitore o tamanho da tabela faturável. Para obter mais informações, consulte the section called “Métricas e dimensões do Amazon Keyspaces”. | 7 de julho de 2022 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| AmazonKeyspacesFullAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões para permitir que os usuários visualizem o tamanho faturável de uma tabela usando o CloudWatch.<br><br>O Amazon Keyspaces se integra ao Amazon CloudWatch para permitir que você monitore o tamanho da tabela faturável. Para obter mais informações, consulte the section called “Métricas e dimensões do Amazon Keyspaces”. | 7 de julho de 2022 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| AmazonKeyspacesReadOnlyAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões para permitir que os usuários visualizem chaves AWS KMS que foram configuradas para criptografia do Amazon Keyspaces em repouso.<br><br>A criptografia em repouso do Amazon Keyspaces integra-se ao AWS KMS para proteger e gerenciar as chaves de criptografia usadas para criptografar dados em repouso. Para visualizar a chave AWS KMS configurada para o Amazon Keyspaces, permissões somente leitura foram adicionadas. | 1º de junho de 2021 |

| Alteração | Descrição | Data |
| --- | --- | --- |
| AmazonKeyspacesFullAccess – Atualização para uma política existente | O Amazon Keyspaces adicionou novas permissões para permitir que os usuários visualizem chaves AWS KMS que foram configuradas para criptografia do Amazon Keyspaces em repouso.<br><br>A criptografia em repouso do Amazon Keyspaces integra-se ao AWS KMS para proteger e gerenciar as chaves de criptografia usadas para criptografar dados em repouso. Para visualizar a chave AWS KMS configura da para o Amazon Keyspaces , permissões somente leitura foram adicionadas. | 1º de junho de 2021 |
| O Amazon Keyspaces passou a monitorar alterações | O Amazon Keyspaces passou a controlar as alterações para as políticas gerenciadas pela AWS. | 1º de junho de 2021 |

## Solução de problemas de identidade e acesso do Amazon Keyspaces

Use as informações a seguir para ajudar a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com o Amazon Keyspaces e o IAM.

Tópicos

- Não estou autorizado a executar uma ação no Amazon Keyspaces
- Eu modifiquei um usuário ou perfil do IAM e as alterações não entraram em vigor imediatamente

- Não consigo restaurar uma tabela usando a point-in-time recuperação do Amazon Keyspaces (PITR)
- Não estou autorizado a realizar iam: PassRole
- Sou administrador e quero permitir que outras pessoas tenham acesso ao Amazon Keyspaces
- Quero permitir que pessoas de fora da minha Conta da AWS acessem meus recursos do Amazon Keyspaces

## Não estou autorizado a executar uma ação no Amazon Keyspaces

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador para obter ajuda. O administrador é a pessoa que forneceu o seu nome de usuário e senha.

O exemplo de erro a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para visualizar detalhes sobre uma *`tabela`*, mas não tem as permissões `cassandra:`*`Select`* para a tabela.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cassandra:Select on resource: mytable
```

Neste caso, Mateo pede ao administrador para atualizar suas políticas para permitir a ele o acesso ao recurso *`mytable`* usando a ação `cassandra:`*`Select`*.

## Eu modifiquei um usuário ou perfil do IAM e as alterações não entraram em vigor imediatamente

As alterações na política do IAM podem levar até 10 minutos para entrar em vigor em aplicativos com conexões existentes e estabelecidas com o Amazon Keyspaces. As alterações na política do IAM entram em vigor imediatamente quando os aplicativos estabelecem uma nova conexão. Se você fez modificações em um usuário ou perfil do IAM existente e elas não tiveram efeito imediato, aguarde 10 minutos ou desconecte-se e reconecte-se ao Amazon Keyspaces.

## Não consigo restaurar uma tabela usando a point-in-time recuperação do Amazon Keyspaces (PITR)

Se você está tentando restaurar uma tabela do Amazon Keyspaces com point-in-time recuperação (PITR) e vê o processo de restauração começar, mas não ser concluído com sucesso, talvez você

não tenha configurado todas as permissões necessárias para o processo de restauração. Você deve entrar em contato com o administrador para obter assistência e pedir a essa pessoa para atualizar suas políticas para permitir que você restaure uma tabela no Amazon Keyspaces.

Além das permissões de usuário, o Amazon Keyspaces pode exigir permissões para realizar ações durante o processo de restauração em nome da sua entidade principal. Esse é o caso se a tabela for criptografada com uma chave gerenciada pelo cliente ou se você estiver usando políticas do IAM que restringem o tráfego de entrada. Por exemplo, se você estiver usando chaves de condição em sua política do IAM para restringir o tráfego de origem a endpoints ou intervalos de IP específicos, a operação de restauração falhará. Para permitir que o Amazon Keyspaces execute a operação de restauração da tabela em nome da sua entidade principal, você deve adicionar uma chave de condição `aws:ViaAWSService` global na política do IAM.

Para obter mais informações sobre as permissões para restaurar tabelas, consulte the section called “Restaurar permissões”.

## Não estou autorizado a realizar iam: PassRole

Caso receba uma mensagem de erro informando que você não tem autorização para executar a ação `iam:PassRole`, as políticas deverão ser atualizadas para permitir a transmissão de um perfil ao Amazon Keyspaces.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O erro exemplificado a seguir ocorre quando um usuário do IAM chamado `marymajor` tenta usar o console para executar uma ação no Amazon Keyspaces. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar o perfil para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

## Sou administrador e quero permitir que outras pessoas tenham acesso ao Amazon Keyspaces

Para permitir que outras pessoas acessem o Amazon Keyspaces, é necessário criar uma entidade do IAM (usuário ou função) para a pessoa ou a aplicação que precisa do acesso. Elas usarão as credenciais dessa entidade para acessar a AWS. Você deve anexar uma política à entidade que concede a elas as permissões corretas no Amazon Keyspaces.

Para começar a usar imediatamente, consulte Criar os primeiros usuário e grupo delegados pelo IAM no Guia do usuário do IAM.

## Quero permitir que pessoas de fora da minha Conta da AWS acessem meus recursos do Amazon Keyspaces

Você pode criar um perfil que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. Você pode especificar quem é confiável para assumir o perfil. Para serviços que oferecem compatibilidade com políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o Amazon Keyspaces é compatível com esses recursos, consulte Como o Amazon Keyspaces funciona com o IAM.
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre usar funções e políticas baseadas em recursos para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

# Uso de perfis vinculados ao serviço para o Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) usa perfis vinculados ao serviço (IAM) AWS Identity and Access Management. Um perfil vinculado ao serviço é um tipo exclusivo de perfil do IAM vinculado diretamente ao Amazon Keyspaces. Os perfis vinculados ao serviço são predefinidos pelo Amazon Keyspaces e incluem todas as permissões que o serviço requer para chamar outros produtos da AWS em seu nome.

Tópicos


- Usando funções para o ajuste de escala automático do aplicativo Amazon Keyspaces
- Uso de perfis para replicação multirregional do Amazon Keyspaces


## Usando funções para o ajuste de escala automático do aplicativo Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) usa perfis vinculados ao serviço (IAM) AWS Identity and Access Management. Um perfil vinculado ao serviço é um tipo exclusivo de perfil do IAM vinculado diretamente ao Amazon Keyspaces. Os perfis vinculados ao serviço são predefinidos pelo Amazon Keyspaces e incluem todas as permissões que o serviço requer para chamar outros produtos da AWS em seu nome.

Um perfil vinculado ao serviço facilita a configuração do Amazon Keyspaces porque você não precisa adicionar as permissões necessárias manualmente. O Amazon Keyspaces define as permissões dos perfis vinculados serviços e, a não ser que esteja definido de outra forma, somente o Amazon Keyspaces poderá assumir os perfis. As permissões definidas incluem as políticas de confiança e de permissões, e essa política de permissões não pode ser anexada a nenhuma outra entidade do IAM.

Você só pode excluir um perfil vinculado a serviço depois de excluir os recursos relacionados. Isso protege seus recursos do Amazon Keyspaces, pois você não pode remover por engano as permissões para acessá-los.

Para obter informações sobre outros produtos que oferecem suporte às funções vinculadas a serviços, consulte Serviços da AWS que funcionam com o IAM e procure os serviços que apresentam Yes (Sim) na coluna Funções vinculadas ao serviço. Escolha Sim com um link para visualizar a documentação da função vinculada a esse serviço.

### Permissões de perfil vinculado ao serviço para o Amazon Keyspaces

O Amazon Keyspaces usa a função vinculada ao serviço nomeada AWSServiceRoleForApplicationAutoScaling_CassandraTablepara permitir que o Application Auto Scaling chame o Amazon Keyspaces e a Amazon em seu nome. CloudWatch

A função AWSServiceRoleForApplicationAutoScaling_CassandraTable vinculada ao serviço confia nos seguintes serviços para assumir a função:

- `cassandra.application-autoscaling.amazonaws.com`

A política de permissões do perfil permite que o ajuste de escala automático ao aplicativo conclua as seguintes ações nos recursos especificados:

- Ação: `cassandra:Select` em `arn:*:cassandra:*:*:/keyspace/system/table/*`
- Ação: `cassandra:Select` no recurso `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
- Ação: `cassandra:Select` no recurso `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Ação: `cassandra:Alter` no recurso `arn:*:cassandra:*:*:"*"`

### Criação de um perfil vinculado ao serviço para o Amazon Keyspaces

Você não precisa criar manualmente um perfil vinculado ao serviço para escalar automaticamente o Amazon Keyspaces. Quando você ativa o escalonamento automático do Amazon Keyspaces em uma tabela com aAWS Management Console, CQL, a ou a AWS APIAWS CLI, o Application Auto Scaling cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, você poderá usar esse mesmo processo para recriar o perfil em sua conta. Quando você ativa o escalonamento automático do Amazon Keyspaces para uma tabela, o Application Auto Scaling cria novamente a função vinculada ao serviço para você.

> **Important**
>
> Esse perfil vinculado ao serviço pode aparecer em sua conta se você concluiu uma ação em outro serviço que usa os atributos compatíveis com esse perfil. Para saber mais, consulte Um novo perfil apareceu na minha Conta da AWS.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, você poderá usar esse mesmo processo para recriar o perfil em sua conta. Quando você ativa a escalabilidade automática de aplicativo do Amazon Keyspaces para uma tabela, o ajuste de escala automático do aplicativo cria o perfil vinculado a serviços para você novamente.

Edição de um perfil vinculado ao serviço do Amazon Keyspaces

O Amazon Keyspaces não permite que você edite a função vinculada ao AWSServiceRoleForApplicationAutoScaling_CassandraTable serviço. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ela. No entanto, será possível editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Exclusão de um perfil vinculado ao serviço do Amazon Keyspaces

Se você não precisar mais usar um atributo ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-la. Dessa forma, você não terá uma entidade não utilizada que não seja monitorada ativamente ou mantida. Contudo, você deve primeiro desabilitar o escalonamento automático em todas as tabelas da conta em todas as Regiões da AWS antes de poder excluir manualmente o perfil vinculado ao serviço. Para desativar a escalabilidade automática nas tabelas do Amazon Keyspaces, consulte Modificação ou desativação das configurações de escalabilidade automática do Amazon Keyspaces.

> **Note**
>
> Se a escalabilidade automática do Amazon Keyspaces estiver usando o perfil quando você tentar modificar os recursos, o cancelamento do registro poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Como excluir manualmente o perfil vinculado ao serviço usando o IAM

Use o console do IAMAWS CLI, o ou a AWS API para excluir a função AWSServiceRoleForApplicationAutoScaling_CassandraTable vinculada ao serviço. Para obter mais informações, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

> **Note**
>
> Para excluir o perfil vinculado ao serviço usado pela escalabilidade automática do Amazon Keyspaces, você deve primeiro desativar a escalabilidade automática em todas as tabelas da conta.

Regiões com suporte para perfis vinculados ao serviço do Amazon Keyspaces

O Amazon Keyspaces oferece suporte a perfis vinculados ao serviço em todas as regiões em que o serviço está disponível. Para obter mais informações, consulte Endpoints de serviço do Amazon Keyspaces.

## Uso de perfis para replicação multirregional do Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) usa perfis vinculados ao serviço (IAM) AWS Identity and Access Management. Um perfil vinculado ao serviço é um tipo exclusivo de perfil do IAM vinculado diretamente ao Amazon Keyspaces. Os perfis vinculados ao serviço são predefinidos pelo Amazon Keyspaces e incluem todas as permissões que o serviço requer para chamar outros produtos da AWS em seu nome.

Um perfil vinculado ao serviço facilita a configuração do Amazon Keyspaces porque você não precisa adicionar as permissões necessárias manualmente. O Amazon Keyspaces define as permissões dos perfis vinculados serviços e, a não ser que esteja definido de outra forma, somente o Amazon Keyspaces poderá assumir os perfis. As permissões definidas incluem as políticas de confiança e de permissões, e essa política de permissões não pode ser anexada a nenhuma outra entidade do IAM.

Você só pode excluir um perfil vinculado a serviço depois de excluir os recursos relacionados. Isso protege seus recursos do Amazon Keyspaces, pois você não pode remover por engano as permissões para acessá-los.

Para obter informações sobre outros produtos que oferecem suporte às funções vinculadas a serviços, consulte Serviços da AWS que funcionam com o IAM e procure os serviços que apresentam Yes (Sim) na coluna Funções vinculadas ao serviço. Escolha Sim com um link para visualizar a documentação da função vinculada a esse serviço.

### Permissões de perfil vinculado ao serviço para o Amazon Keyspaces

O Amazon Keyspaces usa a função vinculada ao serviço chamada para permitir que o AWSServiceRoleForAmazonKeyspacesReplicationAmazon Keyspaces replique gravações em todas as réplicas de uma tabela multirregional em seu nome.

A função AWSServiceRoleForAmazonKeyspacesReplication vinculada ao serviço confia nos seguintes serviços para assumir a função:

- `replication.cassandra.amazonaws.com`

A política de permissões de função nomeada KeyspacesReplicationServiceRolePolicy permite que o Amazon Keyspaces conclua as seguintes ações:

- Ação: `cassandra:Select`
- Ação: `cassandra:SelectMultiRegionResource`
- Ação: `cassandra:Modify`
- Ação: `cassandra:ModifyMultiRegionResource`

Embora a função vinculada ao serviço Amazon Keyspaces AWSServiceRoleForAmazonKeyspacesReplication forneça as permissões: “Ação:” para o nome de recurso da Amazon (ARN) “arn: *” especificado na política, o Amazon Keyspaces fornece o ARN da sua conta.

Você deve configurar permissões para permitir que seus usuários, grupos ou perfis criem, editem ou excluam um perfil vinculado ao serviço. Para ter mais informações, consulte Permissões de função vinculada a serviços no Guia do usuário do IAM.

### Criação de um perfil vinculado ao serviço para o Amazon Keyspaces

Não é possível criar manualmente um perfil vinculado ao serviço. Quando você cria um espaço de chaves multirregional no AWS Management Console, no AWS CLI ou na API da AWS, o Amazon Keyspaces cria o perfil vinculado ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, você poderá usar esse mesmo processo para recriar o perfil em sua conta. Quando você cria um espaço de chaves multirregional, o Amazon Keyspaces cria o perfil vinculado ao serviço para você novamente.

Edição de um perfil vinculado ao serviço do Amazon Keyspaces

O Amazon Keyspaces não permite que você edite a função vinculada ao AWSServiceRoleForAmazonKeyspacesReplication serviço. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ela. No entanto, será possível editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Exclusão de um perfil vinculado ao serviço do Amazon Keyspaces

Se você não precisar mais usar um recurso ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-lo. Dessa forma, você não tem uma entidade não utilizada que não seja monitorada ativamente ou mantida. Contudo, você deve primeiro excluir todos os espaços de chaves multirregionais da conta em todas as Regiões da AWS antes de poder excluir manualmente o perfil vinculado ao serviço.

Limpar um perfil vinculado ao serviço

Antes de usar o IAM para excluir um perfil vinculado ao serviço, você deve primeiro excluir quaisquer espaços de chaves e tabelas multirregionais usados pelo perfil.

> **Note**
>
> Se o serviço do Amazon Keyspaces estiver usando o perfil quando você tentar excluir os recursos, a exclusão poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Para excluir recursos do Amazon Keyspaces usados pelo AWSServiceRoleForAmazonKeyspacesReplication (console)

1. Faça login no AWS Management Console e abra o console do Amazon Keyspaces em https://console.aws.amazon.com/keyspaces/home.
2. Selecione Espaços de chaves no painel do lado esquerdo.
3. Selecione todos os espaços de chaves multirregionais da lista.
4. Selecione Excluir, confirme a exclusão e selecione Excluir espaços de chaves.

Você também pode excluir os espaços de chaves multirregionais de forma programática usando qualquer um dos seguintes métodos.

- A instrução ??? do Cassandra Query Language (CQL).
- A operação delete-keyspace do CLI da AWS.
- A DeleteKeyspaceoperação da API Amazon Keyspaces.

Excluir manualmente o perfil vinculado ao serviço

Use o console do IAM, a AWS CLI ou a API da AWS para excluir o perfil vinculado ao serviço AWSServiceRoleForAmazonKeyspacesReplication. Para obter mais informações, consulte Excluir uma função vinculada ao serviço no Guia do usuário do IAM.

Regiões com suporte para perfis vinculados ao serviço do Amazon Keyspaces

O Amazon Keyspaces não oferece suporte ao uso de perfis vinculados a serviços em todas as regiões em que o serviço está disponível. Você pode usar a AWSServiceRoleForAmazonKeyspacesReplication função nas seguintes regiões.

| Nome da região | Identidade da região | Suporte no Amazon Keyspaces |
|---|---|---|
| Leste dos EUA (Norte da Virgínia) | us-east-1 | Sim |
| Leste dos EUA (Ohio) | us-east-2 | Sim |
| Oeste dos EUA (Norte da Califórnia) | us-west-1 | Sim |
| Oeste dos EUA (Oregon) | us-west-2 | Sim |
| Ásia-Pacífico (Mumbai) | ap-south-1 | Sim |
| Ásia-Pacífico (Osaka) | ap-northeast-3 | Sim |
| Ásia-Pacífico (Seul) | ap-northeast-2 | Sim |
| Ásia-Pacífico (Singapura) | ap-southeast-1 | Sim |
| Ásia-Pacífico (Sydney) | ap-southeast-2 | Sim |
| Ásia-Pacífico (Tóquio) | ap-northeast-1 | Sim |
| Canadá (Central) | ca-central-1 | Sim |

| Nome da região | Identidade da região | Suporte no Amazon Keyspaces |
|---|---|---|
| Europa (Frankfurt) | eu-central-1 | Sim |
| Europa (Irlanda) | eu-west-1 | Sim |
| Europa (Londres) | eu-west-2 | Sim |
| Europa (Paris) | eu-west-3 | Sim |
| América do Sul (São Paulo) | sa-east-1 | Sim |
| AWS GovCloud (Leste dos EUA) | us-gov-east-1 | Não |
| AWS GovCloud (Oeste dos EUA) | us-gov-west-1 | Não |

# Validação de conformidade do Amazon Keyspaces (para Apache Cassandra)

Auditores terceirizados avaliam a segurança e a conformidade do Amazon Keyspaces (para Apache Cassandra) como parte de vários programas de conformidade. AWS Isso inclui:

- ISO/IEC 27001:2013, 27017:2015, 27018:2019, e ISO/IEC 9001:2015. Para obter mais informações, consulte as certificações e serviços ISO e CSA STAR da AWS.
- Controles do Sistema e da Organização (CSO)
- PCI (Payment Card Industry)
- Federal Risk and Authorization Management Program (FedRAMP) High
- Health Insurance Portability and Accountability Act (HIPAA)

Para saber se um AWS service (Serviço da AWS) está dentro do escopo de programas de conformidade específicos, consulte Serviços da AWS Escopo por Programa de Conformidade Serviços da AWS e escolha o programa de conformidade em que você está interessado. Para obter informações gerais, consulte Programas de AWS conformidade Programas AWS de .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade de conformidade ao usar Serviços da AWS é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido sobre segurança e conformidade — Esses guias de implantação discutem considerações arquitetônicas e fornecem etapas para a implantação de ambientes básicos AWS focados em segurança e conformidade.
- Arquitetura para segurança e conformidade com a HIPAA na Amazon Web Services — Este whitepaper descreve como as empresas podem usar AWS para criar aplicativos qualificados para a HIPAA.

> **Note**
>
> Nem todos Serviços da AWS são elegíveis para a HIPAA. Para obter mais informações, consulte a Referência dos serviços qualificados pela HIPAA.

- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- AWS Guias de conformidade do cliente — Entenda o modelo de responsabilidade compartilhada sob a ótica da conformidade. Os guias resumem as melhores práticas de proteção Serviços da AWS e mapeiam as diretrizes para controles de segurança em várias estruturas (incluindo o Instituto Nacional de Padrões e Tecnologia (NIST), o Conselho de Padrões de Segurança do Setor de Cartões de Pagamento (PCI) e a Organização Internacional de Padronização (ISO)).
- Avaliação de recursos com regras no Guia do AWS Config desenvolvedor — O AWS Config serviço avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- AWS Security Hub— Isso AWS service (Serviço da AWS) fornece uma visão abrangente do seu estado de segurança interno AWS. O Security Hub usa controles de segurança para avaliar os atributos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a Referência de controles do Security Hub.
- Amazon GuardDuty — Isso AWS service (Serviço da AWS) detecta possíveis ameaças às suas cargas de trabalho Contas da AWS, contêineres e dados monitorando seu ambiente em busca de atividades suspeitas e maliciosas. GuardDuty pode ajudá-lo a atender a vários requisitos de conformidade, como o PCI DSS, atendendo aos requisitos de detecção de intrusões exigidos por determinadas estruturas de conformidade.

- AWS Audit Manager— Isso AWS service (Serviço da AWS) ajuda você a auditar continuamente seu AWS uso para simplificar a forma como você gerencia o risco e a conformidade com as regulamentações e os padrões do setor.

# Resiliência e recuperação de desastres no Amazon Keyspaces

A infraestrutura global da AWS se baseia em Regiões da AWS e zonas de disponibilidade. A Regiões da AWS oferece várias zonas de disponibilidade separadas e isoladas fisicamente que são conectadas com baixa latência, altas taxas de transferência e em redes altamente redundantes. Com as Zonas de Disponibilidade, você pode projetar e operar aplicativos e bancos de dados que executam o failover automaticamente entre as Zonas de Disponibilidade sem interrupção. As Zonas de Disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de datacenter tradicionais.

O Amazon Keyspaces replica dados automaticamente três vezes em várias Zonas de Disponibilidade da AWS dentro da mesma Região da AWS para maior durabilidade e alta disponibilidade.

Para obter mais informações sobre Regiões da AWS e Zonas de Disponibilidade, consulte Infraestrutura global da AWS.

Além da infraestrutura global da AWS, o Amazon Keyspaces oferece vários atributos para ajudar a dar suporte às necessidades de resiliência de dados e backup.

Replicação multirregional

O Amazon Keyspaces fornece replicação em várias regiões se você precisar replicar seus dados ou aplicações para distâncias geográficas maiores. Você pode replicar suas tabelas do Amazon Keyspaces em até seis Regiões da AWS diferentes de sua escolha. Para obter mais informações, consulte Replicação multirregional.

Recuperação para um ponto no tempo (PITR)

A PITR ajuda a proteger as tabelas do Amazon Keyspaces contra operações acidentais de gravação ou exclusão ao fornecer backups contínuos dos dados da tabela. Para obter mais informações, consulte Recuperação para um ponto no tempo para o Amazon Keyspaces.

# Segurança da infraestrutura no Amazon Keyspaces

Como um serviço gerenciado, o Amazon Keyspaces (para Apache Cassandra) é protegido pela segurança da rede global da AWS. Para obter informações sobre serviços de segurança da AWS e como a AWS protege a infraestrutura, consulte Segurança na Nuvem AWS. Para projetar seu ambiente da AWS usando as práticas recomendadas de segurança de infraestrutura, consulte Proteção de infraestrutura em Pilar segurança: AWS Well-Architected Framework.

Você usa as chamadas de API publicadas da AWS para acessar o Amazon Keyspaces por meio da rede. Os clientes devem oferecer suporte para:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

O Amazon Keyspaces oferece suporte a dois métodos de autenticação de solicitações de clientes. O primeiro método usa credenciais específicas do serviço, que são credenciais baseadas em senha geradas para um usuário do IAM específico. Você pode criar e gerenciar a senha usando o console do IAM, o AWS CLI ou o API da AWS. Para obter mais informações, consulte Como usar o IAM com o Amazon Keyspaces.

O segundo método usa um complemento de autenticação para o driver Java DataStax de código aberto para o Cassandra. Esse plug-in permite que usuários, perfis e identidades federadas do IAM adicionem informações de autenticação às solicitações de API do Amazon Keyspaces (para Apache Cassandra) usando o AWSProcesso do Signature Version 4 (SigV4). Para obter mais informações, consulte the section called “Credenciais do IAM para autenticação AWS ”.

Você pode usar um endpoint da VPC de interface para manter o tráfego entre sua Amazon VPC e o Amazon Keyspaces saindo da rede da Amazon. Os endpoints da VPC de interface são desenvolvidos pelo AWS PrivateLink, uma tecnologia da AWS que permite a comunicação privada entre os serviços da AWS usando uma interface de rede elástica com IPs privados em sua Amazon

VPC. Para obter mais informações, consulte the section called “Usar VPC endpoints da interface do ”.

## Como usar o Amazon Keyspaces com endpoint da VPC de interface

Os endpoints da VPC de interface permitem a comunicação privada entre a nuvem privada virtual (VPC) em execução no Amazon VPC e no Amazon Keyspaces. Os endpoints VPC de interface são alimentados por AWS PrivateLink, que é um AWS serviço que permite a comunicação privada entre VPCs e serviços. AWS

AWS PrivateLink permite isso usando uma interface de rede elástica com endereços IP privados em sua VPC para que o tráfego de rede não saia da rede Amazon. Os endpoints da VPC de interface não exigem um gateway da Internet, dispositivo NAT, conexão VPN ou conexão do AWS Direct Connect . Para obter mais informações, consulte a Amazon Virtual Private Cloud e os Endpoints da VPC de interface (AWS PrivateLink).

Tópicos

- Como usar o endpoint da VPC para o Amazon Keyspaces
- Como preencher entradas da tabela system.peers com informações do endpoint da VPC de interface
- Como controlar o acesso aos endpoints da VPC de interface para o Amazon Keyspaces
- Disponibilidade
- Políticas de endpoint de VPC e recuperação do Amazon point-in-time Keyspaces (PITR)
- Erros e avisos comuns

### Como usar o endpoint da VPC para o Amazon Keyspaces

Você pode criar uma endpoint da VPC de interface para que o tráfego entre o Amazon Keyspaces e seus recursos do Amazon VPC comece a fluir pelo endpoint da VPC de interface. Para começar, siga as etapas para criar um endpoint de interface. Em seguida, edite o grupo de segurança associado ao endpoint que você criou na etapa anterior e configure uma regra de entrada para a porta 9142. Para obter mais informações consulte Adicionar, remover e atualizar regras.

Para obter um step-by-step tutorial sobre como configurar uma conexão com o Amazon Keyspaces por meio de um VPC endpoint, consulte. the section called “Conexão com VPC endpoints” Para saber como configurar o acesso entre contas para recursos do Amazon Keyspaces separados de

aplicativos Contas da AWS diferentes em uma VPC, consulte. the section called “Acesso entre contas”

## Como preencher entradas da tabela `system.peers` com informações do endpoint da VPC de interface

Os drivers do Apache Cassandra usam a tabela `system.peers` para consultar as informações do nó sobre o cluster. Os drivers do Cassandra usam as informações do nó para balancear a carga das conexões e repetir as operações. O Amazon Keyspaces preenche nove entradas na tabela `system.peers` automaticamente para clientes que se conectam por meio do endpoint público.

Para fornecer aos clientes uma conexão por meio de endpoints da VPC de interface com funcionalidade semelhante, o Amazon Keyspaces preenche a tabela `system.peers` em sua conta com uma entrada para cada zona de disponibilidade em que um endpoint da VPC está disponível. Para pesquisar e armazenar endpoints da VPC de interface disponíveis na tabela `system.peers`, o Amazon Keyspaces exige que você conceda à entidade IAM usada para se conectar ao Amazon Keyspaces permissões de acesso para consultar sua VPC para obter informações sobre o endpoint e a interface de rede.

> **Important**
>
> Preencher a tabela `system.peers` com os endpoints da VPC de interface disponível melhora o balanceamento de carga e aumenta o throughput de leitura/gravação. É recomendado para todos os clientes que acessam o Amazon Keyspaces usando endpoint da VPC de interface e é obrigatório para o Apache Spark.

Para conceder à entidade IAM usada para se conectar ao Amazon Keyspaces permissões para pesquisar as informações necessárias do endpoint da VPC de interface, você pode atualizar seu perfil do IAM ou política de usuário existente ou criar uma nova política do IAM, conforme mostrado no exemplo a seguir.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
```

```
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> As políticas gerenciadas `AmazonKeyspacesReadOnlyAccess_v2` e `AmazonKeyspacesFullAccess` incluem as permissões necessárias para permitir que o Amazon Keyspaces acesse a instância do Amazon EC2 para ler informações sobre os endpoints da VPC de interface disponíveis.

Para confirmar se a política foi configurada corretamente, consulte a tabela `system.peers` para ver as informações de rede. Se a tabela `system.peers` estiver vazia, isso pode indicar que a política não foi configurada com êxito ou que você excedeu a cota de taxa de solicitação para as ações da API `DescribeNetworkInterfaces` e `DescribeVPCEndpoints`. `DescribeVPCEndpoints` se enquadra na categoria `Describe*` e é considerada uma ação não mutante. `DescribeNetworkInterfaces` se enquadra no subconjunto de ações não filtradas e não paginadas e não mutantes, e diferentes cotas se aplicam. Para obter mais informações, consulte Solicitar tamanhos de buckets de tokens e taxas de recarga na Referência de API do Amazon EC2.

Se você vir uma tabela vazia, tente novamente alguns minutos depois para descartar problemas de cota de taxa de solicitação. Para verificar se você configurou os endpoints da VPC corretamente, consulte the section called “Erros de conexão do endpoint da VPC”. Se sua consulta retornar resultados da tabela, sua política foi configurada corretamente.

## Como controlar o acesso aos endpoints da VPC de interface para o Amazon Keyspaces

Com as políticas de endpoint da VPC, você pode controlar o acesso aos recursos de duas maneiras:

- Política do IAM – Você pode controlar as solicitações, os usuários ou os grupos que têm permissão para acessar o Amazon Keyspaces por meio de um endpoint da VPC específico. É possível fazer isso usando uma chave de condição na política anexada a um usuário, grupo ou perfil do IAM.

- Política da VPC – Você pode controlar quais endpoints da VPC têm acesso aos recursos do Amazon Keyspaces anexando políticas a eles. Para restringir o acesso a um espaço de chaves ou a uma tabela específica para permitir apenas o tráfego proveniente de um endpoint da VPC específico, edite a política do IAM existente que restringe o acesso aos recursos e adicione esse endpoint da VPC.

Veja a seguir exemplos de políticas de endpoint para acessar os recursos do Amazon Keyspaces.

- Exemplo de política do IAM: restringir todo o acesso a uma tabela específica do Amazon Keyspaces, a menos que o tráfego venha do endpoint da VPC especificado – Esse exemplo de política pode ser anexado a um usuário, um perfil ou um grupo do IAM. Ela restringe o acesso a uma tabela específica do Amazon Keyspaces, a menos que o tráfego de entrada seja originado de um endpoint da VPC específico.

```
{
   "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**
>
> Para restringir o acesso a uma tabela específica, você também deve incluir o acesso às tabelas do sistema. As tabelas do sistema são somente leitura.

- Exemplo de política de VPC: acesso somente leitura – esse exemplo de política pode ser anexado a um endpoint da VPC. (Para obter mais informações, consulte Como controlar o acesso aos

recursos da Amazon VPC). Ela restringe as ações ao acesso somente de leitura aos recursos do Amazon Keyspaces por meio do endpoint da VPC ao qual está conectada.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadOnly",
      "Principal": "*",
      "Action": [
        "cassandra:Select"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- Exemplo de política de VPC: restringir o acesso a uma tabela específica do KinesisAmazon Keyspaces – Esse exemplo de política pode ser anexado a um endpoint da VPC. Ela restringe o acesso a uma tabela específica por meio do endpoint da VPC ao qual está anexada.

```
{
   "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**
>
> Para restringir o acesso a uma tabela específica, você também deve incluir o acesso às tabelas do sistema. As tabelas do sistema são somente leitura.

## Disponibilidade

O Amazon Keyspaces oferece suporte ao uso de endpoints VPC de interface em todos os lugares em que o serviço Regiões da AWS está disponível. Para ter mais informações, consulte ???.

## Políticas de endpoint de VPC e recuperação do Amazon point-in-time Keyspaces (PITR)

Se você estiver usando políticas do IAM com chaves de condição para restringir o tráfego de entrada, a operação de restauração da tabela poderá falhar. Por exemplo, se você restringir o tráfego de origem a endpoints da VPC específicos usando chaves de condição `aws:SourceVpce`, a operação de restauração da tabela falhará. Para permitir que o Amazon Keyspaces execute uma operação de restauração em nome de sua entidade principal, você deve adicionar uma chave de condição `aws:ViaAWSService` à sua política do IAM. A chave de `aws:ViaAWSService` condição permite o acesso quando qualquer AWS serviço faz uma solicitação usando as credenciais do diretor. Para obter mais informações, consulte Elementos de política JSON do IAM: chave de condição no Guia do usuário do IAM. A política a seguir é um exemplo disso.

```
{
   "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"
             }
          }
       }
    ]
}
```

### Erros e avisos comuns

Se você estiver usando a Amazon Virtual Private Cloud e se conectar ao Amazon Keyspaces, talvez veja o seguinte aviso.

```
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.
```

Esse aviso ocorre porque a tabela `system.peers` contém entradas para todos os endpoint da VPC da Amazon que o Amazon Keyspaces tem permissão para visualizar, incluindo o endpoint da VPC da Amazon ao qual você está conectado. Você pode ignorar esse aviso com segurança.

Para outros erros, consulte the section called “Erros de conexão do endpoint da VPC”.

## Análise de configuração e vulnerabilidade no Amazon Keyspaces

A AWS se encarrega das tarefas básicas de segurança, como aplicação de patches a bancos de dados e sistemas operacionais (SOs) convidados, configuração de firewalls e recuperação de desastres. Esses procedimentos foram revisados e certificados por terceiros certificados. Para obter mais detalhes, consulte os seguintes recursos da :

- Modelo de responsabilidade compartilhada
- Amazon Web Services: Overview of security processes (Amazon Web Services: visão geral do processo de segurança) (whitepaper)

# Práticas recomendadas de segurança para o Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) fornece uma série de atributos de segurança a serem considerados no desenvolvimento e na implementação das suas próprias políticas de segurança. As práticas recomendadas a seguir são diretrizes gerais e não representam uma solução completa de segurança. Como essas práticas recomendadas podem não ser adequadas ou suficientes no seu ambiente, trate-as como considerações úteis em vez de requisitos.

Tópicos

- Práticas preventivas de segurança recomendadas para o Amazon Keyspaces
- Práticas recomendadas de detecção de segurança para o Amazon Keyspaces

## Práticas preventivas de segurança recomendadas para o Amazon Keyspaces

As práticas recomendadas de segurança a seguir são consideradas preventivas porque podem ajudar você a antecipar e prevenir incidentes de segurança no Amazon Keyspaces.

Use criptografia em repouso

O Amazon Keyspaces criptografa em repouso todos os dados de usuário gravados em tabelas, usando chaves de criptografia armazenadas no AWS Key Management Service (AWS KMS). Isso oferece uma camada de proteção de dados adicional ao proteger seus dados contra acesso não autorizado ao armazenamento subjacente.

Por padrão, o Amazon Keyspaces usa uma Chave pertencente à AWS para criptografar todas as suas tabelas. Se essa chave não existir, ela é criada para você. As chaves padrão do serviço não podem ser desabilitadas.

Como alternativa, você pode usar uma chave gerenciada pelo cliente para criptografia em repouso. Para obter mais informações, consulte Criptografia em repouso do Amazon Keyspaces.

Usar perfis do IAM para autenticar o acesso ao Amazon Keyspaces

Para que usuários, aplicações e outros serviços da AWS possam acessar o Amazon Keyspaces, eles devem incluir credenciais da AWS válidas em suas solicitações à API da AWS. Você não deve armazenar credenciais da AWS diretamente na aplicação ou na instância do EC2. Essas são credenciais de longo prazo que não são automaticamente alternadas e, portanto, podem ter impacto comercial significativo se forem comprometidas. Um perfil do IAM permite obter chaves de acesso temporárias que podem ser usadas para acessar os serviços e recursos da AWS.

Para obter mais informações, consulte Funções do IAM.

Usar políticas do IAM para autorizações de base do Amazon Keyspaces

Ao conceder permissões, você decide quem as recebe, a quais APIs do Amazon Keyspaces as permissões se referem e as ações específicas que deseja permitir nesses recursos. A implementação do privilégio mínimo é fundamental para reduzir os riscos de segurança e o impacto que pode resultar de erros ou usuários mal-intencionados.

Anexe políticas de permissões para identidades do IAM (ou seja, usuários, grupos e perfis) e, assim, dê permissões para eles executarem operações nos recursos do Amazon Keyspaces.

Para isso, você pode usar o seguinte:

- Políticas gerenciadas (predefinidas) da AWS
- Políticas gerenciadas pelo cliente

Uso de condições de política do IAM para controle de acesso refinado

Ao conceder permissões no Amazon Keyspaces, você pode especificar as condições que determinam como uma política de permissões entra em vigor. A implementação do privilégio mínimo é fundamental para reduzir os riscos de segurança e o impacto que pode resultar de erros ou usuários mal-intencionados.

É possível especificar as condições ao conceder permissões usando uma política do IAM. Por exemplo, você pode fazer o seguinte:

- Conceda permissões para permitir que os usuários tenham acesso somente leitura a tabelas ou espaços de chaves específicos.
- Conceda permissões para permitir que um usuário tenha acesso de gravação a uma determinada tabela, com base na identidade desse usuário.

Para obter mais informações, consulte Exemplos de políticas baseadas em identidade.

### Considere utilizar a criptografia do lado do cliente

Se você armazena dados confidenciais e sensíveis no Amazon Keyspaces, talvez seja melhor criptografar os dados o mais próximo possível da origem, para que eles fiquem protegidos durante todo o ciclo de vida. A criptografia dos seus dados confidenciais em trânsito e em repouso ajuda você a garantir que os dados em texto simples não estejam disponíveis a terceiros.

## Práticas recomendadas de detecção de segurança para o Amazon Keyspaces

As práticas recomendadas de segurança a seguir são consideradas de detecção porque podem ajudá-lo a detectar possíveis falhas e incidentes de segurança.

### Use AWS CloudTrail para monitorar o uso da chave AWS Key Management Service (AWS KMS) AWS KMS

Se você estiver usando uma chave do AWS KMS gerenciada pelo cliente para criptografia em repouso, o uso dessa chave será registrado no AWS CloudTrail. O CloudTrail fornece visibilidade da atividade do usuário ao registrar ações realizadas em sua conta. O CloudTrail registra informações importantes sobre cada ação, incluindo quem fez a solicitação, os serviços usados, as ações realizadas, os parâmetros das ações e os elementos de resposta retornados pelo serviço da AWS. Essas informações ajudam você a rastrear as alterações feitas em seus recursos da AWS e solucionar problemas operacionais. O CloudTrail facilita garantir a conformidade com as políticas internas e os padrões regulatórios.

Use o CloudTrail para auditar o uso de chaves. O CloudTrail cria arquivos de log que contêm um histórico de chamadas da AWS API e eventos relacionados da sua conta. Esses arquivos de log incluem todas as solicitações da API do AWS KMS feitas usando o console, SDKs da AWS e ferramentas da linha de comando, além daquelas feitas por meio de serviços AWS integrados. Você pode usar esses arquivos de log para obter informações sobre quando a chave do AWS KMS foi usada, a operação solicitada, a identidade do solicitante, o endereço IP de origem da solicitação e assim por diante. Para obter mais informações, consulte Como registrar chamadas de API do AWS Key Management Service com o AWS CloudTrail no Guia do usuário do AWS CloudTrail.

Use o CloudTrail para monitorar operações da linguagem de definição de dados (DDL) do Amazon Keyspaces

O CloudTrail fornece visibilidade da atividade do usuário ao registrar ações realizadas em sua conta. O CloudTrail registra informações importantes sobre cada ação, incluindo quem fez a solicitação, os serviços usados, as ações realizadas, os parâmetros das ações e os elementos de resposta retornados pelo serviço da AWS. Essas informações o ajudam a rastrear as alterações feitas em seus recursos da AWS e solucionar problemas operacionais. O CloudTrail facilita garantir a conformidade com as políticas internas e os padrões regulatórios.

Todas as operações de DDL do Amazon Keyspaces são registradas automaticamente no CloudTrail. As operações DDL permitem criar e gerenciar tabelas e espaços de chaves do Amazon Keyspaces.

Quando uma atividade ocorre no Amazon Keyspaces, ela é registrada em um evento do CloudTrail com outros eventos de serviços da AWS no histórico de eventos. Para obter mais informações, consulte Registrar operações do Amazon Keyspaces usando o AWS CloudTrail. Você pode visualizar, pesquisar e baixar eventos recentes em sua Conta da AWS. Para obter mais informações, consulte o tópico sobre como Visualizar eventos com o histórico de eventos do CloudTrail, no Guia do usuário do AWS CloudTrail.

Para obter um registro dos eventos em andamento na sua Conta da AWS, incluindo eventos do Amazon Keyspaces, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon Simple Storage Service (Amazon S3). Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as Regiões da AWS. A trilha registra eventos de todas as regiões na partição da AWS e fornece os arquivos de log ao bucket do S3 que você especificar. Além disso, é possível configurar outros serviços da AWS para analisar mais ainda mais e agir com base nos dados de eventos coletados nos logs do CloudTrail.

Marque seus recursos do Amazon Keyspaces para identificação e automação

Você pode atribuir metadados aos seus recursos da AWS na forma de tags. Cada tag é um rótulo simples que consiste em uma chave definida pelo cliente e um valor opcional que pode facilitar o gerenciamento, a pesquisa e a filtragem de recursos.

A atribuição de tags (tagging) permite a implementação de controles agrupados. Embora não haja tipos de tags inerentes, elas permitem categorizar recursos por finalidade, proprietário, ambiente ou outros critérios. Veja os seguintes exemplos:

- Acesso – usado para controlar o acesso aos recursos do Amazon Keyspaces com base em tags. Para obter mais informações, consulte the section called “Autorização baseada em tags do Amazon Keyspaces”.
- Segurança – usada para determinar requisitos como configurações de proteção de dados.
- Confidencialidade – um identificador para o nível de confidencialidade de dados específico suportado por um recurso.
- Ambiente: usado para distinguir entre as infraestruturas de desenvolvimento, teste e produção.

Para obter mais informações, consulte AWS estratégias de marcação e Adicionar tags e rótulos a recursos.

# Referência de idiomas CQL para Amazon Keyspaces (para Apache Cassandra)

Depois de se conectar a um endpoint do Amazon Keyspaces (para Apache Cassandra), você usa o Cassandra Query Language (CQL) para trabalhar com seu banco de dados. O CQL é semelhante em muitos aspectos ao Structured Query Language (SQL).

Tópicos


- Elementos do Cassandra Query Language (CQL) no Amazon Keyspaces
- Instruções DDL (linguagem de definição de dados) no Amazon Keyspaces
- Declarações DML (linguagem de manipulação de dados) no Amazon Keyspaces
- Funções integradas no Amazon Keyspaces


## Elementos do Cassandra Query Language (CQL) no Amazon Keyspaces

Saiba mais sobre os elementos do Cassandra Query Language (CQL) que são compatíveis com o Amazon Keyspaces, incluindo identificadores, constantes, termos e tipos de dados.

Tópicos


- Identificadores
- Constantes
- Termos
- Tipos de dados
- Codificação JSON dos tipos de dados do Amazon Keyspaces


### Identificadores

Identificadores (ou nomes) são usados para identificar tabelas, colunas e outros objetos. Um identificador pode ser citado ou não. O seguinte se aplica.

```
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

As seguintes constantes são definidas.

```
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+
```

## Termos

Um termo denota o tipo de valores que são compatíveis. Os termos são definidos pelo seguinte.

```
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
```

## Tipos de dados

O Amazon Keyspaces é compatível com os seguintes tipos de dados:

### Tipos de string

| Tipo de dados | Descrição |
| --- | --- |
| `ascii` | Representa uma cadeia de caracteres ASCII. |

| Tipo de dados | Descrição |
| --- | --- |
| `text` | Representa uma cadeia de caracteres codificad a UTF-8. |
| `varchar` | Representa uma string codificada em UTF-8 (`varchar` é um alias para `text`). |

## Tipos numéricos

| Tipo de dados | Descrição |
| --- | --- |
| `bigint` | Representa um comprimento assinado de 64 bits. |
| `counter` | Representa um contador de números inteiros assinado de 64 bits. Para ter mais informações, consulte the section called “Contadores”. |
| `decimal` | Representa um decimal de precisão variável. |
| `double` | Representa um ponto flutuante IEEE 754 de 64 bits. |
| `float` | Representa um ponto flutuante IEEE 754 de 32 bits. |
| `int` | Representa um int assinado de 32 bits. |
| `varint` | Representa um número inteiro de precisão arbitrária. |

### Contadores

Uma `counter` coluna contém um número inteiro assinado de 64 bits. O valor do contador é incrementado ou diminuído usando a instrução the section called “UPDATE” e não pode ser definido diretamente. Isso torna as colunas `counter` úteis para rastrear contagens. Por exemplo, você pode usar contadores para rastrear o número de entradas em um arquivo de log ou o número de vezes

que uma publicação foi visualizada em uma rede social. As restrições a seguir se aplicam às colunas `counter`:

- Uma coluna do tipo `counter` não pode fazer parte do `primary key` de uma tabela.
- Em uma tabela que contém uma ou mais colunas do tipo `counter`, todas as colunas dessa tabela devem ser do tipo `counter`.

Nos casos em que uma atualização do contador falha (por exemplo, devido ao tempo limite ou à perda de conexão com o Amazon Keyspaces), o cliente não sabe se o valor do contador foi atualizado. Se a atualização for repetida, a atualização do valor do contador poderá ser aplicada pela segunda vez.

### Tipo Blob

| Tipo de dados | Descrição |
|---|---|
| `blob` | Representa bytes arbitrários. |

### Tipo booliano

| Tipo de dados | Descrição |
|---|---|
| `boolean` | Representa `true` ou `false`. |

### Tipos relacionados ao tempo

| Tipo de dados | Descrição |
|---|---|
| `timestamp` | Número inteiro assinado de 64 bits representando a data e a hora desde a época (1º de janeiro de 1970 às 00:00:00 GMT) em milissegundos. |
| `timeuuid` | Representa um UUID da versão 1. |

## Tipos de coleção

| Tipo de dados | Descrição |
|---|---|
| `list` | Representa uma coleção ordenada de elementos literais. |
| `map` | Representa uma coleção não ordenada de pares de chave-valor. |
| `set` | Representa uma coleção não ordenada de um ou mais elementos literais. |

Você declara uma coluna de coleta usando o tipo de coleta seguido por outro tipo de dados (por exemplo, `TEXT` ou `INT`) entre parênteses angulares “**<>**”. Você pode criar uma coluna com um `SET` de `TEXT`, ou você pode criar um par de valor-chave `MAP` de `TEXT` e `INT`, conforme mostrado no exemplo a seguir.

```
SET <TEXT>
MAP <TEXT, INT>
```

Uma coleção não congelada permite que você faça atualizações em cada elemento individual da coleção. Os carimbos de data e hora do lado do cliente e as configurações de tempo de vida (TTL) são armazenados para elementos individuais.

Quando você usa a palavra-chave `FROZEN` em um tipo de coleção, os valores da coleção são serializados em um único valor imutável, e o Amazon Keyspaces os trata como um `BLOB`. Esta é uma coleção congelada. Uma declaração `INSERT` ou `UPDATE` sobrescreve toda a coleção congelada. Você não pode fazer atualizações em elementos individuais dentro de uma coleção congelada.

Os carimbos de data/hora do lado do cliente e as configurações de tempo de vida (TTL) se aplicam a toda a coleção congelada, não a elementos individuais. Colunas da coleção `Frozen` podem fazer parte de uma tabela de `PRIMARY KEY`.

Você pode aninhar coleções congeladas. Por exemplo, você pode definir o `MAP` dentro de um `SET` se `MAP` estiver usando a palavra-chave `FROZEN`, conforme mostrado no exemplo a seguir.

```
SET <FROZEN> <MAP <TEXT, INT>>>
```

Por padrão, o Amazon Keyspaces suporta o aninhamento de até cinco níveis de coleções congeladas. Para ter mais informações, consulte the section called “Service Quotas do Amazon Keyspaces”. Para obter mais informações sobre diferenças funcionais com o Apache Cassandra, consulte the section called “Coleções FROZEN”. Para obter mais informações sobre sintaxe de CQL, consulte the section called “CRIAR TABELA” e the section called “ALTER TABLE”.

### Tipo de tupla

O tipo de dados `tuple` representa um grupo limitado de elementos literais. Você pode usar uma tupla como alternativa a `user defined type`. Você não precisa usar a palavra-chave `FROZEN` para tuplas. Isso ocorre porque uma tupla está sempre congelada e você não pode atualizar elementos individualmente.

### Outros tipos

| Tipo de dados | Descrição |
| --- | --- |
| `inet` | Uma cadeia de caracteres representando um endereço IP, no formato IPv4 ou IPv6. |

### Estático

Em uma tabela do Amazon Keyspaces com colunas de agrupamento, você pode usar a `STATIC` palavra-chave para criar uma coluna estática de qualquer tipo.

A instrução a seguir é um exemplo disso.

```
my_column INT STATIC
```

Para obter mais informações sobre como trabalhar com colunas estáticas, consultethe section called “Colunas estáticas”.

## Codificação JSON dos tipos de dados do Amazon Keyspaces

O Amazon Keyspaces oferece os mesmos mapeamentos de tipos de dados JSON que o Apache Cassandra. A tabela a seguir descreve os tipos de dados que o Amazon Keyspaces aceita em

declarações `INSERT JSON` e os tipos de dados que o Amazon Keyspaces usa ao retornar dados com a declaração `SELECT JSON`.

Para tipos de dados de campo único como `float`, `int`, `UUID` e `date`, você também pode inserir dados como `string`. Para tipos de dados compostos e coleções, como `tuple`, `map` e `list`, você também pode inserir dados como JSON ou codificados `JSON string`.

| Tipo de dados do JSON | Tipos de dados aceitos em declarações **`INSERT JSON`** | Tipos de dados retornados em declarações **`SELECT JSON`** | Observações |
|---|---|---|---|
| `ascii` | `string` | `string` | Usa `\u` escape de caracteres JSON. |
| `bigint` | `integer, string` | `integer` | A string deve ser um número inteiro válido de 64 bits. |
| `blob` | `string` | `string` | A string deve começar com `0x` seguida por um número par de dígitos hexadecimais. |
| `boolean` | `boolean, string` | `boolean` | String deve ser `true` ou `false`. |
| `date` | `string` | `string` | Data em formato `YYYY-MM-DD`, fuso horário UTC. |
| `decimal` | `integer, float, string` | `float` | Pode exceder a precisão de ponto flutuante IEEE-754 de 32 bits ou 64 bits no decodificador do lado do cliente. |

| Tipo de dados do JSON | Tipos de dados aceitos em declaraçõ es **`INSERT JSON`** | Tipos de dados retornados em declarações **`SELECT JSON`** | Observações |
|---|---|---|---|
| `double` | `integer, float, string` | `float` | A string deve ser um número inteiro ou flutuante válido. |
| `float` | `integer, float, string` | `float` | A string deve ser um número inteiro ou flutuante válido. |
| `inet` | `string` | `string` | Endereço IPv4 ou IPv6. |
| `int` | `integer, string` | `integer` | A string deve ser um número inteiro válido de 32 bits. |
| `list` | `list, string` | `list` | Usa a representação nativa da lista JSON. |
| `map` | `map, string` | `map` | Usa a represent ação nativa do mapa JSON. |
| `smallint` | `integer, string` | `integer` | A string deve ser um número inteiro válido de 16 bits. |
| `set` | `list, string` | `list` | Usa a representação nativa da lista JSON. |
| `text` | `string` | `string` | Usa \u escape de caracteres JSON. |

| Tipo de dados do JSON | Tipos de dados aceitos em declaraçõ es **`INSERT JSON`** | Tipos de dados retornados em declarações **`SELECT JSON`** | Observações |
|---|---|---|---|
| `time` | `string` | `string` | Hora do dia em formato `HH-MM-SS[ .fffffffff]` . |
| `timestamp` | `integer, string` | `string` | Um carimbo de data/ hora. As constante s de string permitem que você armazene carimbos de data/ hora como datas. Os carimbos de data com formato `YYYY-MM-D D HH:MM:SS.SSS` são retornados. |
| `timeuuid` | `string` | `string` | Tipo 1 UUID. Consulte constants para o formato UUID. |
| `tinyint` | `integer, string` | `integer` | A string deve ser um número inteiro válido de 8 bits. |
| `tuple` | `list, string` | `list` | Usa a representação nativa da lista JSON. |
| `uuid` | `string` | `string` | Consulte constants para o formato UUID. |
| `varchar` | `string` | `string` | Usa \u escape de caracteres JSON. |

| Tipo de dados do JSON | Tipos de dados aceitos em declaraçõ es **`INSERT JSON`** | Tipos de dados retornados em declarações **`SELECT JSON`** | Observações |
|---|---|---|---|
| `varint` | `integer, string` | `integer` | Comprimento variável; pode ultrapassar números inteiros de 32 bits ou 64 bits no decodific ador do lado do cliente. |

# Instruções DDL (linguagem de definição de dados) no Amazon Keyspaces

A linguagem de definição de dados (DDL) é o conjunto de instruções do Cassandra Query Language (CQL) que você usa para gerenciar estruturas de dados no Amazon Keyspaces (para Apache Cassandra), como espaços de chaves e tabelas. Você usa o DDL para criar essas estruturas de dados, modificá-las depois de criadas e removê-las quando não estiverem mais em uso. O Amazon Keyspaces executa operações de DDL de forma assíncrona. Para obter mais informações sobre como confirmar se uma operação assíncrona foi concluída, consulte the section called “Criação e exclusão assíncronas de espaços de chave e tabelas”.

As seguintes instruções DLL são compatíveis:

- CRIAR ESPAÇO DE CHAVES
- ALTERAR ESPAÇO DE CHAVES
- DESCARTAR ESPAÇO DE CHAVES
- CRIAR TABELA
- ALTER TABLE
- RESTAURAR TABELA
- DESCARTAR TABELA

Tópicos


- Keyspaces
- Tabelas


# Keyspaces

Um espaço de chaves agrupa tabelas relacionadas que são relevantes para um ou mais aplicativos. Em termos de um sistema de gerenciamento de banco de dados relacional (RDBMS), os espaços de chave são aproximadamente semelhantes aos bancos de dados, espaços de tabela ou construções similares.

> **Note**
>
> No Apache Cassandra, os espaços de chave determinam como os dados são replicados entre vários nós de armazenamento. No entanto, o Amazon Keyspaces é um serviço totalmente gerenciado: os detalhes de sua camada de armazenamento são gerenciados em seu nome. Por esse motivo, os espaços de chave no Amazon Keyspaces são apenas estruturas lógicas e não estão relacionados ao armazenamento físico subjacente.

Para informações sobre limites e restrições de cota para os espaços de chaves do Amazon Keyspaces, consulte Cotas.

Declarações para espaços chave

- CRIAR ESPAÇO DE CHAVES
- ALTERAR ESPAÇO DE CHAVES
- DESCARTAR ESPAÇO DE CHAVES

## CRIAR ESPAÇO DE CHAVES

Use a instrução `CREATE KEYSPACE` para criar um novo espaço de chaves.

Sintaxe

```
create_keyspace_statement ::=
    CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name
```

```
    WITH options
```

Em que:

- *keyspace_name* é o nome do espaço de chaves a ser criado.
- opções são uma ou mais das seguintes:
  - REPLICATION: um mapa que indica a estratégia de replicação para o espaço de chaves:
    - SingleRegionStrategy: para um espaço de chaves de região única. (Obrigatório)
    - NetworkTopologyStrategy— Especifique pelo menos dois e até seis Regiões da AWS. O fator de replicação para cada região é três. (Optional)
  - DURABLE_WRITES: as gravações no Amazon Keyspaces são sempre duráveis, portanto, essa opção não é necessária. No entanto, se especificado, o valor deve ser true.
  - TAGS: uma lista de tags de pares de chave-valor a serem anexadas ao recurso ao ser criado. (Optional)

Exemplo

Crie um espaço de chaves da seguinte forma.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class': 'SingleRegionStrategy'} and TAGS ={'key1':'val1',
 'key2':'val2'} ;
```

Para criar um espaço de chave multirregional, especifique NetworkTopologyStrategy e inclua pelo menos dois e até seis Regiões da AWS. O fator de replicação para cada região é três.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3'};
```

## ALTERAR ESPAÇO DE CHAVES

Use o ALTER KEYSPACE para adicionar ou remover tags de um espaço de chaves.

Sintaxe

```
alter_keyspace_statement ::=
    ALTER KEYSPACE keyspace_name
```

```
    [[ADD | DROP] TAGS
```

Em que:

- *keyspace_name* é o nome do espaço de chaves a ser alterado.
- TAGS: uma lista de tags de pares chave-valor a serem adicionadas ou removidas do espaço de chaves.

Exemplo

Altere um espaço de chaves da seguinte forma.

```
ALTER KEYSPACE "myGSGKeyspace" ADD TAGS {'key1':'val1', 'key2':'val2'};
```

### DESCARTAR ESPAÇO DE CHAVES

Use a instrução DROP KEYSPACE para remover um espaço de chaves, incluindo todo o seu conteúdo, como tabelas.

Sintaxe

```
drop_keyspace_statement ::=
    DROP KEYSPACE [ IF EXISTS ] keyspace_name
```

Em que:

- keyspace_name é o nome do espaço de chaves a ser eliminado.

Exemplo

```
DROP KEYSPACE "myGSGKeyspace";
```

## Tabelas

As tabelas são as principais estruturas de dados no Amazon Keyspaces. Os dados em uma tabela são organizados em linhas e colunas. Um subconjunto dessas colunas é usado para determinar o particionamento (e, em última análise, o posicionamento dos dados) por meio da especificação de uma chave de partição.

Outro conjunto de colunas pode ser definido em colunas de cluster, o que significa que elas podem participar como predicados na execução da consulta.

Por padrão, novas tabelas são criadas com capacidade de throughput sob demanda. Você pode alterar o modo de capacidade de tabelas novas e existentes. Para obter mais informações sobre os modos de throughput da capacidade de leitura/gravação, consulte the section called “Modos de capacidade de leitura/gravação”.

Para tabelas no modo provisionado, você pode configurar opcionais. `AUTOSCALING_SETTINGS` Para obter mais informações sobre o auto scaling do Amazon Keyspaces e as opções disponíveis, consulte. the section called “Usar SSL”

Para informações sobre limites e restrições de cotas para tabelas do Amazon Keyspaces, consulte Cotas.

Declarações para tabelas

- CRIAR TABELA
- ALTER TABLE
- RESTAURAR TABELA
- DESCARTAR TABELA

## CRIAR TABELA

Use a instrução do `CREATE TABLE` para criar uma nova tabela.

Sintaxe

```
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) )*
```

Em que:

- *table_name* é o nome da tabela a ser criada.
- *column_definition* consiste no seguinte:
  - *column_name*: o nome da coluna.
  - *cql_type*: um tipo de dados do Amazon Keyspaces (consulteTipos de dados).
  - *FROZEN*: designa essa coluna do tipo `collection` (por exemplo, `LIST`, `SET` ou`MAP`) como congelada. Uma coleção congelada é serializada em um único valor imutável e tratada como a `BLOB`. Para ter mais informações, consulte the section called “Tipos de coleção”.
  - *STATIC*: designa essa coluna como estática. As colunas estáticas armazenam valores que são compartilhados por todas as linhas na mesma partição.
  - *PRIMARY KEY*: designa essa coluna como a chave primária da tabela.
- *primary_key* consiste no seguinte:
  - `partition_key`
  - `clustering_columns`
- *partition_key*:
  - A chave de partição pode ser uma única coluna ou um valor composto formado por duas ou mais colunas. A parte da chave de partição da chave primária é necessária e determina como o Amazon Keyspaces armazena seus dados.
- *clustering_columns*:
  - A parte opcional da coluna de cluster da sua chave primária determina como os dados são agrupados e classificados em cada partição.

- *table_options* consiste no seguinte:
  - *CLUSTERING ORDER BY*: a ORDEM DE CLUSTER padrão em uma tabela é composta por suas chaves de cluster na direção de classificação ASC (ascendente). Especifique-o para substituir o comportamento de classificação padrão.
  - *CUSTOM_PROPERTIES*: um mapa de configurações específicas do Amazon Keyspaces.
    - `capacity_mode`: especifica o modo de capacidade de throughput de leitura/gravação da tabela. As opções são `throughput_mode:PAY_PER_REQUEST` e `throughput_mode:PROVISIONED`. O modo de capacidade provisionada requer `read_capacity_units` e `write_capacity_units` como entradas. O padrão é `throughput_mode:PAY_PER_REQUEST`.
    - `client_side_timestamps`: especifica se os carimbos de data/hora do lado do cliente estão habilitados ou desabilitados para a tabela. As opções são `{'status': 'enabled'}` e `{'status': 'disabled'}`. Se não especificado, o padrão será `status:disabled`. Depois que os carimbos de data/hora do lado do cliente forem habilitados para uma tabela, essa configuração não poderá ser desativada.
    - `encryption_specification`: especifica as opções de criptografia para criptografia em repouso. Se não especificado, o padrão será `encryption_type:AWS_OWNED_KMS_KEY`. A opção de criptografia chave gerenciada pelo cliente exige a AWS KMS chave no formato Amazon Resource Name (ARN) como entrada::`kms_key_identifier:ARN`. `kms_key_identifier:ARN`
    - `point_in_time_recovery`: especifica se a point-in-time restauração está ativada ou desativada para a tabela. As opções são `status:enabled` e `status:disabled`. Se não especificado, o padrão será `status:disabled`.
    - `replica_updates`: especifica as configurações de uma tabela multirregional que são específicas de uma. Região da AWS Para uma tabela multirregional, você pode configurar a capacidade de leitura da tabela de forma diferente por Região da AWS. Você pode fazer isso configurando os seguintes parâmetros. Para ter mais informações e exemplos, consulte the section called “Criação de uma tabela multirregional com modo de capacidade provisionada e escalabilidade automática (CQL)”.
      - `region`— A réplica Região da AWS da tabela com as seguintes configurações:
        - `read_capacity_units`
    - TTL: habilita as configurações personalizadas de tempo de vida para a tabela. Para habilitá-la, use `status:enabled`. O padrão é `status:disabled`. Depois de habilitado o TTL, você não poderá desabilitá-lo para a tabela.

- *AUTOSCALING_SETTINGS*inclui as seguintes configurações opcionais para tabelas no modo provisionado. Para ter mais informações e exemplos, consulte the section called “Crie uma nova tabela com escalabilidade automática usando CQL”.
  - `provisioned_write_capacity_autoscaling_update`:
    - `autoscaling_disabled`— Para ativar o escalonamento automático para capacidade de gravação, defina o valor como. `false` O padrão é `true`. (Optional)
    - `minimum_units`— O nível mínimo de taxa de transferência de gravação que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
    - `maximum_units`— O nível máximo de taxa de transferência de gravação que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
    - `scaling_policy`— O Amazon Keyspaces oferece suporte à política de rastreamento de alvos. A meta do auto scaling é a capacidade de gravação provisionada da tabela.
      - `target_tracking_scaling_policy_configuration`— Para definir a política de rastreamento de metas, você deve definir o valor alvo. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.
        - `target_value`— A taxa de utilização alvo da tabela. O escalonamento automático do Amazon Keyspaces garante que a proporção entre a capacidade consumida e a capacidade provisionada permaneça igual ou próxima a esse valor. Você define `target_value` como uma porcentagem. Um duplo entre 20 e 90. (Obrigatório)
        - `scale_in_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra escala na atividade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)
        - `scale_out_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra atividade de escalabilidade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)
        - `disable_scale_in`: Um `boolean` que especifica se `scale-in` está desativado ou ativado para a tabela. Esse parâmetro está desativado por padrão. Para ativar`scale-in`, defina o `boolean` valor como`FALSE`. Isso significa que a capacidade é reduzida automaticamente para uma tabela em seu nome. (Optional)
  - `provisioned_read_capacity_autoscaling_update`:

- `autoscaling_disabled`— Para ativar o escalonamento automático da capacidade de leitura, defina o valor como. `false` O padrão é `true`. (Optional)
- `minimum_units`— O nível mínimo de produtividade que a mesa deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
- `maximum_units`— O nível máximo de produtividade que a mesa deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
- `scaling_policy`— O Amazon Keyspaces oferece suporte à política de rastreamento de alvos. A meta do auto scaling é a capacidade de leitura provisionada da tabela.
  - `target_tracking_scaling_policy_configuration`— Para definir a política de rastreamento de metas, você deve definir o valor alvo. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.
    - `target_value`— A taxa de utilização alvo da tabela. O escalonamento automático do Amazon Keyspaces garante que a proporção entre a capacidade consumida e a capacidade provisionada permaneça igual ou próxima a esse valor. Você define `target_value` como uma porcentagem. Um duplo entre 20 e 90. (Obrigatório)
    - `scale_in_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra escala na atividade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)
    - `scale_out_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra atividade de escalabilidade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)
    - `disable_scale_in`: Um `boolean` que especifica se `scale-in` está desativado ou ativado para a tabela. Esse parâmetro está desativado por padrão. Para ativar`scale-in`, defina o `boolean` valor como`FALSE`. Isso significa que a capacidade é reduzida automaticamente para uma tabela em seu nome. (Optional)
- `replica_updates`: especifica as configurações Região da AWS específicas de escalonamento automático de uma tabela multirregional. Para uma tabela multirregional, você pode configurar a capacidade de leitura da tabela de forma diferente por Região da AWS. Você pode fazer isso configurando os seguintes parâmetros. Para ter mais informações e exemplos, consulte the section called “Criação de uma tabela multirregional com modo de capacidade provisionada e escalabilidade automática (CQL)”.

- `region`— A réplica Região da AWS da tabela com as seguintes configurações:
  - `provisioned_read_capacity_autoscaling_update`
    - `autoscaling_disabled`— Para ativar o escalonamento automático da capacidade de leitura da tabela, defina o valor como. `false` O padrão é `true`. (Optional)

      > **Note**
      >
      > O escalonamento automático de uma tabela multirregional precisa estar ativado ou desativado para todas as réplicas da tabela.

    - `minimum_units`— O nível mínimo de taxa de transferência de leitura que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
    - `maximum_units`— O nível máximo de taxa de transferência de leitura que a tabela deve estar sempre pronta para suportar. O valor deve estar entre 1 e a cota máxima de taxa de transferência por segundo da sua conta (40.000 por padrão).
    - `scaling_policy`— O Amazon Keyspaces oferece suporte à política de rastreamento de alvos. A meta do auto scaling é a capacidade de leitura provisionada da tabela.
      - `target_tracking_scaling_policy_configuration`— Para definir a política de rastreamento de metas, você deve definir o valor alvo. Para obter mais informações sobre metas de rastreamento e períodos de espera, consulte Políticas de escalabilidade de rastreamento de metas no Guia do usuário do Application Auto Scaling.
        - `target_value`— A taxa de utilização alvo da tabela. O escalonamento automático do Amazon Keyspaces garante que a proporção entre a capacidade de leitura consumida e a capacidade de leitura provisionada permaneça igual ou próxima a esse valor. Você define `target_value` como uma porcentagem. Um duplo entre 20 e 90. (Obrigatório)
        - `scale_in_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra escala na atividade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)
        - `scale_out_cooldown`— Um período de espera em segundos entre as atividades de escalonamento que permite que a tabela se estabilize antes que outra atividade de escalabilidade comece. Se nenhum valor for fornecido, o padrão será 0. (Optional)

      - `disable_scale_in`: Um `boolean` que especifica se `scale-in` está desativado ou ativado para a tabela. Esse parâmetro está desativado por padrão. Para ativar`scale-in`, defina o `boolean` valor como`FALSE`. Isso significa que a capacidade de leitura é reduzida automaticamente para uma tabela em seu nome. (Optional)
  - `default_time_to_live`: a configuração padrão de tempo de vida em segundos para a tabela.
  - TAGS: uma lista de tags de pares de chave-valor a serem anexadas ao recurso ao ser criado.
- *`clustering_order`* consiste no seguinte:
  - *`column_name`*: o nome da coluna.
  - *`ASC | DESC`*: define o modificador de ordem ascendente (ASC) ou descendente (DESC). Se não especificado, o pedido padrão será ASC.

Exemplo

```
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;
```

Em uma tabela que usa colunas de cluster, as colunas sem cluster podem ser declaradas como estáticas na definição da tabela. Para obter mais informações sobre as colunas estáticas, consulte the section called “Colunas estáticas”.

Exemplo

```
CREATE TABLE "my_keyspace".my_table (
                                                id int,
                                                name text,
                                                region text,
                                                division text,
                                                project text STATIC,
                                                PRIMARY KEY (id,division));
```

## ALTER TABLE

Use a instrução `ALTER TABLE` para adicionar novas colunas, adicionar tags ou alterar as propriedades personalizadas da tabela.

Sintaxe

```
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
```

Em que:

- *`table_name`* é o nome da tabela a ser alterada.
- *`column_definition`* é o nome da coluna e do tipo de dado a ser adicionado.

- *column_definition_list* é uma lista de colunas separadas por vírgulas colocadas entre parênteses.
- *table_options* consiste no seguinte:
  - *CUSTOM_PROPERTIES*: um mapa de configurações específicas para o Amazon Keyspaces.
    - `capacity_mode`: especifica o modo de capacidade de throughput de leitura/gravação da tabela. As opções são `throughput_mode:PAY_PER_REQUEST` e `throughput_mode:PROVISIONED`. O modo de capacidade provisionada requer `read_capacity_units` e `write_capacity_units` como entradas. O padrão é `throughput_mode:PAY_PER_REQUEST`.
    - `client_side_timestamps`: especifica se os carimbos de data/hora do lado do cliente estão habilitados ou desabilitados para a tabela. As opções são `{'status': 'enabled'}` e `{'status': 'disabled'}`. Se não especificado, o padrão será `status:disabled`. Depois que os carimbos de data/hora do lado do cliente forem habilitados para uma tabela, essa configuração não poderá ser desativada.
    - `encryption_specification`: especifica a opção de criptografia para criptografia em repouso. As opções são `encryption_type:AWS_OWNED_KMS_KEY` e `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. A opção de criptografia de chave gerenciada pelo cliente exige a chave AWS KMS no formato Nome do Recurso da Amazon (ARN) como entrada: `kms_key_identifier:ARN`.
    - `point_in_time_recovery`: especifica se a point-in-time restauração está ativada ou desativada para a tabela. As opções são `status:enabled` e `status:disabled`. O padrão é `status:disabled`.
    - `replica_updates`: especifica as configurações Região da AWS específicas de uma tabela multirregional. Para uma tabela multirregional, você pode configurar a capacidade de leitura da tabela de forma diferente por Região da AWS. Você pode fazer isso configurando os seguintes parâmetros. Para ter mais informações e exemplos, consulte the section called “Atualização da capacidade provisionada e das configurações de escalabilidade automática de uma tabela multirregional (CQL)”.
      - `region`— A réplica Região da AWS da tabela com as seguintes configurações:
        - `read_capacity_units`
    - `ttl`: habilita as configurações personalizadas de tempo de vida para a tabela. Para habilitá-la, use `status:enabled`. O padrão é `status:disabled`. Depois de habilitado o `ttl`, você não poderá desabilitá-lo para a tabela.

- *AUTOSCALING_SETTINGS*inclui as configurações opcionais de auto scaling para tabelas provisionadas. Para obter a sintaxe e as descrições detalhadas, consultethe section called "CRIAR TABELA". Para ver exemplos, consulte the section called "Ative o escalonamento automático em uma tabela existente usando CQL".
- `default_time_to_live`: A configuração padrão de tempo de vida em segundos para a tabela.
- *TAGS* é uma lista de tags de pares de chave-valor a serem anexadas ao recurso.

> **Note**
>
> Com ALTER TABLE, é possível alterar somente uma propriedade personalizada. Você não pode combinar mais de um comando ALTER TABLE na mesma instrução.

Exemplos

A instrução a seguir mostra como adicionar uma coluna a uma tabela existente.

```
ALTER TABLE mykeyspace.mytable ADD (ID int);
```

Essa instrução mostra como adicionar duas colunas de coleção a uma tabela existente:

- Uma coluna de coleção congelada `col_frozen_list` que contém uma coleção congelada aninhada
- Uma coluna de coleção não congelada `col_map` que contém uma coleção congelada aninhada

```
ALTER TABLE my_Table ADD(col_frozen_list FROZEN<LIST<FROZEN<SET<TEXT>>>>, col_map
 MAP<INT, FROZEN<SET<INT>>>);
```

Para alterar o modo de capacidade de uma tabela e especificar unidades de capacidade de leitura e gravação, você pode usar a seguinte declaração.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units':
 20}};
```

A declaração a seguir especifica uma chave KMS gerenciada pelo cliente para a tabela.

```
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'
                  }
         };
```

Para ativar a point-in-time restauração de uma tabela, você pode usar a instrução a seguir.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'point_in_time_recovery':
 {'status': 'enabled'}};
```

Para definir um valor padrão de tempo de vida em segundos para uma tabela, você pode usar a seguinte declaração.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Essa instrução habilita configurações personalizadas de tempo de vida para uma tabela.

```
ALTER TABLE mytable WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

## RESTAURAR TABELA

Use a instrução do `RESTORE TABLE` ou restaure uma tabela para um ponto no tempo. Essa declaração exige que a point-in-time recuperação seja ativada em uma tabela. Para ter mais informações, consulte Recuperação pontual.

Sintaxe

```
restore_table_statement ::=
    RESTORE TABLE restored_table_name FROM TABLE source_table_name
                   [ WITH table_options [ , ... ] ];
```

Em que:

- *`restored_table_name`* é o nome da tabela restaurada.
- *`source_table_name`* é o nome da tabela de origem.
- *`table_options`* consiste no seguinte:

- *restore_timestamp* é a hora do ponto de restauração no formato ISO 8601. Se não for especificado, o carimbo de data/hora atual será usado.
- *CUSTOM_PROPERTIES*: um mapa de configurações específicas para o Amazon Keyspaces.
  - `capacity_mode`: especifica o modo de capacidade de throughput de leitura/gravação da tabela. As opções são `throughput_mode:PAY_PER_REQUEST` e `throughput_mode:PROVISIONED`. O modo de capacidade provisionada requer `read_capacity_units` e `write_capacity_units` como entradas. O padrão é a configuração atual da tabela de origem.
  - `encryption_specification`: especifica a opção de criptografia para criptografia em repouso. As opções são `encryption_type:AWS_OWNED_KMS_KEY` e `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. A opção de criptografia chave gerenciada pelo cliente exige a AWS KMS chave no formato Amazon Resource Name (ARN) como entrada:. `kms_key_identifier:ARN` Para restaurar uma tabela criptografada com uma chave gerenciada pelo cliente para uma tabela criptografada com uma Chave pertencente à AWS, o Amazon Keyspaces requer acesso à AWS KMS chave da tabela de origem.
  - `point_in_time_recovery`: especifica se a point-in-time restauração está ativada ou desativada para a tabela. As opções são `status:enabled` e `status:disabled`. Diferentemente de quando você cria novas tabelas, o status padrão das tabelas restauradas é `status:enabled` porque a configuração é herdada da tabela de origem. Para desativar o PITR para tabelas restauradas, você deve definir `status:disabled` explicitamente.
  - `replica_updates`: especifica as configurações Região da AWS específicas de uma tabela multirregional. Para uma tabela multirregional, você pode configurar a capacidade de leitura da tabela de forma diferente por Região da AWS. Você pode fazer isso configurando os seguintes parâmetros.
    - `region`— A réplica Região da AWS da tabela com as seguintes configurações:
      - `read_capacity_units`
- *AUTOSCALING_SETTINGS*inclui as configurações opcionais de auto scaling para tabelas provisionadas. Para obter sintaxe e descrições detalhadas, consultethe section called “CRIAR TABELA”.
- *TAGS* é uma lista de tags de pares de chave-valor a serem anexadas ao recurso.

> **Note**
>
> As tabelas excluídas só podem ser restauradas até o momento da exclusão.

Exemplo

```
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'};
```

### DESCARTAR TABELA

Use a instrução `DROP TABLE` para remover uma tabela do espaço de chaves.

Sintaxe

```
drop_table_statement ::=
    DROP TABLE [ IF EXISTS ] table_name
```

Em que:

- `IF EXISTS` evita que `DROP TABLE` falhe se a tabela não existir. (Optional)
- *`table_name`* é o nome do banco de dados a ser descartado.

Exemplo

```
DROP TABLE "myGSGKeyspace".employees_tbl;
```

## Declarações DML (linguagem de manipulação de dados) no Amazon Keyspaces

Linguagem de manipulação de dados (DML) é o conjunto de instruções Cassandra Query Language (CQL) que você usa para gerenciar dados em tabelas do Amazon Keyspaces (para Apache Cassandra). Você usa instruções DML para adicionar, modificar ou excluir dados em uma tabela.

Você também usa instruções DML para consultar dados em uma tabela. (Observe que o CQL não oferece suporte a junções ou subconsultas.)

Tópicos


- SELECT
- INSERT
- UPDATE
- DELETE


# SELECT

Use uma instrução SELECT para consultar dados.

Sintaxe

```
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 ] )*
```

Exemplos

```
SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;

SELECT JSON name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

Para obter uma tabela que mapeia os tipos de dados codificados em JSON para os tipos de dados do Amazon Keyspaces, consulte the section called “Codificação JSON dos tipos de dados do Amazon Keyspaces”.

Usando a palavra-chave **`IN`**

A palavra-chave `IN` especifica igualdade para um ou mais valores. Ela pode ser aplicada à chave de partição e à coluna de cluster. Os resultados são retornados na ordem em que as chaves são apresentadas na declaração `SELECT`.

Exemplos

```
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;
```

Para obter mais informações sobre a palavra-chave `IN` e como o Amazon Keyspaces processa a declaração, consulte the section called “`IN SELECT` Instrução”.

Como ordenar resultados

A cláusula `ORDER BY` especifica a ordem de classificação dos resultados retornados. Ela usa como argumentos uma lista de nomes de colunas junto com a ordem de classificação de cada coluna. Você só pode especificar colunas de cluster nas cláusulas de ordenação. Colunas sem cluster não são permitidas. As opções de ordem de classificação são `ASC` para ordem de classificação crescente e `DESC` para decrescente. Se a ordem de classificação for omitida, a ordem padrão da coluna de cluster será usada. Para possíveis ordens de classificação, consulte the section called “Como ordenar resultados”.

Exemplo

```
SELECT name, id, division, manager_id FROM "myGSGKeyspace".employees_tbl WHERE id =
 '012-34-5678' ORDER BY division;
```

Ao usar `ORDER BY` com a palavra-chave `IN`, os resultados são ordenados em uma página. A reordenação completa com paginação desativada não é compatível.

TOKEN

Você pode aplicar a função `TOKEN` à coluna `PARTITION KEY` nas cláusulas `SELECT` e `WHERE`. Com a função `TOKEN`, o Amazon Keyspaces retorna linhas com base no valor do token mapeado do `PARTITION_KEY` e não no valor do `PARTITION KEY`.

As relações `TOKEN` não são compatíveis com a palavra-chave `IN`.

Exemplos

```
SELECT TOKEN(id) from my_table;

SELECT TOKEN(id) from my_table WHERE TOKEN(id) > 100 and TOKEN(id) < 10000;
```

Função TTL

Você pode usar a função `TTL` com a instrução `SELECT` para recuperar o tempo de expiração em segundos armazenado em uma coluna. Se nenhum valor de um conjunto for `TTL`, a função retorna `null`.

Exemplo

```
SELECT TTL(my_column) from my_table;
```

A função `TTL` não pode ser usada em colunas com várias células, como coleções.

Função do WRITETIME

Você pode usar a função `WRITETIME` com a instrução `SELECT` para recuperar o carimbo de data/hora armazenado como metadados para o valor de uma coluna somente se a tabela usar carimbos de data e hora do lado do cliente. Para ter mais informações, consulte Carimbos de data/hora do lado do cliente.

```
SELECT WRITETIME(my_column) from my_table;
```

A função `WRITETIME` não pode ser usada em colunas com várias células, como coleções.

> **Note**
>
> Para compatibilidade com o comportamento estabelecido do driver Cassandra, as políticas de autorização baseadas em tags não são aplicadas quando você executa operações em

tabelas do sistema usando chamadas de API do Cassandra Query Language (CQL) por meio de drivers e ferramentas de desenvolvedor do Cassandra. Para ter mais informações, consulte the section called “ Acesso a recursos do Amazon Keyspaces com base em tags”.

## INSERT

Use a instrução `INSERT` para adicionar uma linha a uma tabela.

Sintaxe

```
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 )* ')'
```

Exemplo

```
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') ;
```

Atualizar parâmetros

`INSERT` compatível com os seguintes valores como `update_parameter`:

- `TTL`: um valor de tempo em segundos. O valor máximo configurável é 630.720.000 segundos, o que equivale a 20 anos.
- `TIMESTAMP`: um valor `bigint` que representa o número de microssegundos desde a hora base padrão conhecida como epoch: 1º de janeiro de 1970 às 00:00:00 GMT. Um carimbo de data/hora no Amazon Keyspaces deve estar entre o intervalo de 2 dias no passado e 5 minutos no futuro.

Exemplo

```
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;
```

Suporte para JSON

Para obter uma tabela que mapeia os tipos de dados codificados em JSON para os tipos de dados do Amazon Keyspaces, consulte the section called “Codificação JSON dos tipos de dados do Amazon Keyspaces”.

Você pode usar a palavra-chave `JSON` para inserir um mapa codificado em `JSON` como uma única linha. Para colunas que existem na tabela, mas são omitidas na instrução de inserção do JSON, use `DEFAULT UNSET` para preservar os valores existentes. Use `DEFAULT NULL` para gravar um valor NULL em cada linha de colunas omitidas e sobrescrever os valores existentes (taxas de gravação padrão se aplicam). `DEFAULT NULL`é a opção padrão.

Exemplo

```
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"}';
```

Se os dados JSON contiverem chaves duplicadas, o Amazon Keyspaces armazenará o último valor da chave (semelhante ao Apache Cassandra). No exemplo a seguir, onde está a chave duplicada `id`, o valor `234-56-7890` é usado.

Exemplo

```
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

Use a instrução UPDATE para modificar uma linha em uma tabela.

Sintaxe

```
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
```

Exemplo

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale = 5 WHERE id = '567-89-0123' AND
 division = 'Marketing' ;
```

Para incrementar um `counter`, use a sintaxe a seguir. Para ter mais informações, consulte the section called “Contadores”.

```
UPDATE ActiveUsers SET counter = counter + 1  WHERE user = A70FE1C0-5408-4AE3-
BE34-8733E5K09F14 AND action = 'click';
```

Atualizar parâmetros

UPDATE compatível com os seguintes valores como `update_parameter`:

- TTL: um valor de tempo em segundos. O valor máximo configurável é 630.720.000 segundos, o que equivale a 20 anos.

- `TIMESTAMP`: um valor `bigint` que representa o número de microssegundos desde a hora base padrão conhecida como epoch: 1º de janeiro de 1970 às 00:00:00 GMT. Um carimbo de data/hora no Amazon Keyspaces deve estar entre o intervalo de 2 dias no passado e 5 minutos no futuro.

Exemplo

```
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

Use a instrução `DELETE` para remover uma linha de uma tabela.

Sintaxe

```
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
```

Em que:

- *`table_name`* é a tabela que contém a linha a ser excluída.

Exemplo

```
DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive' ;
```

`DELETE` suporta o seguinte valor como `update_parameter`:

- `TIMESTAMP`: um valor `bigint` que representa o número de microssegundos desde a hora base padrão conhecida como epoch: 1º de janeiro de 1970 às 00:00:00 GMT.

# Funções integradas no Amazon Keyspaces

O Amazon Keyspaces (para Apache Cassandra) oferece suporte a uma variedade de funções incorporadas que você pode usar nas instruções do Cassandra Query Language (CQL).

Tópicos


- Funções escalares


## Funções escalares

Uma função escalar executa um cálculo em um único valor e retorna o resultado como um valor único. O Amazon Keyspaces oferece suporte às seguintes funções escalares.

| Função | Descrição |
|---|---|
| `blobAsType` | Retorna um valor do tipo de dados especific ado. |
| `cast` | Converte um tipo de dados nativo em outro tipo de dados nativo. |
| `currentDate` | Retorna a data/hora atual como uma data. |
| `currentTime` | Retorna data/hora atual como hora. |
| `currentTimestamp` | Retorna o tempo atual em um carimbo de data/ hora. |
| `currentTimeUUID` | Retorna a data/hora atual como um `timeuuid`. |
| `fromJson` | Converte a string JSON no tipo de dados da coluna selecionada. |
| `maxTimeuuid` | Retorna o maior `timeuuid` possível para carimbo de data/hora ou string de data. |

| Função | Descrição |
| --- | --- |
| `minTimeuuid` | Retorna o menor `timeuuid` possível para carimbo de data/hora ou string de data. |
| `now` | Retorna um novo `timeuuid` exclusivo. Apoiado por declarações `INSERT`, `UPDATE` e `DELETE`, e como parte da cláusula `WHERE` nas declarações `SELECT`. |
| `toDate` | Converte um `timeuuid` ou carimbo de data/hora em um tipo de data. |
| `toJson` | Retorna o valor da coluna selecionada no formato JSON. |
| `token` | Retorna o valor de hash da chave de partição. |
| `toTimestamp` | Converte um `timeuuid` ou uma data em um carimbo de data/hora. |
| `TTL` | Retorna o tempo de expiração em segundos para uma coluna. |
| `typeAsBlob` | Converte o tipo de dados especificado em um `blob`. |
| `toUnixTimestamp` | Converte um `timeuuid` ou carimbo de data/hora em um `bigInt`. |
| `uuid` | Retorna um UUID randomizado da versão 4. Apoiado por declarações `INSERT`, `UPDATE` e `DELETE`, e como parte da cláusula `WHERE` nas declarações `SELECT`. |
| `writetime` | Retorna o carimbo de data/hora do valor da coluna especificada. |

| Função | Descrição |
| --- | --- |
| `dateOf` | (Obsoleto) Extrai o carimbo de data/hora de um `timeuuid` e retorna o valor como uma data. |
| `unixTimestampOf` | (Obsoleto) Extrai o carimbo de data/hora de um `timeuuid` e retorna o valor como um carimbo de data/hora inteiro bruto de 64 bits. |

# Cotas para Amazon Keyspaces (para Apache Cassandra)

Esta seção descreve as cotas atuais e os valores padrão do Amazon Keyspaces (para Apache Cassandra).

Tópicos


- Service Quotas do Amazon Keyspaces
- Aumentar ou diminuir throughput (para tabelas provisionadas)
- Criptografia em repouso do Amazon Keyspaces


## Service Quotas do Amazon Keyspaces

A tabela a seguir contém as cotas e os valores padrão do Amazon Keyspaces (para Apache Cassandra). As informações sobre quais cotas podem ser ajustadas estão disponíveis no console Service Quotas, onde você também pode solicitar aumentos de cotas. Para obter mais informações sobre cotas, entre em contato com AWS Support.

| Cota | Descrição | Padrão do Amazon Keyspaces |
|---|---|---|
| Máximo de espaços de teclas por Região da AWS | O número máximo de espaços de chave para esse assinante por Região. Você pode ajustar esse valor padrão no console Service Quotas. | 256 |
| Máximo de tabelas por Região da AWS | O número máximo de tabelas em todos os espaços de chaves para esse assinante por Região. Você pode ajustar esse valor padrão no console Service Quotas. | 256 |
| Tamanho máx. do esquema de tabela | O tamanho máximo de um esquema de tabela. | 350 KB |

| Cota | Descrição | Padrão do Amazon Keyspaces |
| --- | --- | --- |
| Máx. de operações DDL simultâneas | O número máximo de operações DDL simultâneas permitidas para esse assinante por Região. | 50 |
| Máx. de consultas por conexão | O número máximo de consultas CQL que podem ser processadas por uma única conexão TCP de cliente por segundo. | 3000 |
| Tamanho máx. da linha | O tamanho máximo de uma linha, excluindo dados estáticos da coluna. Para obter detalhes, consulte the section called “Como calcular o tamanho da linha”. | 1 MB |
| Número máx. de colunas nas instruções `INSERT` e `UPDATE` | O número máximo de colunas permitidas nas instruções CQL `INSERT` ou `UPDATE`. Uma instrução `INSERT` ou `UPDATE` suporta até 225 colunas regulares quando a vida útil (TTL) está desativada. Se a TTL estiver ativada, até 166 colunas regulares poderão ser modificadas em uma única operação. | 225/166 |

| Cota | Descrição | Padrão do Amazon Keyspaces |
| --- | --- | --- |
| Máx. de dados estáticos por partição lógica | O tamanho máx. agregado de dados estáticos em uma partição lógica. Para obter detalhes, consulte the section called “Como calcular o tamanho estático da coluna por partição lógica”. | 1 MB |
| Máx. de subconsultas por instrução `IN SELECT` | O número máximo de subconsultas que você pode usar para a palavra-chave `IN` em uma instrução `SELECT`. Você pode ajustar esse valor padrão no console Service Quotas. | 100 |
| Número máximo de coleções congeladas aninhadas por Região da AWS | O número máximo de coleções aninhadas aceitas quando você usa a palavra-chave `FROZEN` para uma coluna com um tipo de dados de coleção. Para obter mais informações sobre coleções congeladas, consulte the section called “Tipos de coleção”. Para aumentar o nível de aninhamento, entre em contato AWS Support. | 5 |

| Cota | Descrição | Padrão do Amazon Keyspaces |
| --- | --- | --- |
| Throughput máx. de leitura por segundo | O throughput máximo de leitura por segundo — unidades de solicitação de leitura (RRUs) ou unidades de capacidade de leitura (RCUs) — que pode ser alocado a uma tabela por região. Você pode ajustar esse valor padrão no console Service Quotas. | 40.000 |
| Throughput máx. de gravação por segundo | O throughput máximo de gravação por segundo — unidades de solicitação de gravação (WRUs) ou unidades de capacidade de gravação (WCUs) — que pode ser alocado a uma tabela por região. Você pode ajustar esse valor padrão no console Service Quotas. | 40.000 |
| Throughput de leitura em nível de conta (provisionada) | O número máximo de unidades de capacidade de leitura agregada (RCUs) alocadas para a conta por região. Isso é aplicável somente para tabelas no modo de capacidade de leitura/gravação provisionada. Você pode ajustar esse valor padrão no console Service Quotas. | 80.000 |

| Cota | Descrição | Padrão do Amazon Keyspaces |
| --- | --- | --- |
| Throughput de gravação em nível de conta (provisionada) | O número máximo de unidades de capacidade de gravação agregada (WCU) alocadas para a conta por região. Isso é aplicável somente para tabelas no modo de capacidade de leitura/gravação provisionada. Você pode ajustar esse valor padrão no console Service Quotas. | 80.000 |
| Número máximo de alvos escaláveis por região por conta | O número máximo de metas escaláveis para a conta por região. Uma tabela do Amazon Keyspaces conta como uma meta escalável se a escalabilidade automátic a estiver habilitada para capacidade de leitura e como outra meta escalável se a escalabilidade automátic a estiver habilitada para capacidade de gravação. Você pode ajustar esse valor padrão no console Service Quotas para o Application Auto Scaling escolhendo destinos escaláveis para o Amazon Keyspaces. | 1.500 |

| Cota | Descrição | Padrão do Amazon Keyspaces |
| --- | --- | --- |
| Tamanho máx. da chave de partição | O tamanho máximo da chave de partição composta. Até 3 bytes de armazenamento adicional são adicionados ao tamanho bruto de cada coluna incluída na chave de partição para metadados. | 2048 bytes |
| Tamanho máx. da chave de clustering | O tamanho máximo combinado de todas as colunas de clustering. Até 4 bytes de armazenamento adicional são adicionados ao tamanho bruto de cada coluna de clustering para metadados. | 850 bytes |
| Máximo de restaurações simultâneas de tabelas usando P oint-in-time Recovery (PITR) | O número máximo de restaurações simultâneas de tabelas usando PITR por assinante é 4. Você pode ajustar esse valor padrão no console Service Quotas. | 4 |
| Quantidade máxima de dados restaurados usando point-in-time recuperação (PITR) | O tamanho máximo dos dados que podem ser restaurados usando PITR em 24 horas. Você pode ajustar esse valor padrão no console Service Quotas. | 5 TB |

# Aumentar ou diminuir throughput (para tabelas provisionadas)

## Aumentar throughput provisionado

Você pode aumentar `ReadCapacityUnits` ou `WriteCapacityUnits` quantas vezes for necessário usando o console ou a `ALTER TABLE` declaração. As novas configurações não têm efeito até que a operação `ALTER TABLE` seja concluída.

Você não pode exceder suas cotas por conta ao adicionar capacidade provisionada. E você pode aumentar a capacidade provisionada de suas tabelas o quanto precisar. Para obter mais informações sobre cotas por conta, consulte a seção anterior, the section called “Service Quotas do Amazon Keyspaces”.

## Diminuir throughput provisionado

Para cada tabela em uma instrução `ALTER TABLE`, você pode diminuir `ReadCapacityUnits` ou`WriteCapacityUnits` (ou ambos). As novas configurações não têm efeito até que a operação `ALTER TABLE` seja concluída.

É permitido diminuir até quatro vezes a qualquer momento por dia. Um dia é definido pelo Tempo Universal Coordenado (UTC). Além disso, se não houver redução na última hora, será permitido fazer uma redução adicional. Isso leva o número máximo de diminuições em um dia para 27 (4 diminuições na primeira hora e 1 diminuição para cada uma das janelas subsequentes de 1 hora em um dia).

# Criptografia em repouso do Amazon Keyspaces

Você pode alterar as opções de criptografia entre AWS uma AWS KMS chave própria e uma AWS KMS chave gerenciada pelo cliente até quatro vezes em uma janela de 24 horas, por tabela, a partir de quando a tabela foi criada. Se não houve mudança nas últimas seis horas, uma alteração adicional será permitida. Isso leva o número máximo de alterações em um dia para oito vezes (quatro alterações nas primeiras seis horas e uma alteração para cada uma das janelas de seis horas subsequentes em um dia).

Você pode alterar a opção de criptografia para usar uma AWS AWS KMS chave própria sempre que necessário, mesmo que a cota anterior tenha sido esgotada.

Estas são as cotas, a menos que você solicite uma quantidade maior. Para solicitar um aumento da cota de serviço, consulte AWS Support.

# Histórico de documentos para Amazon Keyspaces (para Apache Cassandra)

A tabela a seguir descreve as alterações importantes na documentação desde a última versão do Amazon Keyspaces (para Apache Cassandra). Para receber notificações sobre atualizações dessa documentação, você poderá se inscrever em um feed RSS.

- Última atualização da documentação: 7 de fevereiro de 2024

| Alteração | Descrição | Data |
| --- | --- | --- |
| Conecte-se ao Amazon Keyspaces usando o Amazon Elastic Kubernetes Service | Agora você pode seguir um step-by-step tutorial para se conectar ao Amazon Keyspaces a partir do Amazon EKS. | 7 de fevereiro de 2024 |
| APIs de escalabilidade automática do Amazon Keyspaces para tabelas provisionadas | O Amazon Keyspaces agora oferece CQL suporte de AWS API para configurar o auto scaling com o modo de capacidade provisionada. | 23 de janeiro de 2024 |
| Suporte de replicação multirregional do Amazon Keyspaces para tabelas provisionadas | O Amazon Keyspaces agora oferece suporte ao modo de capacidade provisionada para tabelas multirregionais. | 23 de janeiro de 2024 |
| Atividade de DML do Amazon Keyspaces incluída nos registros CloudTrail | Agora você pode auditar as chamadas de API de linguagem de manipulação de dados (DML) do Amazon Keyspaces no AWS CloudTrail. | 20 de dezembro de 2023 |

| | | |
|---|---|---|
| Suporte do Amazon Keyspaces para a palavra-chave `FROZEN` | O Amazon Keyspaces agora oferece suporte à palavra-chave `FROZEN` para tipos de dados de coleção. | 15 de novembro de 2023 |
| Atualização da política gerenciada pelo Amazon Keyspaces | O Amazon Keyspaces adicionou novas permissões à política gerenciada do `AmazonKeyspacesFullAccess` para permitir que os clientes se conectem ao Amazon Keyspaces por meio de endpoints da VPC de interface acessem a instância do Amazon EC2 para atualizar a tabela `system.peers` do Amazon Keyspaces com informações de rede da VPC. | 3 de outubro de 2023 |
| Atualização da política gerenciada pelo Amazon Keyspaces | O Amazon Keyspaces criou uma nova política gerenciada `AmazonKeyspacesReadOnlyAccess_v2` para permitir que os clientes que se conectam ao Amazon Keyspaces por meio de endpoints da VPC de interface acessem a instância do Amazon EC2 para atualizar a tabela `system.peers` do Amazon Keyspaces com informações de rede da VPC. | 12 de setembro de 2023 |
| Melhores práticas para criar conexões no Amazon Keyspaces | Saiba como melhorar e otimizar as configurações de drivers de clientes no Amazon Keyspaces. | 30 de junho de 2023 |

| | | |
|---|---|---|
| Os espaços de chaves do sistema agora estão documentados para o Amazon Keyspaces | Saiba o que está armazenado nos espaços de chaves do sistema e como consultá-los para obter informações úteis no Amazon Keyspaces. | 21 de junho de 2023 |
| O Amazon Keyspaces agora oferece suporte à replicação multirregional | A replicação multirregional do Amazon Keyspaces ajuda você a manter aplicativos distribuídos globalmente, fornecendo melhor tolerância a falhas, estabilidade e resiliência. | 5 de junho de 2023 |
| Atualização da política gerenciada pelo Amazon Keyspaces | O Amazon Keyspaces adicionou novas permissões à política gerenciada `AmazonKeyspacesFullAccess` para permitir que o Amazon Keyspaces crie uma função vinculada ao serviço quando um administrador cria um espaço de chaves multirregional. | 5 de junho de 2023 |
| Suporte do Amazon Keyspaces para a palavra-chave `IN` | O Amazon Keyspaces agora oferece suporte à palavra-chave `IN` em declarações `SELECT`. | 25 de abril de 2023 |
| Acesso entre contas para Amazon Keyspaces e endpoints da VPC de interface | Saiba como implementar o acesso entre contas para o Amazon Keyspaces com endpoints da VPC. | 20 de abril de 2023 |

| | | |
|---|---|---|
| Suporte do Amazon Keyspaces para carimbos de data/hora do lado do cliente | Os carimbos de data/hora do lado do cliente do Amazon Keyspaces são carimbos de data e hora em nível de célula compatíveis com o Cassandra que ajudam aplicativos distribuídos a determinar a ordem das operações de gravação quando clientes diferentes fazem alterações nos mesmos dados. | 14 de março de 2023 |
| Conceitos básicos do Amazon Keyspaces e de endpoints da VPC de interface | Neste step-by-step tutorial, aprenda como se conectar ao Amazon Keyspaces a partir de uma VPC. | 1 de março de 2023 |
| Como otimizar os custos das tabelas do Amazon Keyspaces | As melhores práticas e orientações estão disponíveis para ajudá-lo a identificar estratégias para otimizar os custos de suas tabelas existentes do Amazon Keyspaces. | 17 de fevereiro de 2023 |
| Agora `Murmur3Partitioner` é o padrão | Agora `Murmur3Partitioner` é o particionador padrão no Amazon Keyspaces. | 17 de novembro de 2022 |
| O Amazon Keyspaces agora oferece suporte a `Murmur3Partitioner` | Agora `Murmur3Partitioner` está disponível no Amazon Keyspaces. | 9 de novembro de 2022 |

| | | |
|---|---|---|
| Suporte à atualização para strings vazias e valores de blob | Agora, o Amazon Keyspaces também oferece suporte a sequências de caracteres vazias e valores de blob como valores de colunas de cluster. | 19 de outubro de 2022 |
| O Amazon Keyspaces agora está disponível em AWS GovCloud (US) | O Amazon Keyspaces agora está disponível no AWS GovCloud (US) Region e está dentro do escopo da alta conformidade com o FedRAMP. Para obter mais informações sobre endpoints disponíveis, consulte Endpoints do FIPS na AWS GovCloud (US) Region. | 4 de agosto de 2022 |
| Monitore os custos de armazenamento de tabelas do Amazon Keyspaces com a Amazon CloudWatch | O Amazon Keyspaces agora ajuda você a monitorar e rastrear os custos de armazenamento de tabelas ao longo do tempo com a `BillableTableSizeInBytes` CloudWatch métrica. | 14 de junho de 2022 |
| O Amazon Keyspaces agora oferece suporte ao Terraform | Agora você pode usar o Terraform para realizar operações de linguagem de definição de dados (DDL) no Amazon Keyspaces. | 9 de junho de 2022 |
| Suporte à função `token` do Amazon Keyspaces | O Amazon Keyspaces agora ajuda você a otimizar as consultas de aplicativos usando a função `token`. | 19 de abril de 2022 |

| | | |
|---|---|---|
| Integração do Amazon Keyspaces com o Apache Spark | Agora, o Amazon Keyspaces ajuda você a ler e gravar dados no Apache Spark com mais facilidade usando o conector Spark Cassandra de código aberto. | 19 de abril de 2022 |
| Referência da API do Amazon Keyspaces | O Amazon Keyspaces oferece suporte a operações de plano de controle para gerenciar espaços chave e tabelas usando o SDK e. AWS AWS CLI O guia de referência da API descreve detalhadamente as operações do ambiente de gerenciamento compatíveis. | 2 de março de 2022 |
| Como solucionar problemas comuns de configuração ao usar o Amazon Keyspaces. | Saiba mais sobre como resolver problemas comuns de configuração que você pode encontrar ao usar o Amazon Keyspaces. | 22 de novembro de 2021 |
| Suporte do Amazon Keyspaces para TTL (Vida útil). | O TTL (Vida Útil) do Amazon Keyspaces ajuda você a simplificar a lógica do seu aplicativo e otimizar o preço do armazenamento ao expirar automaticamente os dados das tabelas. | 18 de outubro de 2021 |
| Como migrar dados para o Amazon Keyspaces usando o DSBulk. | Um tep-by-step tutorial para migrar dados do Apache Cassandra para o Amazon Keyspaces usando DataStax o Bulk Loader (DSBulk). | 9 de agosto de 2021 |

| | | |
|---|---|---|
| O Amazon Keyspaces oferece suporte a entradas de endpoint da VPC na tabela `system.peers` . | O Amazon Keyspaces permite que você preencha a tabela `system.peers` com as informações disponíveis do endpoint da VPC de interface para melhorar o balanceamento de carga e aumentar o throughput de leitura/gravação. | 29 de julho de 2021 |
| Atualize as políticas gerenciadas do IAM para oferecer suporte às AWS KMS chaves gerenciadas pelo cliente. | As políticas gerenciadas do IAM para Amazon Keyspaces agora incluem permissões para listar e visualizar as AWS KMS chaves gerenciadas pelo cliente disponíveis armazenadas em. AWS KMS | 1º de junho de 2021 |
| Suporte do Amazon Keyspaces para chaves gerenciadas AWS KMS pelo cliente. | O Amazon Keyspaces permite que você assuma o controle das AWS KMS chaves gerenciadas pelo cliente armazenadas AWS KMS para criptografia em repouso. | 1º de junho de 2021 |
| Suporte do Amazon Keyspaces para a sintaxe do JSON | O Amazon Keyspaces ajuda você a ler e escrever documentos JSON com mais facilidade, oferecendo suporte à sintaxe do JSON para operações INSERT e SELECT. | 21 de janeiro de 2021 |

| | | |
|---|---|---|
| Suporte ao Amazon Keyspaces para colunas estáticas | O Amazon Keyspaces agora ajuda você a atualizar e armazenar dados comuns entre várias linhas eficiente mente usando colunas estáticas. | 9 de novembro de 2020 |
| Lançamento GA do suporte do NoSQL Workbench para Amazon Keyspaces | O NoSQL Workbench é um aplicativo do lado do cliente que ajuda você a projetar e visualizar modelos de dados não relacionais para o Amazon Keyspaces com mais facilidade. Os clientes do NoSQL Workbench estão disponíveis para Windows, macOS e Linux. | 28 de outubro de 2020 |
| Lançamento prévio do suporte do NoSQL Workbench para Amazon Keyspaces | O NoSQL Workbench é um aplicativo do lado do cliente que ajuda você a projetar e visualizar modelos de dados não relacionais para o Amazon Keyspaces com mais facilidade. Os clientes do NoSQL Workbench estão disponíveis para Windows, macOS e Linux. | 5 de outubro de 2020 |

| | | |
|---|---|---|
| Novos exemplos de código para acesso programático ao Amazon Keyspaces | Continuamos adicionando exemplos de código para acesso programático ao Amazon Keyspaces. Agora, amostras estão disponíveis para drivers Java, Python, Go, C# e Perl Cassandra compatíveis com o Apache Cassandra versão 3.11.2. | 17 de julho de 2020 |
| point-in-time Recuperação do Amazon Keyspaces (PITR) | O Amazon Keyspaces agora oferece point-in-time recuperação (PITR) para ajudar a proteger suas tabelas contra operações acidentais de gravação ou exclusão, fornecendo backups contínuos dos dados da tabela. | 9 de julho de 2020 |
| Disponibilidade geral do Amazon Keyspaces | Com o Amazon Keyspaces, anteriormente conhecido durante a versão prévia como Amazon Managed Apache Cassandra Service (MCS), você pode usar o código Cassandra Query Language (CQL), os drivers Cassandra licenciados pelo Apache 2.0 e as ferramentas de desenvolvedor que você já usa atualmente. | 23 de abril de 2020 |

| | | |
|---|---|---|
| Escalabilidade automática do Amazon Keyspaces | O Amazon Keyspaces (para Apache Cassandra) se integra ao Application Auto Scaling para ajudá-lo a provisionar a capacidade de throughput de forma eficiente para cargas de trabalho variáveis em resposta ao tráfego real do aplicativo, ajustando a capacidade de throughput automaticamente. | 23 de abril de 2020 |
| Endpoints da interface VPC (Nuvem Privada Virtual) do Amazon Keyspaces | O Amazon Keyspaces oferece comunicação privada entre o serviço e sua VPC para que o tráfego de rede não saia da rede da Amazon. | 16 de abril de 2020 |
| Políticas de acesso com base em tags | Você agora pode usar tags de recursos em políticas do IAM para gerenciar o acesso ao Amazon Keyspaces. | 8 de abril de 2020 |
| Tipo de dados do contador | O Amazon Keyspaces agora ajuda você a coordenar incrementos e decréscim os nos valores das colunas usando contadores. | 7 de abril de 2020 |
| Marcar recursos | O Amazon Keyspaces agora permite que você rotule e categorize recursos usando tags. | 31 de março de 2020 |

| | | |
|---|---|---|
| AWS CloudFormation apoio | O Amazon Keyspaces agora ajuda você a automatizar a criação e o gerenciamento dos recursos da usando AWS CloudFormation. | 25 de março de 2020 |
| Suporte a perfis e políticas do IAM e autenticação SigV4 | Foram adicionadas informaçõ es sobre como você pode usar o AWS Identity and Access Management (IAM) para gerenciar permissõe s de acesso e implementar políticas de segurança para o Amazon Keyspaces e como usar o plug-in de autentica ção do DataStax Java Driver for Cassandra para acessar programaticamente o Amazon Keyspaces usando funções do IAM e identidades federadas. | 17 de março de 2020 |
| Modo de capacidade de leitura/gravação | O Amazon Keyspaces agora oferece suporte a dois modos de capacidade de throughpu t de leitura/gravação. O modo de capacidade de leitura/g ravação controla como você é cobrado por throughput de leitura e gravação e como você gerencia a capacidade de throughput. | 20 de fevereiro de 2020 |
| Lançamento inicial | Esta documentação aborda a versão inicial do Amazon Keyspaces (para Apache Cassandra). | 3 de dezembro de 2019 |

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.