



Guia do desenvolvedor

Amazon Keyspaces (para Apache Cassandra)



Amazon Keyspaces (para Apache Cassandra): Guia do desenvolvedor

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestige a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é o Amazon Keyspaces?	1
Como funciona	2
High-level arquitetura	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
Batch	11
Captura de dados de alteração (CDC)	12
Configuração do cluster	12
Conexões	12
Palavra-chave IN	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	16
Carimbos de data/hora	17
User-defined tipos (UDTs)	17
APIs, operações, funções e tipos de dados compatíveis do Cassandra	17
Suporte à API do Cassandra	18
Suporte à API do ambiente de gerenciamento do Cassandra	19
Suporte à API do plano de dados do Cassandra	20
Suporte às funções do Cassandra	20
Suporte ao tipo de dados do Cassandra	21
Níveis de consistência compatíveis com o Cassandra	22

Níveis de consistência de gravação	23
Níveis de consistência de leitura	24
Níveis de consistência não compatíveis	25
Migração para o Amazon Keyspaces	26
Migrar do Cassandra	27
Compatibilidade	28
Custos estimados	29
Estratégia de migração	38
migração online	39
Migração offline	52
Migração híbrida	54
Ferramentas de migração	58
Como carregar dados usando cqlsh	60
Carregamento de dados usando o DSBulk	73
Como acessar o Amazon Keyspaces	91
Configurar AWS Identity and Access Management	91
Inscreva-se para um Conta da AWS	91
Configurando o Amazon Keyspaces	91
Como usar o console	92
CloudShell integração de console	93
Usando AWS CloudShell	94
AWS CloudShell opções de integração	95
Pré-requisitos	95
Começando com AWS CloudShell	95
Obtendo permissões do IAM para AWS CloudShell	96
Instalando o cliente cqlsh	97
Configurando certificados TLS	97
Conectando-se ao Amazon Keyspaces	97
Usando o CQLSH com o Amazon Keyspaces	98
Dados persistentes em AWS CloudShell	98
Práticas recomendadas	98
Solução de problemas	98
Exemplo: Como interagir com o Amazon Keyspaces usando AWS CloudShell	99
Crie credenciais programáticas de acesso	100
Crie credenciais específicas do serviço	101
Crie credenciais do IAM para autenticação AWS	103

Service endpoints	112
Portas e protocolos	113
Endpoints globais	113
AWS GovCloud (US) Region Endpoints do FIPS	118
Endpoints da região da China	119
Endpoints de streaming	119
Conexão a endpoints de pilha dupla	122
Suporte a IPv6 no Amazon Keyspaces	123
Resolução de DNS no Amazon Keyspaces	124
Usar o cqlsh	124
Como usar o cqlsh-expansion	125
Como configurar manualmente as conexões cqlsh para o TLS	131
Usando o AWS CLI	132
Baixando e configurando o AWS CLI	133
Usando o AWS CLI com o Amazon Keyspaces	133
Usar a API	137
Uso de um driver de cliente Cassandra	138
Uso de um driver de cliente Java Cassandra	139
Uso de um driver de cliente Cassandra Python	156
Usando um driver de cliente Cassandra Node.js	161
Uso de um driver de cliente Cassandra .NET Core	165
Uso de um driver de cliente Cassandra Go	167
Uso de um driver de cliente Cassandra Perl	173
Configure o acesso entre contas	174
Configure o acesso entre contas em uma VPC compartilhada	175
Configure o acesso entre contas sem uma VPC compartilhada	179
Introdução	184
Pré-requisitos	185
Criar um keyspace	185
Verifique o status da criação do keyspace	189
Criar uma tabela	189
Verifique o status da criação da tabela	198
Operações de CRUD	199
Criar	200
Ler	203
Atualizar	208

Delete	209
Excluir uma tabela	211
Excluir um keyspace	213
Tutoriais e soluções	217
Conexão com endpoints da VPC	217
Pré-requisitos	218
Etapa 1: iniciar uma instância do Amazon EC2	218
Etapa 2: configurar a instância do Amazon EC2	220
Etapa 3: criar um endpoint da VPC para o Amazon Keyspaces	223
Etapa 4: configurar permissões para a conexão do endpoint da VPC	229
Etapa 5: configurar o monitoramento	233
Etapa 6: (opcional) práticas recomendadas para conexões	234
Etapa 7: (opcional) limpeza	237
Integração com Apache Spark	238
Pré-requisitos	239
Etapa 1: Configurar o Amazon Keyspaces	240
Etapa 2: Configurar o Apache Cassandra Spark Connector	241
Etapa 3: Criar o arquivo de configuração do aplicativo	243
Etapa 4: Preparar os dados de origem e a tabela de destino	246
Etapa 5: Gravar e ler dados do Amazon Keyspaces	247
Solução de problemas	250
Conexão a partir do Amazon EKS	252
Pré-requisitos	253
Etapa 1: configurar o cluster do Amazon EKS	255
Etapa 2: configurar o aplicativo	260
Etapa 3: criar a imagem do aplicativo	263
Etapa 4: implementar o aplicativo no Amazon EKS	265
Etapa 5 (opcional): limpeza	270
Exportando dados para o Amazon S3	272
Pré-requisitos	273
Etapa 1: criar o bucket do Amazon S3, baixar ferramentas e configurar o ambiente	275
Etapa 2: Configurar o AWS Glue trabalho	277
Etapa 3: Executar o AWS Glue trabalho de exportação a partir do AWS CLI	280
Etapa 4: (Opcional) Programar o trabalho de exportação	281
Etapa 5 (opcional): limpeza	283
Como gerenciar recursos de tecnologia sem servidor	285

Estimar o tamanho da linha	286
Estime o tamanho codificado das colunas	287
Estime o tamanho codificado dos valores de dados com base no tipo de dados	288
Considere o impacto dos recursos do Amazon Keyspaces no tamanho da linha	290
Escolha a fórmula certa para calcular o tamanho codificado de uma linha	290
Exemplo de cálculo do tamanho da linha	291
Estimar o consumo de capacidade	293
Estimar o consumo de capacidade das consultas de intervalo	294
Estimar o consumo de capacidade de leitura de consultas limitadas	295
Estimar o consumo da capacidade de leitura das varreduras de tabelas	296
Estimar o consumo de capacidade de LWT	296
Estimar o consumo de capacidade de colunas estáticas	297
Estimar a capacidade de uma tabela multirregional	302
Estime o consumo de capacidade com CloudWatch	303
Configurar modos read/write de capacidade	304
Configurar o modo de capacidade sob demanda	304
Configurar o modo de capacidade provisionada	307
Exibir o modo de capacidade de uma tabela	309
Alterar modo de capacidade	311
Configurar o pré-aquecimento	313
Gerencie a capacidade de throughput com o ajuste de escala automático	328
Como funciona o ajuste de escala automático do Amazon Keyspaces	329
Como o ajuste de escala automático funciona para tabelas multirregionais	331
Observações de uso	332
Configurar e atualizar políticas de ajuste de escala automático	332
Use a capacidade de expansão	347
Como trabalhar com os recursos do Amazon Keyspaces	348
espaços de chaves do sistema	349
system	350
system_schema	352
system_schema_mcs	353
system_multiregion_info	355
User-defined tipos (UDTs)	358
Configurar permissões do	359
Crie um UDT	362
Exibir UDTs	367

Excluir um UDT	370
Como trabalhar com consultas CQL	371
Usar o IN SELECT	372
Use declarações em lote	376
Ordenar resultados	382
Pagar resultados	383
Trabalhando com fluxos do CDC	384
Como funcionam os streams do CDC	385
Use fluxos CDC	389
Como trabalhar com particionadores	423
Alterar o particionador	424
Client-side carimbos de data/hora	425
Integração com AWS serviços	427
Crie uma tabela com carimbos de data/hora no lado do cliente	427
Configurar carimbos de data/hora no lado do cliente	431
Use carimbos de data/hora no lado do cliente em consultas	433
Multi-Region replicação	434
Benefícios	435
Modos de capacidade e preços	436
Como funciona	437
Observações de uso	442
Configurar a replicação em várias regiões	444
Backup e restauração com recuperação para um ponto no tempo	473
Como funciona	474
Usar a recuperação para um ponto no tempo	480
Dados expirados com vida útil	493
Integração com AWS serviços	495
Criar tabela com valor de TTL padrão	495
Atualizar valor de TTL padrão da tabela	499
Criar tabela com TTL personalizada	503
Atualizar TTL personalizada da tabela	505
Usar INSERT para definir TTL personalizada para novas linhas	507
Usar UPDATE para definir TTL personalizada para linhas e colunas	508
Trabalhando com AWS SDKs	509
Como trabalhar com tags	510
Restrições de marcação	511

Marque espaços de teclas, tabelas e fluxos	512
Crie relatórios de alocação de custos	530
Crie CloudFormation recursos	531
Amazon Keyspaces e modelos CloudFormation	532
Saiba mais sobre CloudFormation	532
NoSQL Workbench	532
Baixar	533
Introdução	534
Visualizar um modelo de dados	536
Criar um modelo de dados	540
Editar um modelo de dados	543
Confirmar um modelo de dados	545
Modelos de dados de amostra	555
Histórico de versões	556
Exemplos de código	558
Conceitos básicos	559
Olá, Amazon Keyspaces	559
Conheça os conceitos básicos	564
Ações	627
Bibliotecas e ferramentas	680
Bibliotecas e exemplos	680
Kit de ferramentas para desenvolvedores do Amazon Keyspaces (para Apache Cassandra)	680
Exemplos do Amazon Keyspaces (para Apache Cassandra)	680
AWS Plugins de autenticação Signature Version 4 (SigV4)	681
Repositórios de amostras e ferramentas para desenvolvedores em destaque	681
Buffers de protocolo do Amazon Keyspaces	681
CloudFormation modelo para criar um CloudWatch painel da Amazon para métricas do Amazon Keyspaces (para Apache Cassandra)	681
Usando o Amazon Keyspaces (para Apache Cassandra) com AWS Lambda	682
Como usar o Amazon Keyspaces (para Apache Cassandra) com Spring	682
Como usar o Amazon Keyspaces (para Apache Cassandra) com Scala	682
Usando o Amazon Keyspaces (para Apache Cassandra) com AWS Glue	682
Amazon Keyspaces (para Apache Cassandra) Linguagem de consulta Cassandra (CQL) para conversor CloudFormation	683

Auxiliares do Amazon Keyspaces (para Apache Cassandra) para driver do Apache Cassandra para Java	683
Demonstração de compressão rápida do Amazon Keyspaces (para Apache Cassandra)	683
Demonstração do codec do Amazon Keyspaces (para Apache Cassandra) e Amazon S3 ...	683
Práticas recomendadas	684
Design em NoSQL	685
NoSQL vs. RDBMS	686
Dois conceitos-chave	686
Abordagem geral	687
Conexões	688
Como funcionam	689
Como configurar conexões	690
Como configurar políticas de repetição	691
Conexões de endpoint da VPC	692
Como monitorar conexões	693
Como lidar com erros de conexão	694
Modelagem de dados	694
Design de chave de partição	695
Otimização de custos	698
Avaliar seus custos no nível da tabela	698
Avaliar o modo de capacidade de sua tabela	701
Avalie as configurações do Application Auto Scaling da sua tabela	705
Identifique seus recursos não utilizados	713
Avaliar seus padrões de uso de tabelas	718
Avaliar sua capacidade provisionada para o provisionamento do tamanho certo	719
Solução de problemas	730
Erros gerais	731
Erros gerais	731
Erros de conexão	733
Erros na conexão com um endpoint do Amazon Keyspaces	733
Erros de gerenciamento de capacidade	745
Erros de capacidade de tecnologia sem servidor	746
Erros de linguagem de definição de dados	751
Erros de linguagem de definição de dados	751
Monitoramento do Amazon Keyspaces	757
Monitoramento com CloudWatch	758

Uso de métricas do	759
Métricas e dimensões	760
Visualizar métricas do	785
Criar alarmes	786
Fazendo login com CloudTrail	786
Configurando entradas do arquivo de log no CloudTrail	787
Informações sobre DDL em CloudTrail	788
Informações de DML em CloudTrail	789
Noções básicas sobre entradas de arquivos de log do	790
Segurança	802
Proteção de dados	803
Criptografia em repouso	804
Criptografia em trânsito	826
Privacidade do tráfego entre redes	826
AWS Identity and Access Management	827
Público	828
Autenticação com identidades	828
Gerenciar o acesso usando políticas	829
Como o Amazon Keyspaces funciona com o IAM	831
Identity-based exemplos de políticas	837
AWS políticas gerenciadas	845
Solução de problemas	857
Uso de perfis vinculados ao serviço	861
Validação de conformidade	871
Resiliência	871
Segurança da infraestrutura	872
Usar VPC endpoints da interface do	873
Usando endpoints VPC de interface com fluxos	880
Análise de configuração e vulnerabilidade no Amazon Keyspaces	884
Práticas recomendadas de segurança	884
Práticas recomendadas de segurança preventiva	884
Práticas recomendadas de segurança de detecção	886
Referência da linguagem CQL	889
Elementos de idiomas	890
Identificadores	890
Constantes	890

Termos	891
Tipos de dados	891
Codificação JSON dos tipos de dados do Amazon Keyspaces	895
Instruções DDL	899
Keyspaces	900
Tabelas	904
Tipos	918
Instruções DML	921
SELECT	921
INSERT	924
UPDATE	926
DELETE	927
Built-in funções	928
Funções escalares	928
Cotas	931
Service Quotas do Amazon Keyspaces	931
Aumentar ou diminuir throughput (para tabelas provisionadas)	939
Aumentar throughput provisionado	939
Diminuir throughput provisionado	939
Criptografia em repouso do Amazon Keyspaces	939
Cotas e valores padrão para tipos definidos pelo usuário (UDTs) no Amazon Keyspaces	940
Cotas de UDT e valores padrão do Amazon Keyspaces	940
Histórico do documento	942
.....	cmlvii

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.

Com a [calculadora de preços do Amazon Keyspaces \(para Apache Cassandra\)](#) disponível no Github, você pode estimar seus custos mensais do Amazon Keyspaces com base na sua carga de trabalho atual do Apache Cassandra. Insira métricas da saída de status do nodetool do Cassandra e da configuração sem servidor pretendida para o Amazon Keyspaces para comparar os custos diretos entre as duas soluções. Observe que essa calculadora se concentra apenas nos custos operacionais do Amazon Keyspaces em comparação com sua implantação atual do Cassandra. Não inclui fatores de custo total de propriedade (TCO), como manutenção da infraestrutura, sobrecarga operacional ou custos de suporte para o Cassandra.

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

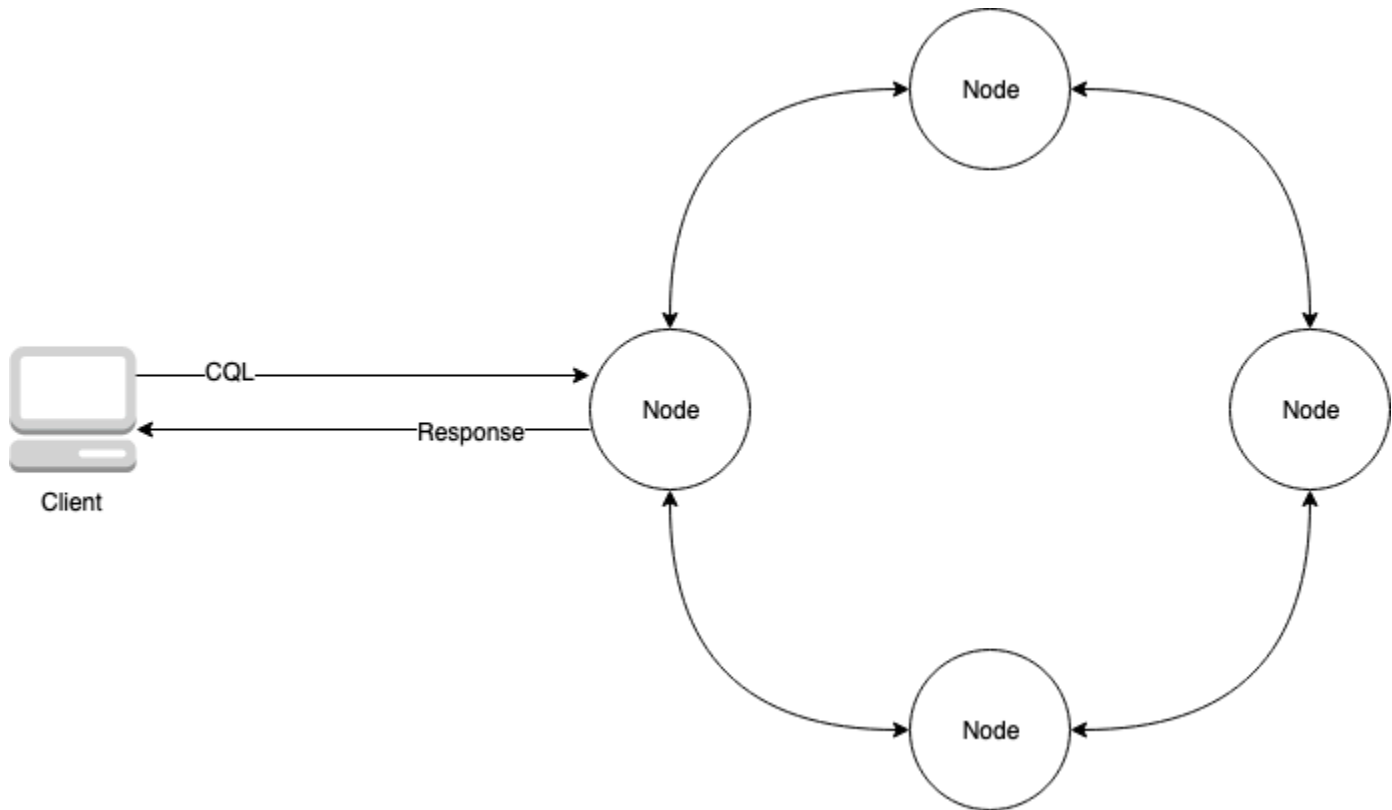
- [High-level arquitetura: Apache Cassandra versus Amazon Keyspaces](#)
- [Modelo de dados do Cassandra](#)
- [Como acessar o Amazon Keyspaces a partir de um aplicativo](#)

High-level arquitetura: 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.

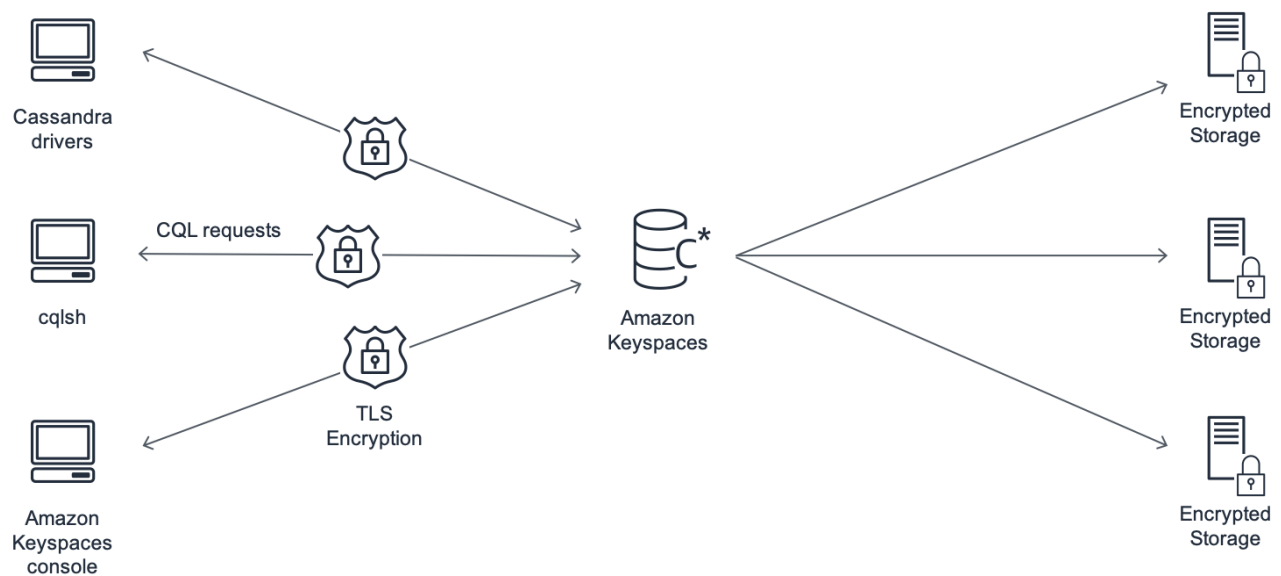
O modo de capacidade provisionada ajuda você a otimizar o preço do throughput se você tiver tráfego previsível de aplicativos e puder prever os requisitos de capacidade da sua tabela com

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 “Configurar modos read/write de capacidade”](#).

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 Management](#) endpoints 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”](#).

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 JOINS 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. Cada tabela do Cassandra deve ter uma chave primária, que é a chave exclusiva 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 I/O operações de 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 o 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, consulte [the section called “Crie credenciais do IAM para autenticação AWS”](#).

Note

Para ajudar você a começar, você pode encontrar exemplos de código de ponta a ponta para se conectar ao 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-1.amazonaws.com` e o número da porta por: 9142.
- Adicione a TLS/SSL configuração: Para obter mais informações sobre como adicionar a TLS/SSL configuração 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 latência de um dígito em milissegundos, 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, PHP, C++ Node.js, 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 obter mais informações, consulte [Como gerenciar 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).

No CQL, os dados são armazenados em tabelas, colunas e linhas. Nesse sentido, o CQL é semelhante ao Structured Query Language (SQL). Aqui estão os principais conceitos do CQL.

- Elementos CQL: os elementos fundamentais do CQL são identificadores, constantes, termos e tipos de dados.
- Data Definition Language (DDL) — As instruções DDL são usadas para gerenciar estruturas de dados como keyspaces e tabelas, que são recursos AWS no Amazon Keyspaces. As instruções DDL são operações do plano de controle em AWS.
- Linguagem de manipulação de dados (DML): as instruções DML são usadas para gerenciar dados em tabelas. As instruções DML são usadas para selecionar, inserir, atualizar e excluir dados. Essas são as operações do plano de dados em AWS.
- Built-in funções — O Amazon Keyspaces oferece suporte a uma variedade de funções escalares integradas que você pode usar em instruções CQL.

Para obter mais informações sobre CQL, consulte [Referência de idiomas CQL para Amazon Keyspaces \(para Apache Cassandra\)](#). Para ver as diferenças funcionais com o Apache Cassandra, consulte [the section called “Diferenças funcionais com o Apache Cassandra”](#).

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

- Usar o editor CQL no Console de gerenciamento da AWS.
- O uso AWS CloudShell e a expansão do [calsh](#).
- Usar um cliente `cqlsh`.
- Usar um driver de cliente Cassandra licenciado pelo Apache 2.0.

Além do CQL, você pode realizar operações de Data Definition Language (DDL) no Amazon Keyspaces usando os AWS SDKs 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 \(para Apache Cassandra\)](#).

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

Para estabelecer uma conexão com o Amazon Keyspaces, você pode usar um [AWS endpoint de serviço](#) público ou um endpoint privado usando [endpoints de interface VPC \(AWS PrivateLink\)](#) na [Amazon 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.

Conexão de endpoint da VPC na interface

Essa é uma conexão privada estabelecida usando um [endpoint da VPC na](#). 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 throughput 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, com suporte de SSDs (unidades de estado sólido). 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 throughput ou necessidades de armazenamento aumentam, o Amazon Keyspaces gerencia o gerenciamento de partições para você e provisiona automaticamente as partições adicionais necessárias.

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 é de [tecnologia sem servidor](#), então, você não precisa provisionar, corrigir nem 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 ambiente de gerenciamento do Cassandra para gerenciar configurações de clusters e nós.

O Amazon Keyspaces define automaticamente configurações como fator de replicação e nível de consistência para fornecer alta disponibilidade, durabilidade e desempenho de um dígito em milissegundos. 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](#)
- [Níveis de consistência de leitura e gravação do Apache Cassandra suportados e custos associados](#)

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](#)
- [Batch](#)
- [Captura de dados de alteração \(CDC\)](#)
- [Configuração do cluster](#)
- [Conexões](#)
- [Palavra-chave IN](#)
- [Coleções FROZEN](#)
- [Transações leves](#)
- [Balanceamento de carga](#)
- [Paginação](#)
- [Particionadores](#)
- [Instruções preparadas](#)
- [Intervalo de exclusão](#)

- [Tabelas de sistema](#)
- [Carimbos de data/hora](#)
- [User-defined tipos \(UDTs\)](#)

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](#).

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, tabelas e tipos de forma assíncrona. Para saber como monitorar o status de criação de recursos, consulte [the section called “Verifique o status da criação do keyspace”](#) e [the section called “Verifique o status da criação da tabela”](#). 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.

Batch

O Amazon Keyspaces oferece suporte aos seguintes comandos em lote:

- Lote registrado — suporta até 100 comandos por lote.
- Lote não registrado — suporta até 30 comandos por lote.

Somente DELETE comandos INSERTUPDATE,, ou são permitidos em um lote.

Captura de dados de alteração (CDC)

No Apache Cassandra, a captura de dados de alteração (CDC) copia segmentos de log de confirmação que capturam atualizações da tabela em um diretório especificado no nó. Cada registro de alteração no registro do CDC pode incluir atualizações em vários espaços de chave e tabelas. Quando uma linha é atualizada ou excluída, o log do Apache Cassandra CDC mostra somente as colunas modificadas. O registro não inclui o estado anterior dos dados antes da alteração. Para entender todos os detalhes das modificações de dados, você precisa desserializar esses registros usando ferramentas específicas do Cassandra.

Com os streams CDC do Amazon Keyspaces, você pode selecionar o tipo de informação que o stream do CDC coleta para cada linha. Por padrão, os streams CDC do Amazon Keyspaces capturam a versão da linha antes e depois da alteração. O Amazon Keyspaces CDC fornece registros de alterações desduplicados e ordenados no nível da tabela.

[No Amazon Keyspaces, cada stream do CDC é um AWS recurso com um nome de recurso da Amazon \(ARN\) e seu aplicativo pode consumir streams do CDC do Amazon Keyspaces usando a API CDC do Amazon Keyspaces usando um dos endpoints disponíveis.](#)

Para obter mais informações sobre o Amazon Keyspaces CDC, consulte [the section called “Trabalhando com fluxos do CDC”](#)

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 obter mais informações, consulte [the section called “Usar o IN SELECT”](#).

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 8 níveis de aninhamento para coleções congeladas. Para obter 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 obter mais informações, consulte [the section called “Paginar resultados”](#).

Em tabelas com colunas estáticas, tanto o Apache Cassandra quanto o Amazon Keyspaces estabelecem o valor da coluna estática da partição no início de cada página em uma consulta de várias páginas. Quando uma tabela tem grandes linhas de dados, como resultado do comportamento de paginação do Amazon Keyspaces, há maior probabilidade de que o resultado de uma operação de leitura de intervalo retorne mais páginas para o Amazon Keyspaces do que para o Apache Cassandra. Consequentemente, há uma probabilidade maior no Amazon Keyspaces de que atualizações simultâneas na coluna estática possam fazer com que o valor da coluna estática seja diferente em páginas diferentes do conjunto de resultados de leitura do intervalo.

Particionadores

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

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 obter 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

Important

A partir de 1º de julho de 2026, as operações de exclusão de intervalos passarão do processamento síncrono para o assíncrono. Após essa alteração, uma resposta bem-

sucedida confirma que sua solicitação de exclusão foi aceita, mas a exclusão ainda pode estar em andamento. Para verificar se a exclusão foi concluída, consulte os itens no intervalo especificado. Quando a consulta não retorna resultados, a exclusão é concluída.

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.

As exclusões de intervalos não são isoladas. As exclusões de linhas individuais são visíveis para outras operações enquanto uma exclusão de intervalo está em andamento.

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. Para obter mais informações, consulte [the section called “espaços de chaves do sistema”](#).

Read-only o acesso às tabelas do sistema é obrigatório e você pode controlá-lo com as políticas de acesso do IAM. Para obter mais informações, consulte [the section called “Gerenciar o 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 [the section called “Client-side carimbos de data/hora”](#).

User-defined tipos (UDTs)

O operador de desigualdade não é compatível com UDTs no Amazon Keyspaces.

Para saber como trabalhar com UDTs no Amazon Keyspaces, consulte [the section called “User-defined tipos \(UDTs\)”](#)

Para analisar quantos UDTs são suportados por espaço de chave, níveis de aninhamento suportados e outros valores e cotas padrão relacionados aos UDTs, consulte [the section called “Cotas e valores padrão para tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces”](#)

APIs, operações, funções e tipos de dados compatíveis do Cassandra

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 compatíveis.

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	Sim
SELECT	Sim
INSERT	Sim
DELETE	Sim
UPDATE	Sim
USE	Sim
CREATE TYPE	Sim
ALTER TYPE	Não
DROP TYPE	Sim

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 atributos do Cassandra não são aplicáveis.

Recurso	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

Recurso	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

Recurso	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 compatíveis, consulte o [the section called “Built-in funções”](#).

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

A tabela a seguir lista os tipos de dados do Apache Cassandra compatíveis com o Amazon Keyspaces. Para obter mais informações sobre os tipos de dados no Amazon Keyspaces, consulte [the section called “Tipos de dados”](#)

Tipo de dados	Compatível	
ascii	Sim	
bigint	Sim	
blob	Sim	
boolean	Sim	
counter	Sim	
date	Sim	
decimal	Sim	
double	Sim	
float	Sim	
frozen	Sim	
inet	Sim	
int	Sim	

Tipo de dados	Compatível	
<code>list</code>	Sim	
<code>map</code>	Sim	
<code>set</code>	Sim	
<code>smallint</code>	Sim	
<code>text</code>	Sim	
<code>time</code>	Sim	
<code>timestamp</code>	Sim	
<code>timeuuid</code>	Sim	
<code>tinyint</code>	Sim	
<code>tuple</code>	Sim	
<code>user-defined types (UDTs)</code>	Sim	
<code>uuid</code>	Sim	
<code>varchar</code>	Sim	
<code>varint</code>	Sim	

Níveis de consistência de leitura e gravação do Apache Cassandra suportados e custos associados

Os tópicos desta seção descrevem quais níveis de consistência do Apache Cassandra são compatíveis com 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 não compatíveis](#)

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-1"))
    .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	On-demand
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 não compatíveis

Os seguintes níveis de consistência não são compatíveis com o Amazon Keyspaces e resultarão em exceções.

Níveis de consistência sem suporte

Apache Cassandra	Amazon Keyspaces
EACH_QUORUM	Não suportado
QUORUM	Sem compatibilidade
ALL	Sem compatibilidade
TWO	Sem compatibilidade
THREE	Sem compatibilidade
ANY	Sem compatibilidade
SERIAL	Sem compatibilidade
LOCAL_SERIAL	Sem compatibilidade

Como migrar para o Amazon Keyspaces (para Apache Cassandra)

A migração para o Amazon Keyspaces (para Apache Cassandra) apresenta uma série de benefícios interessantes para empresas e organizações. Veja a seguir algumas das principais vantagens que tornam o Amazon Keyspaces uma opção atraente para migração.

- **Escalabilidade:** o Amazon Keyspaces foi concebido para lidar com grandes workloads e escalar perfeitamente para acomodar volumes e tráfego de dados crescentes. Com o Cassandra tradicional, o escalonamento não é realizado sob demanda e requer planejamento para futuros picos. Com o Amazon Keyspaces, é fácil aumentar ou reduzir suas tabelas verticalmente com base na demanda, garantindo que as aplicações possam lidar com picos repentinos de tráfego sem comprometer a performance.
- **Performance:** o Amazon Keyspaces oferece acesso a dados de baixa latência, permitindo que as aplicações 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 oferecido 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 com tecnologia sem servidor:** o Amazon Keyspaces está equipado com tecnologia sem 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 modelo de pagamento por solicitação oferece economia e despesas operacionais mínimas, 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, é possível armazenar dados estruturados, semiestruturados e não estruturados, o que o torna adequado para lidar com tipos de dados diversos e dinâmicos. 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 oferece 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 integra-se ao AWS Identity and Access Management para oferecer um controle de acesso minucioso. Ele fornece criptografia em repouso e em trânsito, ajudando a melhorar a segurança dos dados. O Amazon Keyspaces foi avaliado por auditores terceirizados quanto à segurança e conformidade com programas específicos, incluindo HIPAA, PCI DSS e SOC, possibilitando que você atenda aos requisitos regulatórios. Para obter mais informações, consulte [the section called “Validação de conformidade”](#).
- 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. Para obter mais informações, consulte [Monitoramento do Amazon Keyspaces](#).

Tópicos

- [Crie um plano de migração para migrar do Apache Cassandra para o Amazon Keyspaces](#)
- [Como selecionar a ferramenta certa para fazer upload ou migrar dados em massa para o Amazon Keyspaces](#)

Crie um plano de migração para migrar do Apache Cassandra para o Amazon Keyspaces

Para uma migração bem-sucedida do Apache Cassandra para o Amazon Keyspaces, recomendamos uma análise dos conceitos e das melhores práticas de migração aplicáveis, bem como uma comparação das opções disponíveis.

Este tópico descreve como o processo de migração funciona, apresentando vários conceitos-chave e as ferramentas e técnicas disponíveis para você. É possível avaliar as diferentes estratégias de migração para selecionar a mais adequada às suas necessidades.

Tópicos

- [Compatibilidade funcional](#)
- [Custos estimados do Amazon Keyspaces](#)
- [Escolher uma estratégia de migração](#)
- [Migração on-line para o Amazon Keyspaces: estratégias e melhores práticas](#)
- [Processo de migração off-line: Apache Cassandra para Amazon Keyspaces](#)
- [Usando uma solução de migração híbrida: Apache Cassandra para 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, consulte [the section called “APIs, operações, funções e tipos de dados compatíveis do Cassandra”](#). Para obter 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 [GitHub](#) e mova-o para um local que tenha acesso ao seu cluster Apache Cassandra existente.
2. O script de compatibilidade usa parâmetros semelhantes a CQLSH. Para `--host` e `--port` 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 `-username` e `-password`. Para usar o script de compatibilidade, você pode executar o seguinte comando.

```
python toolkit-compat-tool.py --host hostname or IP -u "username" -p "password" --  
port native transport port
```

Custos estimados 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 distinto read/write .

[Pensar em seus requisitos com base em tabelas se alinha aos modos de capacidade de transferência e isolamento read/write de recursos em nível de tabela do Amazon Keyspaces.](#) Com o Amazon Keyspaces, você pode definir [políticas de read/write capacidade e 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 workload no Amazon Keyspaces.

- Nome da tabela: o nome do espaço de chaves 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	Description	Média de leitura por segundo	Média de gravação por segundo	Tamanho médio da linha em bytes	Tamanho de armazenamento em GB	Detalhamento de 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 read/write solicitações por segundo (RPS). Eles permitem que você avalie os requisitos de capacidade de throughput de uma tabela do Amazon Keyspaces e estime os preços.

Como coletar métricas de tabela na tabela de origem do Cassandra

1. Determinar 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 de dados típica em um intervalo de tokens do Cassandra.



Você pode usar um script de amostragem de tamanho de linha disponível [GitHub](#) para 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 `awk` para calcular o mínimo, o máximo, 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 saber mais sobre como o tamanho da linha é calculado no Amazon Keyspaces, consulte [the section called “Estimar o tamanho da linha”](#).

2. Determinar 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-month. 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 em Console de gerenciamento da AWS.

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` para 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 do 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` da tabela e você pode dividi-lo por `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 espaço de chave 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 ajuste de escala 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 apenas pelo que usa. Em geral, o preço da read/write taxa de transferência no Amazon Keyspaces é baseado no número e no tamanho das solicitações.

Há dois modos de capacidade no Amazon Keyspaces:

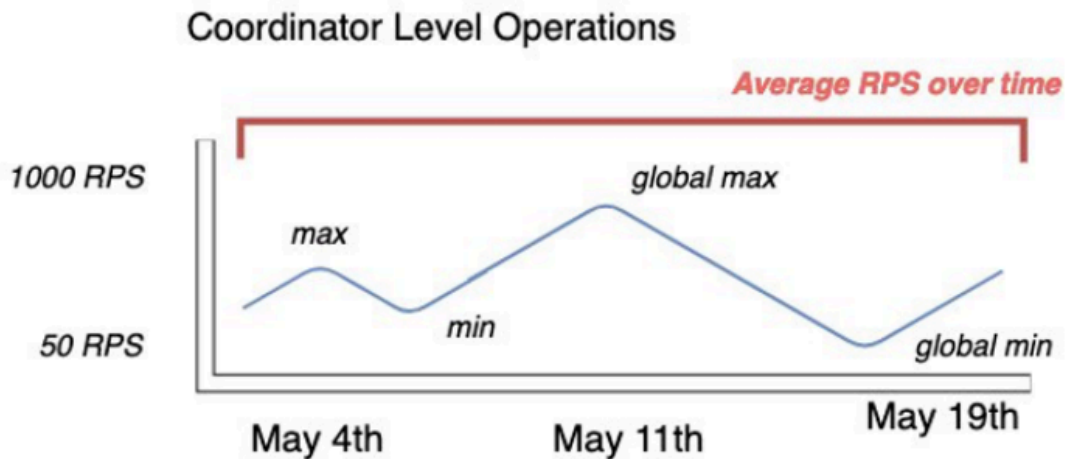
- [On-demand](#)— Essa é uma opção de cobrança flexível capaz de atender a milhares de solicitações por segundo sem a necessidade de planejamento de capacidade. Esta opção oferece preços de pagamento por solicitação para solicitações de leitura e gravação, para que você pague apenas pelo que usar.
- [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 ajuda você a gerenciar o uso do Amazon Keyspaces para permanecer em ou abaixo de uma taxa de solicitação definida para manter a previsibilidade.

O modo provisionado oferece [ajuste de escala automático](#) para ajustar automaticamente sua taxa provisionada para aumentar ou reduzir a escala verticalmente para melhorar a eficiência operacional. Para obter mais informações sobre o gerenciamento de recursos com tecnologia sem servidor, consulte [Como gerenciar recursos de tecnologia sem servidor](#).

Como você provisiona a capacidade de throughput de leitura e gravação no Amazon Keyspaces separadamente, você precisa medir a taxa de solicitações para 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 captura 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

É possível usar as métricas mostradas na tabela a seguir para observar solicitações de leituras e gravações. Observe que os nomes das métricas podem mudar com base na ferramenta de monitoramento que você está usando.

Dimensão	Métrica do Cassandra JMX
Gravações	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count</code>
Leituras	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count</code>

- 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 throughput 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 do 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 neste exemplo, presumimos que a tabela esteja no modo provisionado.

- Leituras: um RCU 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 RCUs adicionais. O número total de RCUs 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 RCUs usando consistência de leitura LOCAL_QUORUM e 1 RCU se você selecionar consistência de leitura LOCAL_ONE.

- Gravação: um WCU 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).

O número total de WCUs necessárias depende do tamanho da linha. Se você precisar gravar uma linha maior que 1 KB, a operação de gravação usará WCUs adicionais. Por exemplo, se o tamanho da sua linha for 2 KB, você precisa 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
```

O uso de eventual consistency permite que você economize até metade da capacidade de throughput em cada solicitação de leitura. Cada leitura final 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 read/write capacidade, 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 ajuste de escala 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
```

On-demand modo — 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

Você pode escolher entre as seguintes estratégias de migração ao migrar do Apache Cassandra para o Amazon Keyspaces:

- On-line: essa é uma migração ao vivo usando gravações duplas para começar a gravar novos dados no Amazon Keyspaces e no cluster Cassandra simultaneamente. Esse tipo de migração é recomendado para aplicativos que exigem zero tempo de inatividade durante a migração e consistência de leitura após gravação.

Para saber mais sobre como planejar e implementar uma estratégia de migração on-line, consulte [the section called “migração online”](#).

- Off-line: essa técnica de migração envolve a cópia de um conjunto de dados do Cassandra para o Amazon Keyspaces durante uma janela de tempo de inatividade. A migração off-line pode simplificar o processo de migração, pois não exige alterações em seu aplicativo ou resolução de conflitos entre dados históricos e novas gravações.

Para saber mais sobre como planejar uma migração off-line, consulte [the section called “Migração offline”](#).

- Híbrida: essa técnica de migração permite que as alterações sejam replicadas para o Amazon Keyspaces quase em tempo real, mas sem consistência de leitura após gravação.

Para saber mais sobre como planejar uma migração híbrida, consulte [the section called “Migração híbrida”](#).

Depois de analisar as técnicas de migração e as melhores práticas discutidas neste tópico, você pode colocar as opções disponíveis em uma árvore decisória para criar uma estratégia de migração com base em seus requisitos e recursos disponíveis.

Migração on-line para o Amazon Keyspaces: estratégias e melhores práticas

Se precisar manter a disponibilidade do aplicativo durante uma migração do Apache Cassandra para o Amazon Keyspaces, você pode preparar uma estratégia de migração on-line personalizada implementando os principais componentes discutidos neste tópico. Ao seguir essas melhores práticas para migrações on-line, você pode garantir que a disponibilidade do aplicativo e a consistência da leitura após gravação sejam mantidas durante todo o processo de migração, minimizando o impacto sobre seus usuários.

Ao criar uma estratégia de migração on-line do Apache Cassandra para o Amazon Keyspaces, você precisa considerar as seguintes etapas principais.

1. Escrever novos dados

- Proxy ZDM Dual Write para migração do Amazon Keyspaces — Use o proxy ZDM Dual Write disponível no [Github para realizar a migração sem tempo de inatividade do Apache Cassandra](#) para o Amazon Keyspaces. O ZDM Proxy executa gravações duplas sem a necessidade de refatorar aplicativos existentes e executa leituras duplas para validação da consulta.
- Gravações duplas em aplicativos: você pode implementar gravações duplas em seu aplicativo usando bibliotecas e drivers de cliente do Cassandra existentes. Designe um banco de dados como líder e o outro como seguidor. As falhas de gravação no banco de dados de seguidores são registradas em uma [fila de mensagens não entregues \(DLQ\)](#) para análise.
- Gravações duplas no nível de mensagens: como alternativa, você pode configurar sua plataforma de mensagens existente para enviar gravações para o Cassandra e para o Amazon Keyspaces usando um consumidor adicional. Eventualmente, isso cria visualizações consistentes em ambos os bancos de dados.

2. Migração de dados históricos

- Copiar dados históricos: você pode migrar dados históricos do Cassandra para o Amazon Keyspaces usando AWS Glue ou scripts personalizados de extração, transformação e

carregamento (ETL). Gerencie a resolução de conflitos entre gravações duplas e cargas em massa usando técnicas como transações leves ou registros de data e hora.

- **Uso Time-To-Live (TTL):** Para períodos mais curtos de retenção de dados, você pode usar o TTL no Cassandra e no Amazon Keyspaces para evitar o upload de dados históricos desnecessários. À medida que os dados antigos expiram no Cassandra e os novos dados são gravados por meio de gravações duplas, o Amazon Keyspaces acaba se atualizando.

3. Validando dados

- **Leituras duplas:** implemente leituras duplas dos bancos de dados Cassandra (primário) e Amazon Keyspaces (secundário), comparando os resultados de forma assíncrona. As diferenças são registradas ou enviadas para uma DLQ.
- **Leituras de amostra:** use as funções λ para amostrar e comparar dados periodicamente em ambos os sistemas, registrando quaisquer discrepâncias em um DLQ.

4. Migrando o aplicativo

- **Blue-green estratégia:** alterne seu aplicativo para tratar o Amazon Keyspaces como o principal e o Cassandra como o armazenamento de dados secundário em uma única etapa. Monitore o desempenho e reverta se surgirem problemas.
- **Implantação canário:** implemente gradualmente a migração primeiro para um subconjunto de usuários, aumentando incrementalmente o tráfego para o Amazon Keyspaces como principal até a migração completa.

5. Desativação do Cassandra

Depois que seu aplicativo for totalmente migrado para o Amazon Keyspaces e a consistência dos dados for validada, você poderá planejar a desativação do seu cluster Cassandra com base nas políticas de retenção de dados.

Ao planejar uma estratégia de migração on-line com esses componentes, você pode fazer a transição sem problemas para o serviço Amazon Keyspaces totalmente gerenciado com o mínimo de tempo de inatividade ou interrupção. As seções a seguir discutem esse componente em mais detalhes.

Tópicos

- [Escrever novos dados durante uma migração on-line](#)
- [Carregando dados históricos durante uma migração on-line](#)
- [Validando a consistência dos dados durante uma migração on-line](#)

- [Migrando o aplicativo durante uma migração on-line](#)
- [Desativação do Cassandra após uma migração on-line](#)

Escrever novos dados durante uma migração on-line

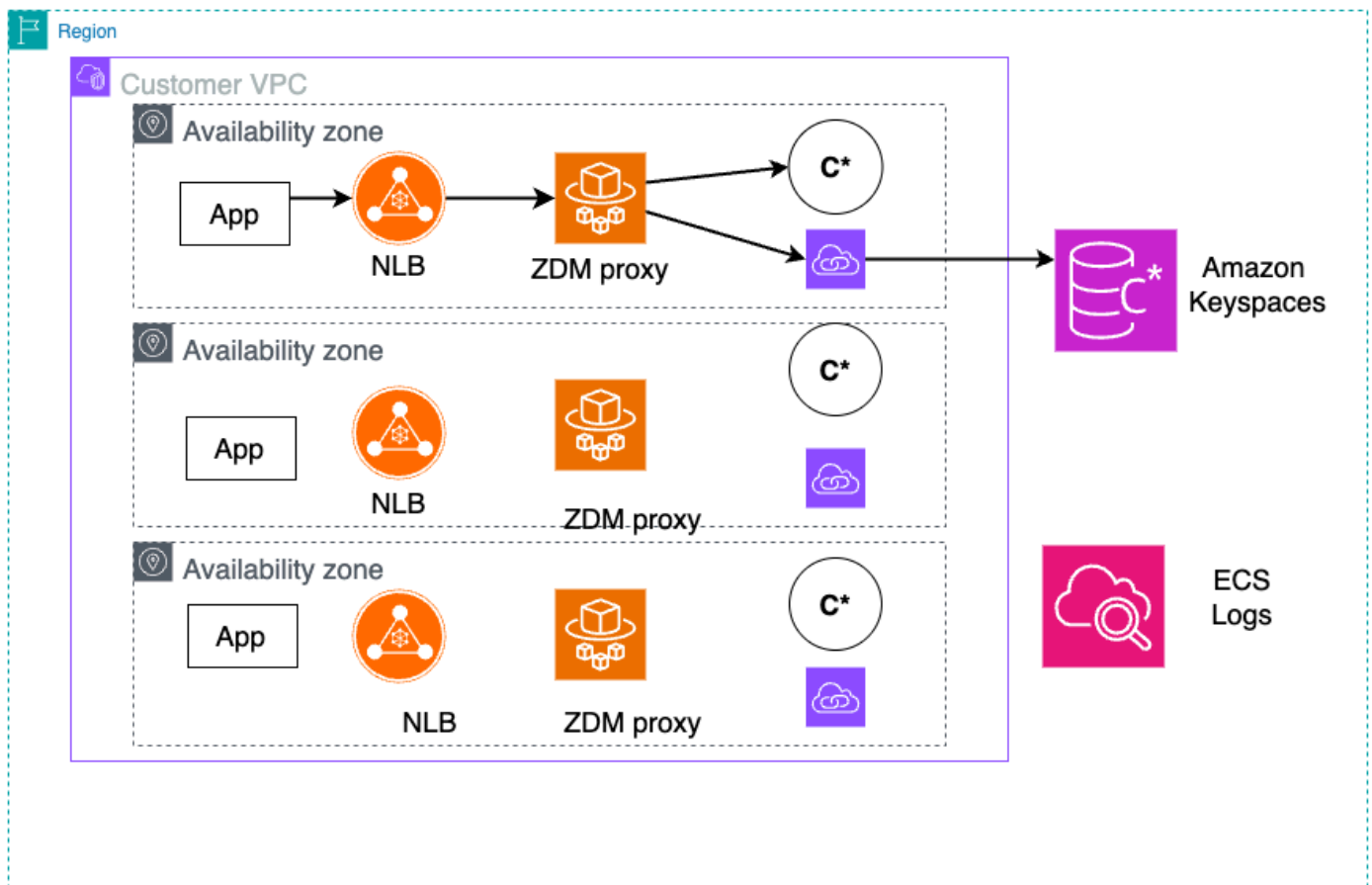
A primeira etapa em um plano de migração on-line é garantir que todos os novos dados gravados pelo aplicativo sejam armazenados nos dois bancos de dados, em seu cluster Cassandra existente e no Amazon Keyspaces. O objetivo é fornecer uma visão consistente dos dois armazenamentos de dados. Você pode fazer isso aplicando todas as novas gravações nos dois bancos de dados. Para implementar gravações duplas, considere uma das três opções a seguir.

- Proxy ZDM Dual Write para migração do Amazon Keyspaces — Usando o proxy ZDM para Amazon Keyspaces disponível no Github, você pode migrar suas cargas de trabalho do Apache [Cassandra](#) para o Amazon Keyspaces sem tempo de inatividade do aplicativo. Essa solução aprimorada implementa as AWS melhores práticas e amplia os recursos oficiais do ZDM Proxy.
 - Execute migrações on-line entre o Apache Cassandra e o Amazon Keyspaces.
 - Grave dados nas tabelas de origem e de destino simultaneamente sem refatorar os aplicativos.
 - Valide consultas por meio de operações de leitura dupla.

A solução oferece os seguintes aprimoramentos para trabalhar com o Amazon Keyspaces e com o AWS Amazon Keyspaces.

- Implantação de contêiner — Use uma imagem Docker pré-configurada do Amazon Elastic Container Registry (Amazon ECR) para implantações. VPC-accessible
- Infraestrutura como código — implante usando AWS CloudFormation modelos para configuração automática AWS Fargate.
- Compatibilidade com Amazon Keyspaces — Acesse tabelas do sistema com adaptações personalizadas para o Amazon Keyspaces.

A solução é executada no Amazon ECS com o Fargate, fornecendo escalabilidade sem servidor com base em suas demandas de carga de trabalho. Um balanceador de carga de rede distribui o tráfego de entrada do aplicativo em várias tarefas do Amazon ECS para obter alta disponibilidade.



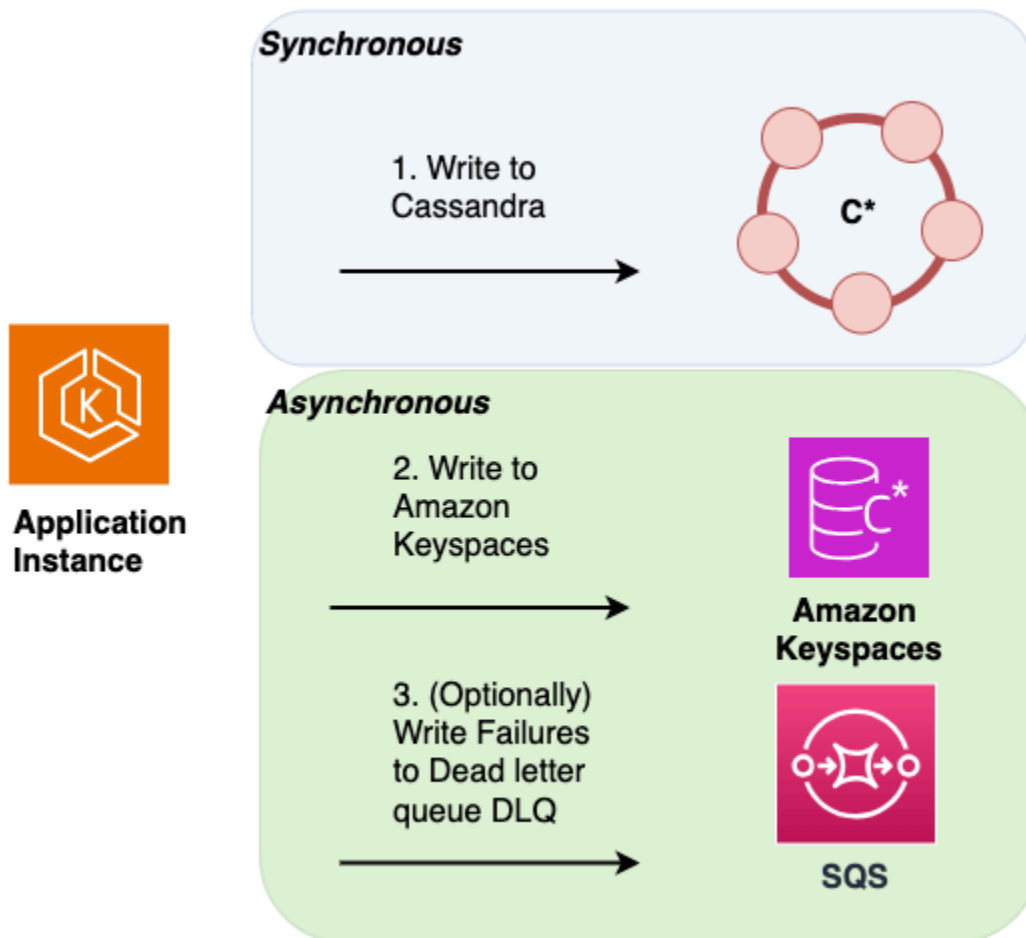
- Gravações duplas do aplicativo: você pode implementar gravações duplas com o mínimo de alterações no código do aplicativo, aproveitando as bibliotecas e os drivers existentes do cliente Cassandra. Você pode implementar gravações duplas em seu aplicativo existente ou criar uma nova camada na arquitetura para lidar com gravações duplas. Para obter mais informações e um estudo de caso de um cliente que mostra como as gravações duplas foram implementadas em um aplicativo existente, consulte o estudo de [caso de migração do Cassandra](#).

Ao implementar gravações duplas, você pode designar um banco de dados como líder e o outro banco de dados como seguidor. Isso permite que você continue gravando no banco de dados original ou principal sem permitir que falhas de gravação no banco de dados seguidor ou no banco de dados de destino interrompam o caminho crítico do seu aplicativo.

Em vez de tentar novamente gravações com falha no seguidor, você pode usar o Amazon Simple Queue Service para registrar gravações com falha em uma [fila de mensagens não entregues \(DLQ\)](#). O DLQ permite que você analise as gravações com falha no seguidor e determine por que o processamento não foi bem-sucedido no banco de dados de destino.

Para uma implementação de gravação dupla mais sofisticada, você pode seguir as AWS melhores práticas para criar uma sequência de transações locais usando o [padrão de saga](#). Um padrão saga certifica que, se uma transação falhar, a saga executa transações compensatórias para reverter as alterações do banco de dados feitas pelas transações anteriores.

Ao usar gravações duplas para uma migração on-line, você pode configurar as gravações duplas seguindo o padrão da saga para que cada gravação seja uma transação local para garantir operações atômicas em bancos de dados heterogêneos. Para obter mais informações sobre como projetar aplicativos distribuídos usando os padrões de design recomendados para o Nuvem AWS, consulte [Padrões de design, arquiteturas e implementações de nuvem](#).



- Gravações duplas no nível de mensagens: em vez de implementar gravações duplas na camada do aplicativo, você pode usar seu nível de mensagens existente para realizar gravações duplas no Cassandra e no Amazon Keyspaces.

Para fazer isso, você pode configurar um consumidor adicional em sua plataforma de mensagens para enviar gravações para os dois armazenamentos de dados. Essa abordagem fornece uma estratégia simples de baixo código usando a camada de mensagens para criar duas visualizações em ambos os bancos de dados que acabam sendo consistentes.

Carregando dados históricos durante uma migração on-line

Depois de implementar gravações duplas para garantir que novos dados sejam gravados em ambos os armazenamentos de dados em tempo real, a próxima etapa do plano de migração é avaliar quantos dados históricos você precisa copiar ou carregar em massa do Cassandra para o Amazon Keyspaces. Isso garante que tanto os novos dados quanto os dados históricos estejam disponíveis no novo banco de dados do Amazon Keyspaces antes de você migrar o aplicativo.

Dependendo dos requisitos de retenção de dados, por exemplo, quantos dados históricos você precisa preservar com base nas políticas da sua organização, você pode considerar uma das duas opções a seguir.

- Upload em massa de dados históricos — A migração de dados históricos de sua implantação atual do Cassandra para o Amazon Keyspaces pode ser realizada por meio de várias técnicas, por exemplo, usando AWS Glue scripts personalizados para extrair, transformar e carregar (ETL) os dados. Para obter mais informações sobre como usar AWS Glue para carregar dados históricos, consulte [the section called “Migração offline”](#).

Ao planejar o upload em massa de dados históricos, você precisa considerar como resolver conflitos que podem ocorrer quando novas gravações estão tentando atualizar os mesmos dados que estão sendo carregados. Espera-se que o upload em massa seja eventualmente consistente, o que significa que os dados chegarão a todos os nós eventualmente.

Se uma atualização dos mesmos dados ocorrer ao mesmo tempo devido a uma nova gravação, você deve garantir que ela não seja substituída pelo upload de dados históricos. Para garantir que você preserve as atualizações mais recentes de seus dados mesmo durante a importação em massa, você deve adicionar a resolução de conflitos nos scripts de upload em massa ou na lógica do aplicativo para gravações duplas.

Por exemplo, você pode usar [the section called “Transações leves”](#) (LWT) para comparar e definir operações. Para fazer isso, você pode adicionar um campo adicional ao seu modelo de dados que represente a hora da modificação ou o estado.

Além disso, o Amazon Keyspaces oferece suporte à função de timestamp do CassandraWRITETIME. Você pode usar os carimbos de data/hora do lado do cliente do Amazon Keyspaces para preservar os carimbos de data e hora do banco de dados de origem e implementar a resolução de conflitos entre os últimos escritores e vencedores. Para obter mais informações, consulte [the section called “Client-side carimbos de data/hora”](#).

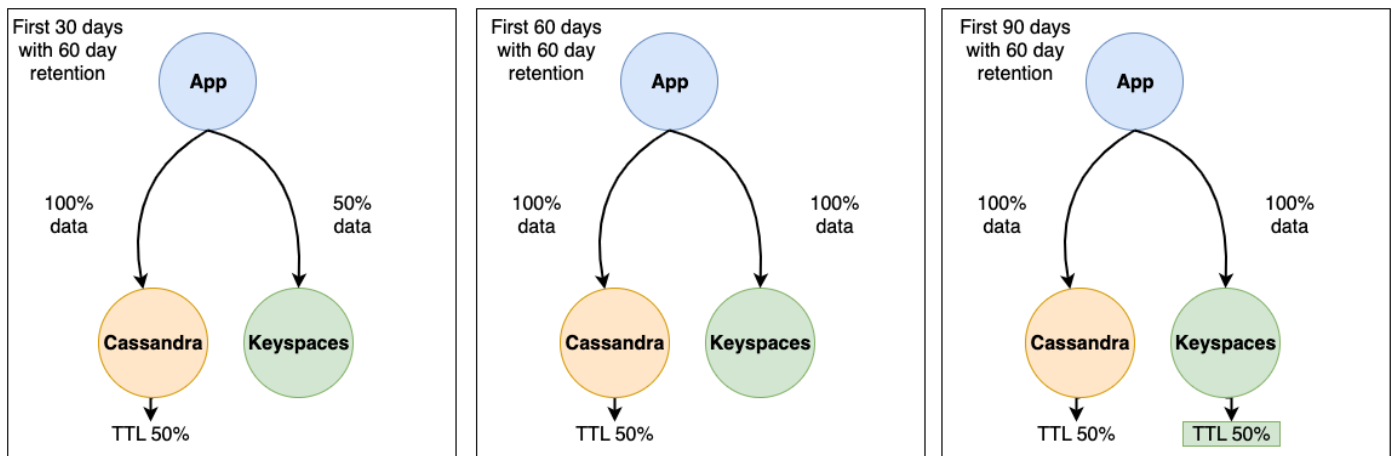
- Usando Time-to-Live (TTL) — Para períodos de retenção de dados menores que 30, 60 ou 90 dias, você pode usar o TTL no Cassandra e no Amazon Keyspaces durante a migração para evitar o upload de dados históricos desnecessários para o Amazon Keyspaces. O TTL permite definir um período de tempo após o qual os dados são removidos automaticamente do banco de dados.

Durante a fase de migração, em vez de copiar dados históricos para o Amazon Keyspaces, você pode definir as configurações de TTL para permitir que os dados históricos expirem automaticamente no sistema antigo (Cassandra), aplicando somente as novas gravações no Amazon Keyspaces usando o método de gravação dupla. Com o tempo, com os dados antigos expirando continuamente no cluster do Cassandra e os novos dados gravados usando o método de gravação dupla, o Amazon Keyspaces automaticamente se atualiza para conter os mesmos dados do Cassandra.

Essa abordagem pode reduzir significativamente a quantidade de dados a serem migrados, resultando em um processo de migração mais eficiente e simplificado. Você pode considerar essa abordagem ao lidar com grandes conjuntos de dados com diferentes requisitos de retenção de dados. Para obter mais informações sobre a TTL, consulte [the section called “Dados expirados com vida útil”](#).

Considere o exemplo a seguir de uma migração do Cassandra para o Amazon Keyspaces usando a expiração de dados TTL. Neste exemplo, definimos o TTL para ambos os bancos de dados para 60 dias e mostramos como o processo de migração progride em um período de 90 dias. Ambos os bancos de dados recebem os mesmos dados recém-gravados durante esse período usando o método de gravação dupla. Vamos analisar três fases diferentes da migração, cada fase dura 30 dias.

O funcionamento do processo de migração em cada fase é mostrado nas imagens a seguir.



1. Após os primeiros 30 dias, o cluster Cassandra e o Amazon Keyspaces estão recebendo novas gravações. O cluster Cassandra também contém dados históricos que ainda não atingiram 60 dias de retenção, o que representa 50% dos dados no cluster.

Os dados com mais de 60 dias estão sendo excluídos automaticamente no cluster do Cassandra usando TTL. Nesse ponto, o Amazon Keyspaces contém 50% dos dados armazenados no cluster Cassandra, que é composto pelas novas gravações menos os dados históricos.

2. Depois de 60 dias, tanto o cluster Cassandra quanto o Amazon Keyspaces contêm os mesmos dados gravados nos últimos 60 dias.
3. Em 90 dias, tanto o Cassandra quanto o Amazon Keyspaces contêm os mesmos dados e estão expirando os dados na mesma taxa.

Este exemplo ilustra como evitar a etapa de carregamento de dados históricos usando TTL com uma data de expiração definida como 60 dias.

Validando a consistência dos dados durante uma migração on-line

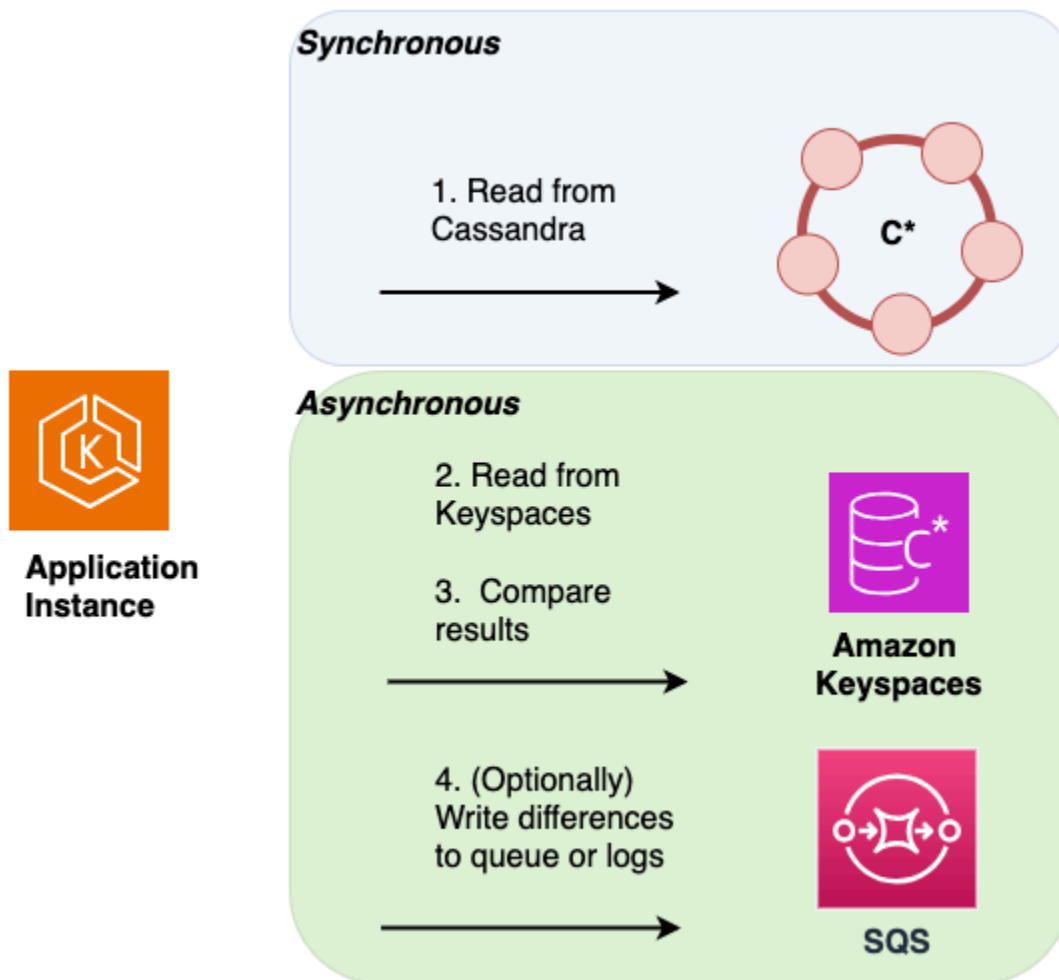
A próxima etapa do processo de migração on-line é a validação de dados. As gravações duplas estão adicionando novos dados ao seu banco de dados do Amazon Keyspaces e você concluiu a migração dos dados históricos usando upload em massa ou expiração de dados com TTL.

Agora você pode usar a fase de validação para confirmar se os dois armazenamentos de dados contêm, de fato, os mesmos dados e retornam os mesmos resultados de leitura. Você pode escolher uma das duas opções a seguir para validar se os dois bancos de dados contêm dados idênticos.

- **Leituras duplas:** para validar se o banco de dados de origem e o banco de dados de destino contêm o mesmo conjunto de dados históricos e recém-gravados, você pode implementar leituras duplas. Para fazer isso, você lê o Cassandra principal e o banco de dados secundário do Amazon Keyspaces de forma semelhante ao método de gravação dupla e compara os resultados de forma assíncrona.

Os resultados do banco de dados primário são retornados ao cliente e os resultados do banco de dados secundário são usados para validar em relação ao conjunto de resultados primário. As diferenças encontradas podem ser registradas ou enviadas para uma [fila de mensagens não entregues \(DLQ\)](#) para posterior reconciliação.

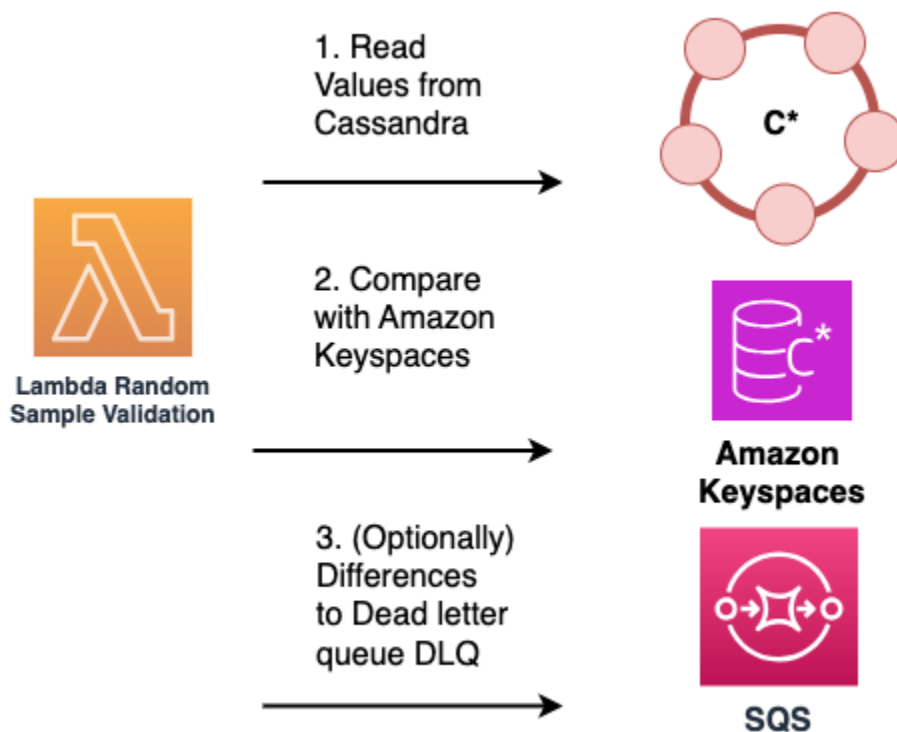
No diagrama a seguir, o aplicativo está realizando uma leitura síncrona do Cassandra (que é o armazenamento de dados primário) e uma leitura assíncrona do Amazon Keyspaces, que é o armazenamento de dados secundário.



- Exemplos de leituras — Uma solução alternativa que não exige alterações no código do aplicativo é usar AWS Lambda funções para obter amostras periódicas e aleatórias de dados do cluster Cassandra de origem e do banco de dados Amazon Keyspaces de destino.

Essas funções do Lambda podem ser configuradas para serem executadas em intervalos regulares. A função do Lambda recupera um subconjunto aleatório de dados dos sistemas de origem e destino e, em seguida, realiza uma comparação dos dados amostrados. Quaisquer discrepâncias ou incompatibilidades entre os dois conjuntos de dados podem ser registradas e enviadas para uma [fila de mensagens não entregues \(DLQ\)](#) dedicada para posterior reconciliação.

Esse processo é ilustrado no diagrama a seguir.



Migrando o aplicativo durante uma migração on-line

Na quarta fase de uma migração on-line, você está migrando seu aplicativo e fazendo a transição para o Amazon Keyspaces como armazenamento de dados primário. Isso significa que você alterna seu aplicativo para ler e gravar diretamente de e para o Amazon Keyspaces. Para garantir o mínimo de interrupção para seus usuários, esse deve ser um processo bem planejado e coordenado.

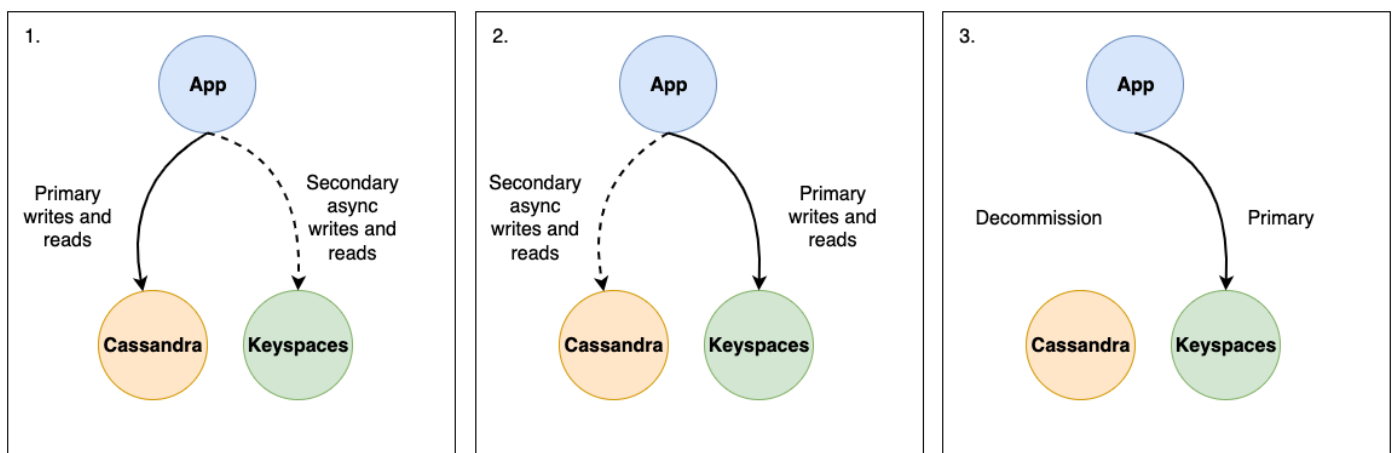
Duas soluções diferentes recomendadas para migração de aplicativos estão disponíveis: a estratégia de recorte azul esverdeado e a estratégia de corte canário. As seções a seguir descrevem essas estratégias em mais detalhes.

- **Estratégia azul e verde:** usando essa abordagem, você muda seu aplicativo para tratar o Amazon Keyspaces como o armazenamento de dados primário e o Cassandra como o armazenamento de dados secundário em uma única etapa. Você pode fazer isso usando um sinalizador de AWS AppConfig recurso para controlar a eleição dos armazenamentos de dados primários e secundários na instância do aplicativo. Para obter mais informações sobre sinalizadores de atributos, consulte [Creating a feature flag configuration profile in AWS AppConfig](#).

Depois de tornar o Amazon Keyspaces o armazenamento de dados principal, você monitora o comportamento e o desempenho do aplicativo, garantindo que o Amazon Keyspaces atenda aos seus requisitos e que a migração seja bem-sucedida.

Por exemplo, se você implementou leituras duplas para seu aplicativo, durante a fase de migração do aplicativo, você faz a transição das leituras primárias do Cassandra para o Amazon Keyspaces e as leituras secundárias do Amazon Keyspaces para o Cassandra. Após a transição, você continua monitorando e comparando os resultados conforme descrito na seção de [validação de dados](#) para garantir a consistência em ambos os bancos de dados antes de desativar o Cassandra.

Se você detectar algum problema, poderá reverter rapidamente para o estado anterior revertendo para o Cassandra como o armazenamento de dados primário. Você só prossegue para a fase de descomissionamento da migração se o Amazon Keyspaces estiver atendendo a todas as suas necessidades como armazenamento de dados primário.

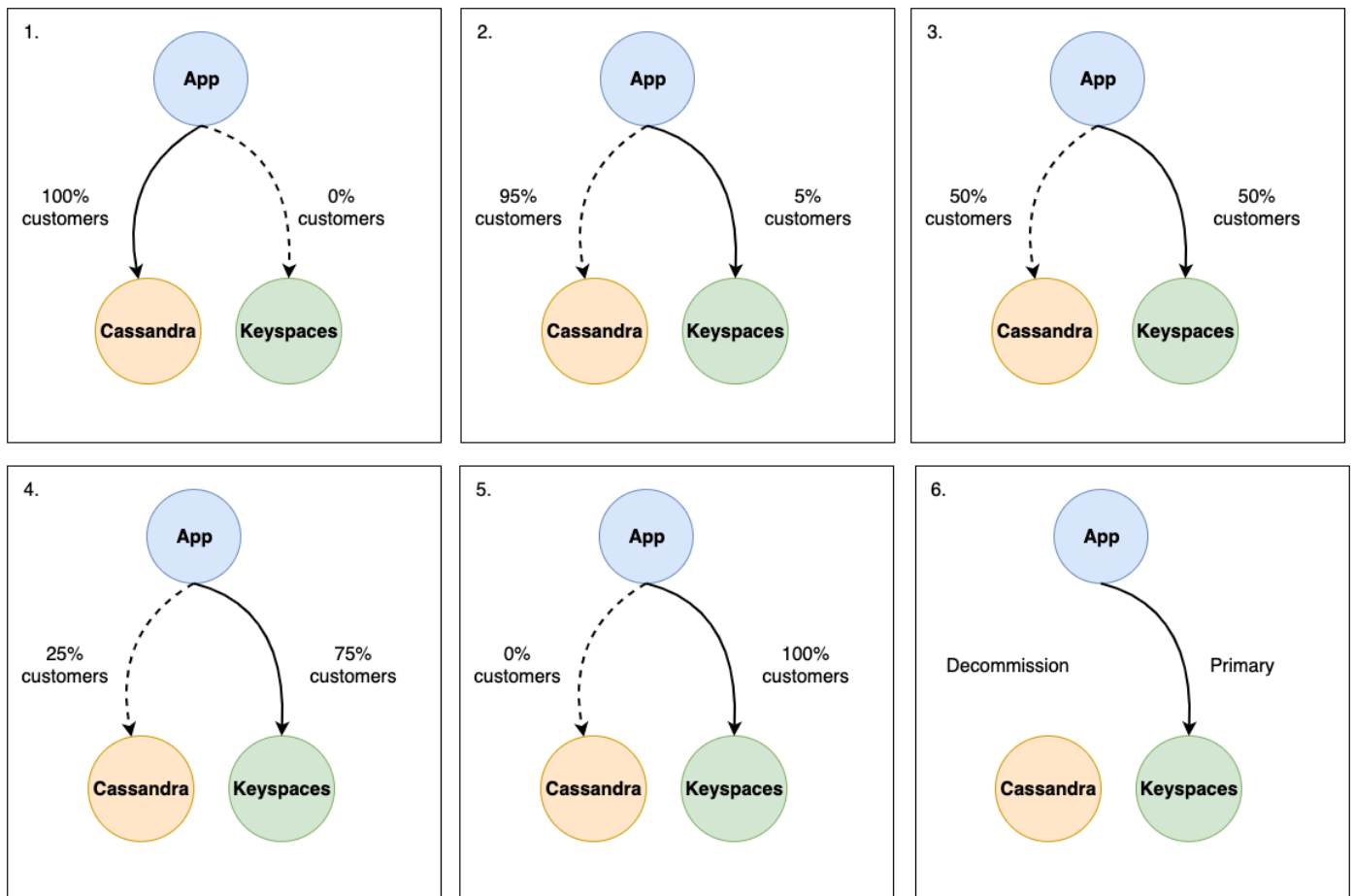


- **Estratégia canário:** nessa abordagem, você gradualmente implementa a migração para um subconjunto de seus usuários ou tráfego. Inicialmente, uma pequena porcentagem do tráfego do seu aplicativo, por exemplo, 5% de todo o tráfego é roteado para a versão usando o Amazon Keyspaces como armazenamento de dados primário, enquanto o restante do tráfego continua usando o Cassandra como armazenamento de dados primário.

Isso permite que você teste minuciosamente a versão migrada com tráfego real e monitore seu desempenho, estabilidade e investigue possíveis problemas. Se você não detectar nenhum problema, poderá aumentar incrementalmente a porcentagem de tráfego roteado para o Amazon Keyspaces até que ele se torne o armazenamento de dados principal para todos os usuários e tráfego.

Essa implantação gradual minimiza o risco de interrupções generalizadas no serviço e permite um processo de migração mais controlado. Se surgirem problemas críticos durante a implantação canário, você poderá reverter rapidamente para a versão anterior usando o Cassandra como armazenamento de dados primário para o segmento de tráfego afetado. Você só prossegue para a fase de descomissionamento da migração depois de validar que o Amazon Keyspaces processa 100% dos seus usuários e tráfego conforme o esperado.

O diagrama a seguir ilustra as etapas individuais da estratégia canário.



Desativação do Cassandra após uma migração on-line

Depois que a migração do aplicativo estiver concluída, seu aplicativo estiver totalmente em execução no Amazon Keyspaces e você tiver validado a consistência dos dados por um período de tempo, você poderá planejar a desativação do seu cluster Cassandra. Durante essa fase, você pode avaliar se os dados restantes em seu cluster do Cassandra precisam ser arquivados ou podem ser excluídos. Isso depende das políticas da sua organização para tratamento e retenção de dados.

Ao seguir essa estratégia e considerar as melhores práticas recomendadas descritas neste tópico ao planejar sua migração on-line do Cassandra para o Amazon Keyspaces, você pode garantir uma transição perfeita para o Amazon Keyspaces e, ao mesmo tempo, manter a consistência de leitura após gravação e a disponibilidade do seu aplicativo.

A migração do Apache Cassandra para o Amazon Keyspaces pode oferecer vários benefícios, incluindo redução da sobrecarga operacional, ajuste de escala automático, segurança aprimorada e uma estrutura que ajuda você a atingir suas metas de conformidade. Ao planejar uma estratégia

de migração on-line com gravações duplas, upload de dados históricos, validação de dados e implantação gradual, você pode garantir uma transição tranquila com o mínimo de interrupção para seu aplicativo e seus usuários.

A implementação da estratégia de migração on-line discutida neste tópico permite que você valide os resultados da migração, identifique e resolva quaisquer problemas e, por fim, desative sua implantação atual do Cassandra em favor do serviço Amazon Keyspaces totalmente gerenciado.

Processo de migração off-line: Apache Cassandra para Amazon Keyspaces

As migrações off-line são adequadas para quando é possível permitir tempo de inatividade para realizar a migração. É comum entre as empresas ter janelas de manutenção para patches, grandes lançamentos ou tempo 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, usamos o Amazon Simple Storage Service (Amazon S3) como uma área de preparação 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. Veja a seção a seguir uma visão geral de alto nível do processo. Você pode encontrar exemplos de código para esse processo no [Github](#).

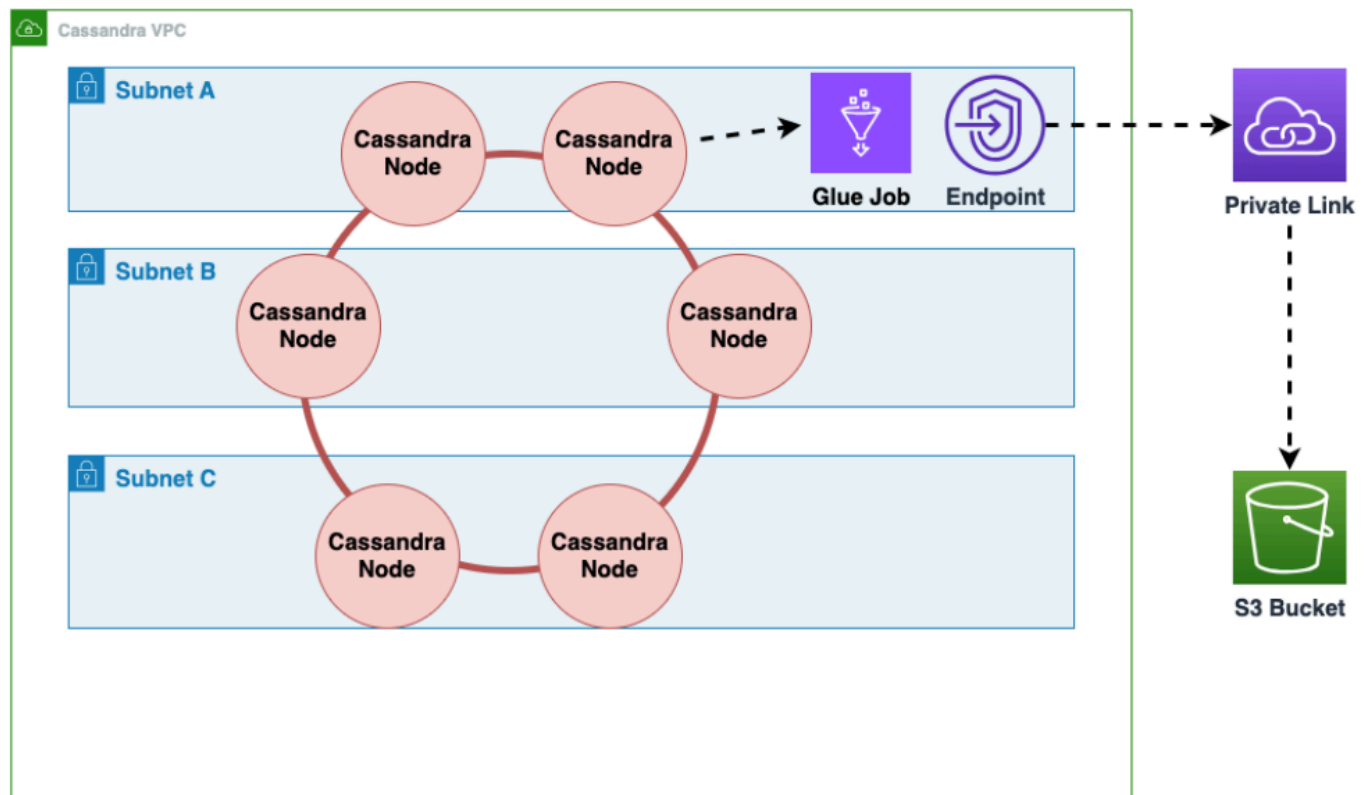
O processo de migração offline do Apache Cassandra para o Amazon Keyspaces usando o Amazon S3 requer os seguintes trabalhos. AWS Glue AWS Glue

1. Um trabalho de ETL que extrai e transforma dados de 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 de 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 `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.



Para obter mais informações sobre o processo de migração off-line, consulte o workshop [Amazon Keyspaces with AWS Glue](#)

Usando uma solução de migração híbrida: Apache Cassandra para Amazon Keyspaces

A solução de migração a seguir pode ser considerada um híbrido entre migração on-line e off-line. Com essa abordagem híbrida, os dados são gravados no banco de dados de destino quase em tempo real, sem fornecer consistência de leitura após gravação. Isso significa que os dados recém-gravados não estarão imediatamente disponíveis e que atrasos são esperados. Se você precisar de consistência de leitura após gravação, consulte [the section called “migração online”](#).

Para uma migração quase em tempo real do Apache Cassandra para o Amazon Keyspaces, você pode escolher entre dois métodos disponíveis.

- CQLReplicator: (recomendado) o CQLReplicator é um utilitário de código aberto disponível no [Github](#) que ajuda você a migrar dados do Apache Cassandra para o Amazon Keyspaces quase em tempo real.

Para determinar as gravações e atualizações a serem propagadas para o banco de dados de destino, o CQLReplicator escaneia o intervalo de tokens do Apache Cassandra e usa um AWS Glue trabalho para remover eventos duplicados e aplicar gravações e atualizações diretamente no Amazon Keyspaces.

- Captura de dados de alterações (CDC): se você estiver familiarizado com o Cassandra CDC, o recurso CDC integrado do Apache Cassandra, que permite capturar alterações copiando o log de confirmação em um diretório CDC separado, é outra opção para implementar uma migração híbrida.

Você pode fazer isso replicando as alterações de dados no Amazon Keyspaces, tornando o CDC uma opção alternativa para cenários de migração de dados.

Se você não precisar de consistência de leitura após gravação, você pode usar o CQLReplicator ou um pipeline CDC para migrar dados do Apache Cassandra para o Amazon Keyspaces com base em suas preferências e familiaridade com as ferramentas usadas em cada solução. Serviços da AWS Usar esses métodos para migrar dados quase em tempo real pode ser considerado uma abordagem híbrida de migração que oferece uma alternativa à migração on-line.

Essa estratégia é considerada uma abordagem híbrida porque, além das opções descritas neste tópico, você precisa implementar algumas etapas do progresso da migração on-line, por exemplo, a cópia histórica dos dados e as estratégias de migração de aplicativos discutidas no tópico de [migração on-line](#).

As seções a seguir detalham mais as opções de migração híbrida.

Tópicos

- [Migre dados usando o CQLReplicator](#)
- [Migre dados usando a captura de dados de alteração \(CDC\)](#)

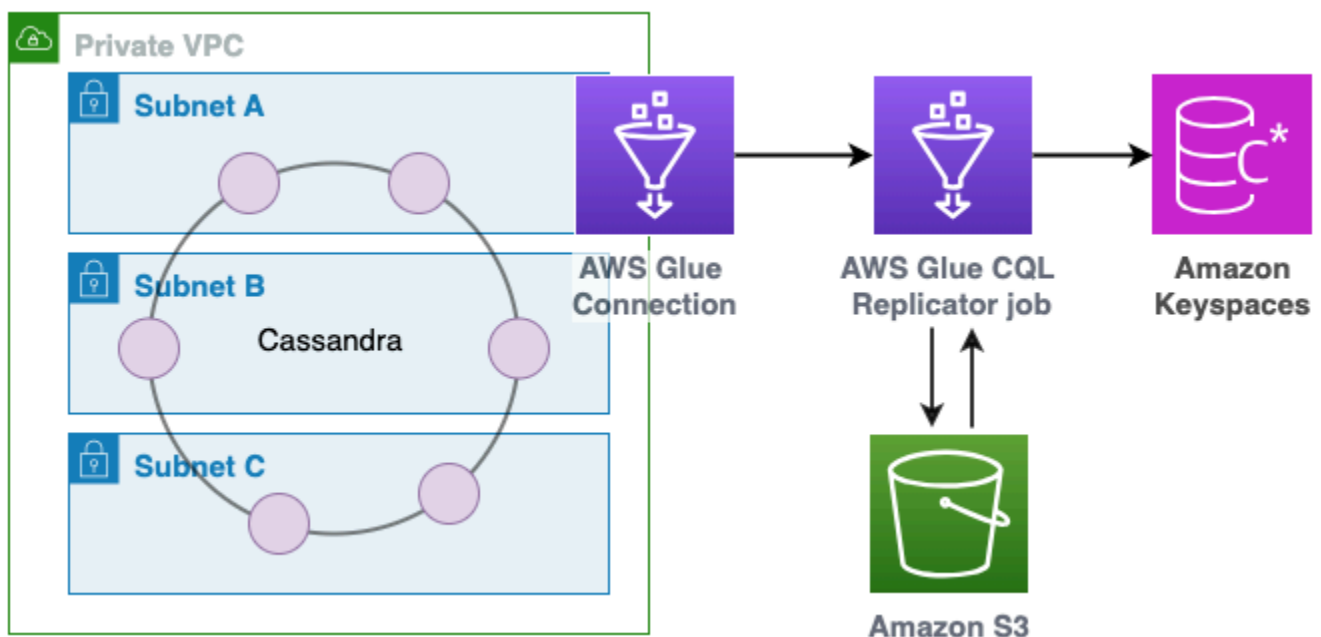
Migre dados usando o CQLReplicator

Com o [CQLReplicator](#), você pode ler dados do Apache Cassandra quase em tempo real por meio da digitalização inteligente do token ring do Cassandra usando consultas CQL. O CQLReplicator não usa o Cassandra CDC e, em vez disso, implementa uma estratégia de armazenamento em cache para reduzir as penalidades de desempenho de varreduras completas.

Para reduzir o número de gravações no destino, o CQLReplicator remove automaticamente os eventos de replicação duplicados. Com o CQLReplicator, você pode ajustar a replicação das alterações do banco de dados de origem para o banco de dados de destino, permitindo uma migração quase em tempo real dos dados do Apache Cassandra para o Amazon Keyspaces.

O diagrama a seguir mostra a arquitetura típica de uma tarefa do CQLReplicator usando AWS Glue.

1. Para permitir o acesso ao Apache Cassandra em execução em uma VPC privada, configure uma AWS Glue conexão com o tipo de conexão Rede.
2. Para remover duplicatas e ativar o armazenamento de chave em cache com a tarefa CQLReplicator, configure o Amazon Simple Storage Service (Amazon S3).
3. O fluxo de trabalho do CQLReplicator transmite alterações verificadas no banco de dados de origem diretamente para o Amazon Keyspaces.



[Para obter mais informações sobre o processo de migração usando o CQLReplicator, consulte a postagem a seguir no blog de AWS banco de dados Migrar cargas de trabalho do Cassandra para o Amazon Keyspaces usando o CQLReplicator e a orientação AWS prescritiva Migrar cargas de trabalho do Apache Cassandra para o Amazon Keyspaces usando AWS Glue](#)

Migre dados usando a captura de dados de alteração (CDC)

Se você já está familiarizado com a configuração de um pipeline de captura de dados de alteração (CDC) com o [Debezium](#), você pode usar essa opção para migrar dados para o Amazon Keyspaces como alternativa ao uso do CQLReplicator. O Debezium é uma plataforma distribuída de código aberto para CDC, projetada para monitorar um banco de dados e capturar mudanças em nível de linha de forma confiável.

O [conector Debezium para Apache Cassandra](#) carrega as alterações no Amazon Managed Streaming for Apache Kafka (Amazon MSK) para que elas possam ser consumidas e processadas por consumidores downstream que, por sua vez, gravam os dados no Amazon Keyspaces. Para obter mais informações, consulte a [orientação para a migração contínua de dados do Apache Cassandra para o Amazon Keyspaces](#).

Para resolver quaisquer possíveis problemas de consistência de dados, você pode implementar um processo com o Amazon MSK em que um consumidor compara as chaves ou partições no Cassandra com as do Amazon Keyspaces.

Para implementar essa solução com sucesso, recomendamos considerar o seguinte.

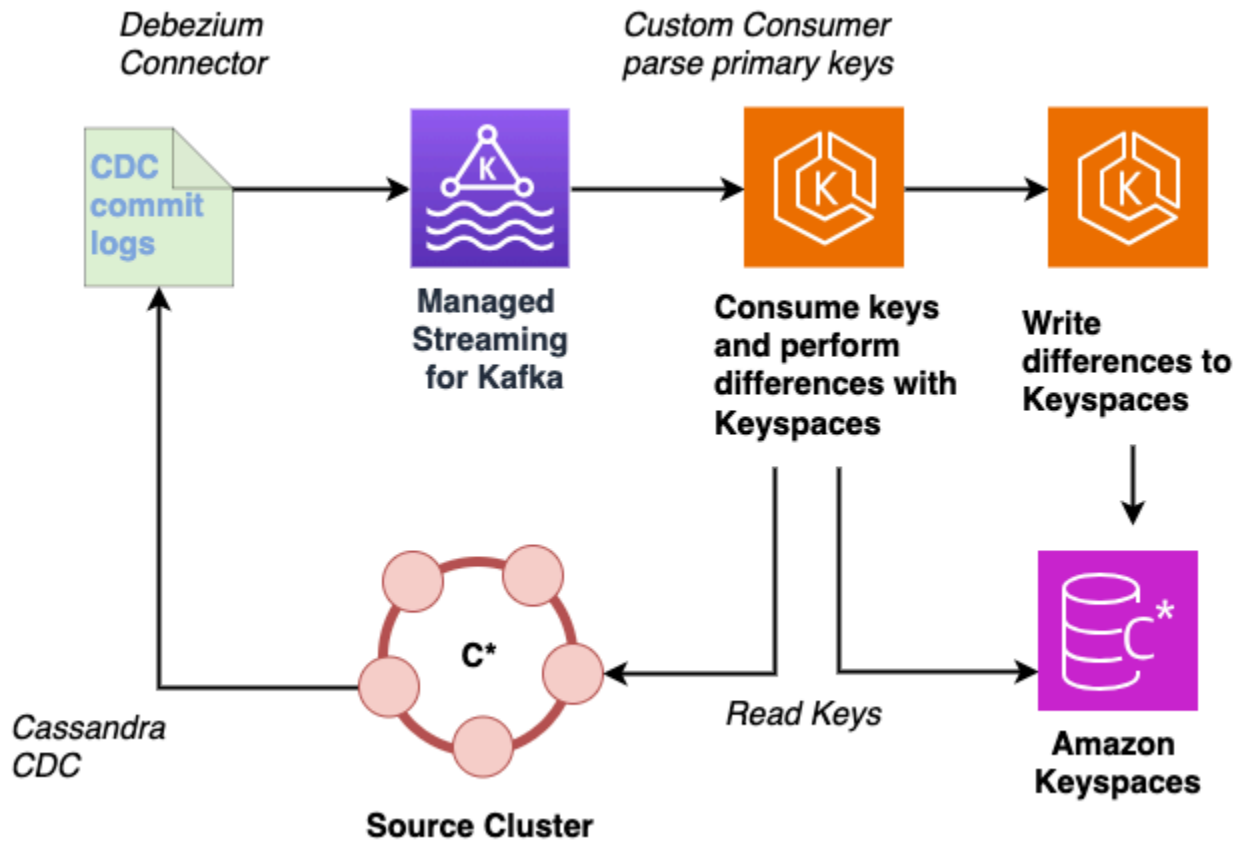
- Como analisar o registro de confirmação do CDC, por exemplo, como remover eventos duplicados.
- Como manter o diretório CDC, por exemplo, como excluir registros antigos.
- Como lidar com falhas parciais no Apache Cassandra, por exemplo, se uma gravação só for bem-sucedida em uma das três réplicas.
- Como lidar com a alocação de recursos, por exemplo, aumentando o tamanho da instância para atender aos requisitos adicionais de CPU, memória, DISCO e E/S para o processo CDC que ocorre em um nó.

Esse padrão trata as mudanças do Cassandra como uma “dica” de que uma chave pode ter mudado em relação ao seu estado anterior. Para determinar se há alterações a serem propagadas para o banco de dados de destino, você deve primeiro ler do cluster Cassandra de origem usando uma operação de LOCAL_QUORUM para receber os registros mais recentes e depois gravá-los no Amazon Keyspaces.

No caso de exclusões ou atualizações de intervalos, talvez seja necessário realizar uma comparação com a partição inteira para determinar quais eventos de gravação ou atualização precisam ser gravados no banco de dados de destino.

Nos casos em que as gravações não são idempotentes, você também precisa comparar suas gravações com o que já está no banco de dados de destino antes de gravar no Amazon Keyspaces.

O diagrama a seguir mostra a arquitetura típica de um pipeline CDC usando o Debezium e o Amazon MSK.



Como selecionar a ferramenta certa para fazer upload ou migrar dados em massa para o Amazon Keyspaces

Nesta seção, você pode analisar as diferentes ferramentas que você pode usar para carregar ou migrar dados em massa para o Amazon Keyspaces e aprender como selecionar a ferramenta correta com base em suas necessidades. Além disso, esta seção fornece uma visão geral e casos de uso dos tutoriais passo a passo disponíveis que demonstram como importar dados para o Amazon Keyspaces.

Para analisar as estratégias disponíveis para migrar workloads do Apache Cassandra para o Amazon Keyspaces, consulte [the section called “Migrar do Cassandra”](#).

- Ferramentas de migração

- Com a [calculadora de preços do Amazon Keyspaces \(para Apache Cassandra\)](#) disponível no Github, você pode estimar seus custos mensais do Amazon Keyspaces com base na sua carga de trabalho atual do Apache Cassandra. Insira métricas da saída de status do nodetool do Cassandra e da configuração sem servidor pretendida para o Amazon Keyspaces para comparar os custos diretos entre as duas soluções. Observe que essa calculadora se concentra apenas nos custos operacionais do Amazon Keyspaces em comparação com sua implantação atual do Cassandra. Não inclui fatores de custo total de propriedade (TCO), como manutenção da infraestrutura, sobrecarga operacional ou custos de suporte para o Cassandra.
- Proxy ZDM Dual Write para migração do Amazon Keyspaces — O ZDM Dual Write Proxy disponível no [Github](#) oferece suporte à migração sem tempo de inatividade do Apache Cassandra para o Amazon Keyspaces.
- CQLReplicator: o CQLReplicator é um utilitário de código aberto disponível no [Github](#) que ajuda você a migrar dados do Apache Cassandra para o Amazon Keyspaces quase em tempo real.

Para obter mais informações, consulte [the section called “CQLReplicator”](#).

- Para saber mais sobre como usar o Amazon Managed Streaming for Apache Kafka para implementar um processo de [migração on-line](#) com gravação dupla, consulte [Guidance for continuous data migration from Apache Cassandra to Amazon Keyspaces](#).
- 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 obter mais informações, consulte [the section called “Migração offline”](#).
- Para saber como usar o conector Apache Cassandra do Spark para gravar dados no Amazon Keyspaces, consulte [the section called “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 obter instruções detalhadas, consulte [the 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 obter instruções detalhadas, consulte [the section called “Carregamento de dados usando o DSbulk”](#).

Considerações gerais sobre uploads de dados para o Amazon Keyspaces

- Divida o upload de dados em componentes menores.

Considere as seguintes unidades de migração e sua presença potencial em termos de tamanho de dados brutos. O upload 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 carregar 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

- [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 tutorial orienta você no processo de migração de dados do Apache Cassandra para o Amazon Keyspaces usando o comando `cqlsh COPY FROM`. O comando `cqlsh COPY FROM` é útil para carregar pequenos conjuntos de dados de forma rápida e fácil no Amazon Keyspaces para fins acadêmicos ou de teste. Para obter mais informações sobre como migrar workloads de produção, consulte [the section called “Migração offline”](#). Você concluirá as seguintes etapas neste tutorial:

Pré-requisitos — Configure uma AWS conta com credenciais, crie um arquivo JKS trust store para o certificado e configure para se conectar `cqlsh` ao Amazon Keyspaces.

1. Crie CSV de origem e tabela de destino: prepare um arquivo CSV como dados de origem e crie o keyspace e a tabela de destino no Amazon Keyspaces.

2. Prepare os dados: randomize os dados no arquivo CSV e analise-os para determinar os tamanhos médio e máximo das linhas.
3. Defina a capacidade de throughput: calcule as unidades de capacidade de gravação (WCUs) necessárias com base no tamanho dos dados e no tempo de carregamento desejado, e configure a capacidade provisionada da tabela.
4. Configurar parâmetros `cqlsh`: determine os valores ideais para parâmetros `decqlsh COPY FROM` como `INGESTRATE`, `NUMPROCESSES`, `MAXBATCHSIZE` e `CHUNKSIZE` distribua a workload uniformemente.
5. Execute o comando **`cqlsh COPY FROM`**: execute o comando `cqlsh COPY FROM` para carregar os dados do arquivo CSV para a tabela do Amazon Keyspaces e monitorar o progresso.

Solução de problemas: resolva problemas comuns, como solicitações inválidas, erros do analisador, erros de capacidade e erros de `cqlsh` durante o processo de upload de dados.

Tópicos

- [Pré-requisitos: etapas a serem concluídas antes de fazer o upload de dados usando `cqlsh COPY FROM`](#)
- [Etapa 1: criar o arquivo CSV de origem e uma tabela de destino para o carregamento de dados](#)
- [Etapa 2: Preparar os dados de origem para um carregamento de dados bem-sucedido](#)
- [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` para carregar dados do arquivo CSV para a tabela de destino](#)
- [Solução de problemas](#)

Pré-requisitos: etapas a serem concluídas antes de fazer o upload de dados usando **`cqlsh COPY FROM`**

É necessário concluir as tarefas a seguir antes de iniciar este tutorial.

1. Se você ainda não fez isso, inscreva-se em uma Conta da AWS seguindo as etapas em [the section called “Configurar AWS Identity and Access Management”](#).
2. Crie credenciais específicas do serviço seguindo as etapas em [the section called “Crie credenciais específicas do serviço”](#).

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 “Usar o cqlsh”](#).

Etapa 1: criar o arquivo CSV de origem e uma tabela de destino para o carregamento de dados

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 T0`, conforme mostrado no exemplo a seguir.

```
cqlsh localhost 9042 -u "username" -p "password" --execute  
"COPY mykeyspace.mytable T0 '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-1.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 de origem para um carregamento de dados bem-sucedido

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 para garantir que o carregamento de dados teve sucesso.

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 obter mais informações, consulte [the section called “Estimar 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 [Como gerenciar 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 saber 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.

Etapas 5: Executar o comando `cqlsh COPY FROM` para carregar dados do arquivo CSV para a tabela de destino

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 compatíveis com o 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 obter 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 obter 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 tutorial passo a passo orienta você na migração de dados do Apache Cassandra para o Amazon Keyspaces DataStax usando o Bulk Loader (DSBulk) disponível em [GitHub](#). O uso do DSBulk é útil para fazer upload de conjuntos de dados no Amazon Keyspaces para fins acadêmicos ou de teste. Para obter mais informações sobre como migrar workloads de produção, consulte [the section called “Migração offline”](#). Neste tutorial, você concluirá as seguintes etapas.

Pré-requisitos — Configure uma AWS conta com credenciais, crie um arquivo de armazenamento confiável JKS para o certificado, configure cqlsh, baixe e instale o DSBulk e configure um arquivo. `application.conf`

1. Crie CSV de origem e tabela de destino: prepare um arquivo CSV como dados de origem e crie o keyspace e a tabela de destino no Amazon Keyspaces.
2. Prepare os dados: randomize os dados no arquivo CSV e analise-os para determinar os tamanhos médio e máximo das linhas.
3. Defina a capacidade de throughput: calcule as unidades de capacidade de gravação (WCUs) necessárias com base no tamanho dos dados e no tempo de carregamento desejado, e configure a capacidade provisionada da tabela.
4. Definir configurações do DSBulk — Crie um arquivo de configuração do DSBulk com configurações como autenticação SSL/TLS, nível de consistência e tamanho do pool de conexão.
5. Execute o comando DSBulk load: execute o comando DSBulk load para carregar os dados do arquivo CSV para a tabela Amazon Keyspaces e monitorar o progresso.

Tópicos

- [Pré-requisitos: Etapas que você precisa concluir antes de fazer o upload de dados com o DSBulk](#)
- [Etapa 1: Criar o arquivo CSV de origem e uma tabela de destino para o upload dos dados usando o DSBulk](#)
- [Etapa 2: Preparar os dados para fazer o upload usando o DSBulk](#)
- [Etapa 3: definir a capacidade de throughput da tabela de destino](#)
- [Etapa 4: definir as configurações de DSBulk para carregar dados do arquivo CSV para a tabela de destino](#)
- [Etapa 5: Execute o comando load de DSBulk para carregar dados do arquivo CSV para a tabela de destino](#)

Pré-requisitos: Etapas que você precisa concluir antes de fazer o upload de dados com o DSBulk

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 em [the section called “Configurar AWS Identity and Access Management”](#).
2. Crie credenciais seguindo as etapas em [the section called “Crie credenciais do IAM para autenticação AWS”](#).
3. Crie um arquivo JKS de armazenamento confiável.
 - a. Faça o download dos seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
```

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

 Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para que confiem tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

- b. Converta os certificados digitais em arquivos TrustStore e adicione-os ao keystore.

```
openssl x509 -outform der -in AmazonRootCA1.pem -out temp_file.der
keytool -import -alias amazon-root-ca-1 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA2.pem -out temp_file.der
keytool -import -alias amazon-root-ca-2 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA3.pem -out temp_file.der
keytool -import -alias amazon-root-ca-3 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA4.pem -out temp_file.der
keytool -import -alias amazon-root-ca-4 -keystore cassandra_truststore.jks -
file temp_file.der

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
```

Na última etapa, você precisa criar uma senha para o armazenamento de chaves e confiar em cada certificado. O comando interativo tem a aparência a seguir.

```
Enter keystore password:
Re-enter new password:
Owner: CN=Amazon Root CA 1, O=Amazon, C=US
Issuer: CN=Amazon Root CA 1, O=Amazon, C=US
Serial number: 66c9fcf99bf8c0a39e2f0788a43e696365bca
```

```

Valid from: Tue May 26 00:00:00 UTC 2015 until: Sun Jan 17 00:00:00 UTC 2038
Certificate fingerprints:
    SHA1: 8D:A7:F9:65:EC:5E:FC:37:91:0F:1C:6E:59:FD:C1:CC:6A:6E:DE:16
    SHA256:
    8E:CD:E6:88:4F:3D:87:B1:12:5B:A3:1A:C3:FC:B1:3D:70:16:DE:7F:57:CC:90:4F:E1:CB:97:C6:AE
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
    CA:true
    PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
    DigitalSignature
    Key_CertSign
    Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 18 CC 85 34 EC BC 0C   94 94 2E 08 59 9C C7 B2   ....4.....Y...
0010: 10 4E 0A 08                               .N..
]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 2, O=Amazon, C=US
Issuer: CN=Amazon Root CA 2, O=Amazon, C=US
Serial number: 66c9fd29635869f0a0fe58678f85b26bb8a37
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
    SHA1: 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A
    SHA256:
    1B:A5:B2:AA:8C:65:40:1A:82:96:01:18:F8:0B:EC:4F:62:30:4D:83:CE:C4:71:3A:19:C3:9C:01:1E
Signature algorithm name: SHA384withRSA

```

Subject Public Key Algorithm: 4096-bit RSA key

Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true

BasicConstraints:[

CA:true

PathLen:2147483647

]

#2: ObjectId: 2.5.29.15 Criticality=true

KeyUsage [

DigitalSignature

Key_CertSign

Crl_Sign

]

#3: ObjectId: 2.5.29.14 Criticality=false

SubjectKeyIdentifier [

KeyIdentifier [

0000: B0 0C F0 4C 30 F4 05 58 02 48 FD 33 E5 52 AF 4B ...L0..X.H.3.R.K

0010: 84 E3 66 52 ..fR

]

]

Trust this certificate? [no]: yes

Certificate was added to keystore

Enter keystore password:

Owner: CN=Amazon Root CA 3, O=Amazon, C=US

Issuer: CN=Amazon Root CA 3, O=Amazon, C=US

Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a

Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040

Certificate fingerprints:

SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E

SHA256:

18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4

Signature algorithm name: SHA256withECDSA

Subject Public Key Algorithm: 256-bit EC (secp256r1) key

Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true

```

BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: AB B6 DB D7 06 9E 37 AC   30 86 07 91 70 C7 9C C4   .....7.0...p...
0010: 19 B1 78 C0               ...x.
]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 4, O=Amazon, C=US
Issuer: CN=Amazon Root CA 4, O=Amazon, C=US
Serial number: 66c9fd7c1bb104c2943e5717b7b2cc81ac10e
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
    SHA1: F6:10:84:07:D6:F8:BB:67:98:0C:C2:E2:44:C2:EB:AE:1C:EF:63:BE
    SHA256:
    E3:5D:28:41:9E:D0:20:25:CF:A6:90:38:CD:62:39:62:45:8D:A5:C6:95:FB:DE:A3:C2:2B:0B:FB:25
Signature algorithm name: SHA384withECDSA
Subject Public Key Algorithm: 384-bit EC (secp384r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true

```

```

KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: D3 EC C7 3A 65 6E CC E1   DA 76 9A 56 FB 9C F3 86   ...:en...v.V....
0010: 6D 57 E5 81                               mW..
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore 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:
    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 (weak)
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               ....
]
]

Warning:
The input uses the SHA1withRSA signature algorithm which is considered a
security risk. This algorithm will be disabled in a future update.

Trust this certificate? [no]: yes
Certificate was added to keystore
```

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 “Usar o cqlsh”](#).
5. Faça download e instale o DSBulk.

Note

A versão mostrada neste tutorial pode não ser a versão mais recente disponível. Antes de baixar o DSBulk, verifique a [página de download do DataStax Bulk Loader](#) para obter a versão mais recente e atualize o número da versão nos comandos a seguir.

- 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 [GitHub](#) 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 uma tabela de destino para o upload dos dados usando o DSBulk

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-1.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 “Verifique o status da criação do keyspace”](#).

```
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 para fazer o upload usando o DSBulk

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 obter mais informações, consulte [the section called “Estimar 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 de destino

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 [Como gerenciar 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 ($360.000 \text{ linhas} / 60 \text{ minutos} / 60 \text{ segundos} = 100 \text{ linhas por segundo}$). 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 saber 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 de **DSBulk** para carregar dados do arquivo CSV para a tabela de destino

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 “Crie credenciais programáticas de acesso”](#).
- 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-1.amazonaws.com`, defina o datacenter local como `us-east-1`. Para todos os disponíveis Regiões da AWS, consulte [the section called “Service endpoints”](#). Para evitar réplicas, defina *slow-replica-avoidance* como `false`.
- c. *SSLConnectionFactory*— Para configurar SSL/TLS, inicie o *SSLConnectionFactory* adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com `class = DefaultSslConnectionFactory`. 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 compatíveis; para obter mais informações, consulte [the section called “Níveis de consistência compatíveis com o 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-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
  }

  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: Execute o comando **load** de DSBulk para carregar dados do arquivo CSV para a tabela de destino

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 obter mais informações, consulte [the section called “Configurar modos read/write de capacidade”](#).

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.

Configurar AWS Identity and Access Management

Inscreva-se para um Conta da AWS

Para começar AWS, você precisa de um Conta da AWS. Para obter informações sobre como criar um Conta da AWS, consulte [Introdução a um Conta da AWS](#) no Guia de AWS Gerenciamento de contas referência.

Configurando o Amazon Keyspaces

O acesso aos recursos do Amazon Keyspaces é gerenciado usando [IAM](#). Usando o IAM, você pode anexar políticas a usuários do IAM, 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 gerenciadas pela AWS para 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, consulte [the section called “AWS políticas gerenciadas”](#).

Para limitar o escopo das ações que uma identidade do IAM pode realizar ou limitar os recursos que a identidade pode acessar, crie uma política personalizada que use a política gerenciada de `AmazonKeyspacesFullAccess` como modelo e remova todas as permissões que você não precisa. Você também pode limitar o acesso a tabelas ou espaços de chaves 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](#)

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 Console de gerenciamento da AWS acesso, consulte Como [controlar o acesso dos usuários do IAM ao Console de gerenciamento da AWS](#) no Guia do usuário do IAM.

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

- Criar, excluir e gerenciar tabelas e keyspaces.
- 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 ou por meio da [AWS CloudShell experiência integrada](#), por exemplo, insira, atualize e exclua dados.
- Conecte-se diretamente ao Amazon Keyspaces por meio do suporte AWS CloudShell integrado ao CQLSH.
- Altere a configuração do particionador da conta.
- Veja as métricas de desempenho e erro da conta no painel.

Conecte-se ao Amazon Keyspaces usando AWS CloudShell do console

AWS CloudShell fornece uma forma simplificada de se conectar ao Amazon Keyspaces diretamente do console. A AWS CloudShell integração inicializa automaticamente o `cqlsh-expansion` para você e estabelece uma conexão com um espaço de teclas específico.

Conceitos básicos de AWS CloudShell integração

1. Acesse a AWS CloudShell integração — No console do Amazon Keyspaces, navegue até o keyspace ao qual você deseja se conectar e escolha o botão **Connect to Keyspaces**. AWS CloudShell
2. Pre-filled comando de configuração — Quando você escolhe o AWS CloudShell botão, o sistema fornece um comando pré-preenchido que inclui os parâmetros necessários de inicialização e conexão:

```
cqlsh-expansion.init ; cqlsh-expansion cassandra.region.amazonaws.com 9142 --ssl  
-my_keyspace
```

3. Clique em **Executar**
4. Aguarde até que o Amazon Keyspaces estabeleça a conexão — Reserve um tempo para que o processo de configuração seja concluído e a conexão com o keyspace selecionado seja estabelecida.
5. Executar comandos — Quando conectado, você pode escrever e executar comandos CQL diretamente no espaço de teclas selecionado.

Comandos CQLSH disponíveis

A AWS CloudShell integração fornece acesso à funcionalidade padrão do CQLSH, incluindo suporte para comandos além do CQL básico.

- **Descrever** — Liste e visualize descrições detalhadas de espaços-chave e tabelas.
- **Copiar** — Exporte a saída da consulta para um arquivo para processamento posterior.
- **Fonte** — Execute instruções CQL a partir de um arquivo.
- **Navegação pelo histórico de comandos** — Percorra os comandos anteriores usando atalhos do teclado.
- **Execução de consultas** — Execute consultas diretamente da interface do teclado.

- Cópia de saída — Copie os resultados como texto simples para uso em outros aplicativos.

Considerações sobre recursos

Ao usar a AWS CloudShell integração, considere as seguintes vantagens e desvantagens.

Os seguintes recursos não são suportados:

- Sugestões de sintaxe CQL e funcionalidade de preenchimento automático.
- Ferramentas de edição, como mostrar preenchimento automático, mostrar dica de ferramenta, localizar, localizar e substituir, refazer, alternar comentários em bloco, alternar dobra de código, alternar comentário em linha e desfazer.

Funcionalidade modificada:

- Downloads de CSV — Para baixar os resultados no formato CSV, especifique o > operador para canalizar a saída do comando:

```
SELECT * FROM my_table > file_name.csv
```

- Visualização JSON - Você pode visualizar itens no formato JSON sem precisar de conversão de dados.
- Terminal-style interface - A experiência funciona como uma interface de terminal em vez de um editor de código completo.

A AWS CloudShell integração simplifica o processo de conexão e, ao mesmo tempo, fornece a funcionalidade essencial do CQL para gerenciar seus dados do Amazon Keyspaces diretamente do Console de gerenciamento da AWS.

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 Console de gerenciamento da AWS. Você pode executar AWS CLI comandos em AWS serviços usando seu shell preferido (Bash PowerShell ou Z shell). AWS CloudShell vem pré-

autenticado com suas credenciais de console e inclui ferramentas comuns para trabalhar com AWS serviços, incluindo AWS CLI Python e. Node.js

Você [inicia a AWS CloudShell partir do Console de gerenciamento da AWS](#), 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).

AWS CloudShell opções de integração

O Amazon Keyspaces oferece duas formas de uso: AWS CloudShell

1. AWS CloudShell Experiência integrada — Um método de conexão simplificado disponível diretamente no console do Amazon Keyspaces que inicializa e se conecta automaticamente `cqlsh-expansion` a um keyspace específico. Para obter instruções completas sobre como usar essa experiência integrada, consulte [the section called “CloudShell integração de console”](#).
2. AWS CloudShell Configuração manual - Instale e configure manualmente `cqlsh` em AWS CloudShell. As etapas estão descritas nas seções a seguir e fornecem mais controle sobre o processo de instalação e permitem configurações personalizadas.

Pré-requisitos

Antes de usar o Amazon Keyspaces com AWS CloudShell, certifique-se de ter:

- Uma AWS conta com acesso ao Amazon Keyspaces e AWS CloudShell
- Permissões apropriadas do IAM para usar os dois serviços
- Familiaridade básica com a Cassandra Query Language (CQL)

Começando com AWS CloudShell

Para acessar AWS CloudShell:

1. Faça login no Console de gerenciamento da AWS
2. Inicie AWS CloudShell escolhendo o AWS CloudShell ícone na barra de navegação na parte superior do console ou navegando até <https://console.aws.amazon.com/cloudshell/>
3. Aguarde até que o AWS CloudShell ambiente seja inicializado

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 o 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, consulte [the section called “AWS políticas gerenciadas”](#).

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”](#).

Instalando o cliente cqlsh

AWS CloudShell não vem com cqlsh pré-instalado. Você precisa instalá-lo para interagir com o Amazon Keyspaces. Você pode instalar a Keyspaces-specific versão Amazon doccqlsh. Para obter instruções de instalação de cqlsh-expansion, consulte [the section called “Como usar o cqlsh-expansion”](#).

Configurando certificados TLS

Para se conectar ao Amazon Keyspaces usando uma conexão TLS, você precisa baixar os certificados digitais da Amazon necessários. Para obter instruções detalhadas sobre como baixar e configurar o certificado SSL, consulte [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#)

Conectando-se ao Amazon Keyspaces

Você pode se conectar ao Amazon Keyspaces usando credenciais do IAM (recomendado) ou credenciais específicas do serviço.

Conexão com as credenciais do IAM (recomendado)

AWS CloudShell já está autenticado com suas credenciais do IAM. Para se conectar usando essas credenciais, você pode usar o plug-in de autenticação SigV4. Para obter instruções detalhadas sobre como usar as credenciais do IAM com cqlsh, consulte [the section called “Como usar o cqlsh-expansion”](#).

Como conectar-se com credenciais específicas do serviço

Se você criou credenciais específicas de serviço para o Amazon Keyspaces, você pode usá-las para se conectar. Para obter informações sobre como criar e usar credenciais específicas do serviço, consulte [the section called “Crie credenciais específicas do serviço”](#)

Usando o CQLSH com o Amazon Keyspaces

Quando conectado, você pode usar comandos CQL padrão para interagir com seus espaços de teclas e tabelas. Para obter informações detalhadas sobre comandos e operações CQL compatíveis, consulte [the section called “Como usar o cqlsh-expansion”](#).

Dados persistentes em AWS CloudShell

AWS CloudShell fornece 1 GB de armazenamento persistente em seu diretório inicial. Isso significa que os scripts, certificados e arquivos de configuração que você cria permanecem disponíveis em todas as sessões. Você pode criar um arquivo de configuração para `cqlsh` simplificar as conexões.

Com um arquivo de configuração, você pode executar `cqlsh` sem especificar parâmetros adicionais. Para obter mais informações sobre como criar e usar arquivos de `cqlsh` configuração, consulte [the section called “Como usar o cqlsh-expansion”](#).

Práticas recomendadas

Ao usar o Amazon Keyspaces com AWS CloudShell, recomendamos as seguintes melhores práticas:

- Crie scripts para tarefas repetitivas e armazene-os em seu diretório inicial persistente.
- Use variáveis de ambiente para informações confidenciais em vez de codificá-las.
- Lembre-se de que AWS CloudShell as sessões podem expirar após a inatividade.
- Esteja atento às cotas e limitações do AWS CloudShell serviço.
- Considere usar o AWS CLI para operações automatizadas com o Amazon Keyspaces.

Solução de problemas

Se você encontrar problemas ao se conectar ao Amazon Keyspaces a partir de: AWS CloudShell

- Verifique se suas permissões do IAM incluem as ações necessárias do Amazon Keyspaces.
- Verifique se você está usando o endpoint correto para sua região.
- Verifique se os certificados SSL foram baixados e referenciados corretamente.
- Verifique se suas credenciais específicas do serviço estão corretas e não expiraram.
- Se estiver usando um arquivo `cqlshrc`, verifique sua formatação e permissões.

Exemplo: Como interagir com o Amazon Keyspaces usando AWS CloudShell

Depois AWS CloudShell de iniciar a partir do Console de gerenciamento da AWS, você pode começar imediatamente a interagir com o Amazon Keyspaces usando nossa interface `cqlsh` de linha de comando. Se você ainda não instalou o `cqlsh-expansion`, consulte [the section called “Como usar o cqlsh-expansion”](#) para 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 espaço de chaves. Em seguida, leia uma tabela do sistema para confirmar se o keyspace foi criado usando AWS CloudShell

1. A partir do Console de gerenciamento da AWS, 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 CloudShell opção.
2. Você pode estabelecer uma conexão com o Amazon Keyspaces usando o comando a seguir. Certifique-se de `cassandra.us-east-1.amazonaws.com` substituir 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 chave com o nome `mykeyspace`. É possível usar o seguinte comando para fazer isso:

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

4. Para confirmar que o espaço de chave 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)
```

Crie credenciais para acesso programático ao Amazon Keyspaces

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

- [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)
- [Crie e configure AWS credenciais para o Amazon Keyspaces](#)

Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces

Service-specific as credenciais são semelhantes ao nome de usuário e senha tradicionais que Cassandra usa para autenticação e gerenciamento de acesso. Service-specific as credenciais permitem que os usuários do IAM acessem um AWS serviço específico. Essas credenciais de longo prazo não podem ser usadas para acessar outros AWS serviços. Elas estão associados a um usuário específico do IAM e não podem ser usadas por outros usuários do IAM.

Important

Service-specific as credenciais 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 uma credencial específica do serviço.

Console

Crie credenciais específicas do serviço usando o console

1. Faça login no Console de gerenciamento da AWS e abra o AWS Identity and Access Management console em <https://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.

CLI

Crie 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 (CLI): AWS CLI

1. Baixe o 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 usuária `alice`, 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.

Crie e configure 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 do IAM com chaves de acesso. Ao usar o Amazon Keyspaces programaticamente, você fornece suas chaves de AWS acesso para que você 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, /b CYEXAMPLEKEY). wJalrXUtnFEMI/K7MDENG PxRfi 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 perfis do IAM às permissões necessárias para realizar tarefas

específicas. Os usuários do IAM podem então assumir temporariamente os perfis do IAM para realizar as tarefas necessárias. Por exemplo, os usuários do IAM na sua conta que estiverem usando o console do Amazon Keyspaces podem mudar para uma função para usarem, 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. Ao contrário, os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança se, em vez de assumir funções, tiverem permissões atribuídas diretamente 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 [Common Scenarios for Roles: Users, Applications, and Services](#) no Guia do usuário do IAM.

Tópicos

- [Credenciais exigidas pelo AWS CLI AWS SDK ou pelo plug-in SigV4 do Amazon Keyspaces para drivers de clientes do Cassandra](#)
- [Crie credenciais temporárias para se conectar ao Amazon Keyspaces usando um perfil do IAM e o plug-in SigV4](#)
- [Crie um usuário do IAM para acesso programático ao Amazon Keyspaces em sua conta AWS](#)
- [Crie novas chaves de acesso para um usuário do IAM](#)
- [Armazene chaves de acesso para acesso programático](#)

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

As seguintes credenciais são necessárias para autenticar o usuário do IAM 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 obter mais informações, consulte [the section called “Crie 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.

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

A forma recomendada de acessar o Amazon Keyspaces programaticamente é usando [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 no Console de gerenciamento da AWS 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.

Crie 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 Console de gerenciamento da AWS Tools for Windows PowerShell ou usando uma operação de AWS API. AWS CLI Se você criar o usuário no Console de gerenciamento da AWS, 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 do IAM, consulte [Permissões necessárias para acessar recursos do IAM](#).

Console

Crie um usuário do IAM com acesso programático (console)

1. Faça login no Console de gerenciamento da AWS 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 em uma conta. Eles não são diferenciados por 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 “AWS políticas gerenciadas”](#).

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 deixe os usuários assumirem temporariamente as funções apropriadas.

Escolha Próximo: tags.

6. Na página Adicionar tags (opcional), você pode adicionar tags para o usuário ou escolher Avançar: Revisão.
7. Na página Revisão, você pode ver todas as escolhas feitas até esse ponto. Quando estiver pronto para continuar, escolha 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.

CLI

Crie 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](#)

Crie novas 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 atualizar 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 no Console de gerenciamento da AWS 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 Chaves de acesso, em Melhores práticas e alternativas das chaves de acesso, escolha o caso de uso Outro. Clique em Avançar, insira as informações opcionais conforme necessário e escolha 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 /b PxrFi CYEXAMPLEKEY

 Note

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

Considere as seguintes práticas recomendadas para o key pair que você criou.

- Nunca armazene sua chave de acesso em texto simples, em um repositório de código ou em código.
 - Desative ou exclua as chaves de acesso quando não forem mais necessárias.
 - Ative as permissões com privilégios mínimos.
 - Gire as teclas de acesso regularmente.
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.

Armazene chaves de acesso para acesso programático

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”](#). Observe que isso inclui o padrão Região da AWS. 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/bPxrFcjYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of security token>
$ export AWS_DEFAULT_REGION=us-east-1
```

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>
C:\> setx AWS_DEFAULT_REGION us-east-1
```

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>"
PS C:\> $Env:AWS_DEFAULT_REGION="us-east-1"
```

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.

Endpoints de serviço para Amazon Keyspaces

Tópicos

- [Portas e protocolos](#)
- [Endpoints globais](#)

- [AWS GovCloud \(US\) Region Endpoints do FIPS](#)
- [Endpoints da região da China](#)
- [Endpoints de streaming](#)
- [Conexão a endpoints de pilha dupla](#)

Portas e protocolos

Você pode acessar o Amazon Keyspaces programaticamente executando um `cqlsh` cliente, com um driver Cassandra licenciado pelo Apache 2.0 ou usando o e o SDK. AWS CLI 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 certificados emitidos sob o Amazon Trust Services (Amazon Root Cas 1—4) 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 oferece suporte a endpoints públicos IPv4 e IPv6. Você pode escolher entre endpoints IPv4 e endpoints de pilha dupla. Os endpoints usam a seguinte convenção de nomenclatura, na qual você pode `us-east-1` substituir por outra disponível na Região da AWS tabela.

- Endpoints IPv4 — `cassandra.us-east-1.amazonaws.com`
- Dual-stack pontos finais — `cassandra.us-east-1.api.aws`

Para obter mais informações sobre endpoints de pilha dupla e como configurar conexões, consulte [the section called “Conexão a endpoints de pilha dupla”](#)

O Amazon Keyspaces está disponível nas seguintes regiões.

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Ohio)	us-east-2	cassandra.us-east-2.amazonaws.com	HTTPS e TLS	
		cassandra.us-east-2.api.aws	HTTPS e TLS	
Leste dos EUA (Norte da Virgínia)	us-east-1	cassandra.us-east-1.amazonaws.com	HTTPS e TLS	
		cassandra-fips.us-east-1.api.aws	HTTPS e TLS	
		cassandra-fips.us-east-1.amazonaws.com	HTTPS e TLS	
		cassandra.us-east-1.api.aws	HTTPS e TLS	
Oeste dos EUA (Norte da Califórnia)	us-west-1	cassandra.us-west-1.amazonaws.com	HTTPS e TLS	
		cassandra.us-west-1.api.aws	HTTPS e TLS	
Oeste dos EUA (Oregon)	us-west-2	cassandra.us-west-2.amazonaws.com	HTTPS e TLS	
		cassandra-fips.us-west-2.api.aws	HTTPS e TLS	
		cassandra-fips.us-west-2.amazonaws.com	HTTPS e TLS	
		cassandra.us-west-2.api.aws	HTTPS e TLS	

Nome da região	Região	Endpoint	Protocolo	
			HTTPS e TLS	
África (Cidade do Cabo)	af-south-1	cassandra.af-south-1.amazonaws.com cassandra.af-south-1.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Hong Kong)	ap-east-1	cassandra.ap-east-1.amazonaws.com cassandra.ap-east-1.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Malásia)	ap-southeast-5	cassandra.ap-southeast-5.amazonaws.com cassandra.ap-southeast-5.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com cassandra.ap-south-1.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Seul)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com cassandra.ap-northeast-2.api.aws	HTTPS e TLS HTTPS e TLS	

Nome da região	Região	Endpoint	Protocolo	
Ásia-Pacífico (Singapura)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com cassandra.ap-southeast-1.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Sydney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com cassandra.ap-southeast-2.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Tailândia)	ap-southeast-7	cassandra.ap-southeast-7.amazonaws.com cassandra.ap-southeast-7.api.aws	HTTPS e TLS HTTPS e TLS	
Ásia-Pacífico (Tóquio)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com cassandra.ap-northeast-1.api.aws	HTTPS e TLS HTTPS e TLS	
Canadá (Central)	ca-central-1	cassandra.ca-central-1.amazonaws.com cassandra.ca-central-1.api.aws	HTTPS e TLS HTTPS e TLS	
Europa (Frankfurt)	eu-central-1	cassandra.eu-central-1.amazonaws.com cassandra.eu-central-1.api.aws	HTTPS e TLS HTTPS e TLS	

Nome da região	Região	Endpoint	Protocolo	
Europa (Irlanda)	eu-west-1	cassandra.eu-west-1.amazonaws.com cassandra.eu-west-1.api.aws	HTTPS e TLS HTTPS e TLS	
Europa (Londres)	eu-west-2	cassandra.eu-west-2.amazonaws.com cassandra.eu-west-2.api.aws	HTTPS e TLS HTTPS e TLS	
Europa (Paris)	eu-west-3	cassandra.eu-west-3.amazonaws.com cassandra.eu-west-3.api.aws	HTTPS e TLS HTTPS e TLS	
Europa (Estocolmo)	eu-north-1	cassandra.eu-north-1.amazonaws.com cassandra.eu-north-1.api.aws	HTTPS e TLS HTTPS e TLS	
Oriente Médio (Barém)	me-south-1	cassandra.me-south-1.amazonaws.com cassandra.me-south-1.api.aws	HTTPS e TLS HTTPS e TLS	
Oriente Médio (Emirados Árabes Unidos)	me-central-1	cassandra.me-central-1.amazonaws.com cassandra.me-central-1.api.aws	HTTPS e TLS HTTPS e TLS	

Nome da região	Região	Endpoint	Protocolo	
América do Sul (São Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com cassandra.sa-east-1.api.aws	HTTPS e TLS HTTPS e TLS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com cassandra.us-gov-east-1.api.aws	HTTPS e TLS HTTPS e TLS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com cassandra.us-gov-west-1.api.aws	HTTPS e TLS HTTPS e TLS	

AWS GovCloud (US) Region Endpoints do FIPS

Endpoints FIPS disponíveis no AWS GovCloud (US) Region. O Amazon Keyspaces oferece suporte a endpoints FIPS IPv4 e IPv6. Você pode escolher entre endpoints IPv4 e endpoints de pilha dupla. Para obter mais informações, consulte [Amazon Keyspaces no AWS GovCloud \(US\) Guia do Usuário](#)

Nome da região	Região	Ponto final FIPS IPv4	Pilha dupla de terminais FIPS	Protocolo
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	cassandra.us-gov-east-1.api.aws	HTTPS e TLS
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	cassandra.us-gov-west-1.api.aws	HTTPS e TLS

Nome da região	Região	Ponto final FIPS IPv4	Pilha dupla de terminais FIPS	Protocolo
(US-West)	west-1			

Endpoints da região da China

O Amazon Keyspaces oferece suporte a endpoints IPv4 nas regiões da China. AWS

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

Endpoints de streaming

Amazon Keyspaces CDC streams está disponível a seguir Regiões da AWS. Esta tabela mostra o endpoint de serviço de pilha dupla disponível para cada região. Para obter mais informações sobre Amazon Keyspaces CDC streams, consulte [the section called “Acesse endpoints de fluxo CDC”](#).

Nome da região	Região	Endpoint	Protocolo
Leste dos EUA (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Norte da Virgínia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
Oeste dos EUA (Norte da Califórnia)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
Oeste dos EUA (Oregon)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
África (Cidade do Cabo)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Ásia-Pacífico (Hong Kong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Ásia-Pacífico (Malásia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Ásia-Pacífico (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Ásia-Pacífico (Seul)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Ásia-Pacífico (Singapura)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	
Ásia-Pacífico (Sydney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS	
Ásia-Pacífico (Tailândia)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS	
Ásia-Pacífico (Tóquio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS	
Canadá (Central)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS	
Europa (Frankfurt)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS	
Europa (Irlanda)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS	
Europa (Londres)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS	
Europa (Paris)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Europa (Estocolmo)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Oriente Médio (Barém)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Oriente Médio (Emirados Árabes Unidos)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
América do Sul (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Conexão a endpoints de pilha dupla

Os endpoints globais do Amazon Keyspaces são endpoints de pilha dupla que aceitam solicitações IPv4 e IPv6.

Ao se conectar ao Amazon Keyspaces usando IPv6, o serviço adapta automaticamente as respostas da tabela do sistema para corresponder ao seu protocolo de conexão. Isso garante que seus aplicativos recebam informações de endereço de rede consistentes que correspondam ao tipo de conexão. Isso fornece informações precisas da topologia de rede ao cliente, mantendo a compatibilidade com versões anteriores dos aplicativos CQL existentes.

O Amazon Keyspaces detecta automaticamente o protocolo de rede (IPv4 ou IPv6) usado pela conexão do seu cliente e ajusta adequadamente as respostas da tabela do sistema. Essa detecção ocorre de forma transparente durante o handshake de conexão inicial, não exigindo nenhuma configuração adicional do aplicativo cliente.

O Amazon Keyspaces retorna endereços IP com base no seu protocolo de conexão. Por exemplo, uma solicitação de uma rede IPv4 retorna a seguinte resposta.

```
SELECT * FROM system.peers;  
-- Returns IPv4 addresses in peer column  
-- Example: 172.31.1.1, 172.31.1.2, etc.
```

Uma conexão de uma rede IPv6 com um endpoint de pilha dupla, por exemplo `cassandra.us-east-1.api.aws`, retorna a seguinte resposta.

```
SELECT * FROM system.peers;  
-- Returns IPv6 addresses in peer column  
-- Example: 2001:db8::1, 2001:db8::2, etc.
```

Para obter mais informações sobre o suporte ao IPv6 no Amazon Keyspaces, consulte [the section called “Suporte a IPv6 no Amazon Keyspaces”](#)

Suporte a IPv6 no Amazon Keyspaces

O suporte ao IPv6 no Amazon Keyspaces permite que os aplicativos estabeleçam conexões usando o Internet Protocol versão 6, o protocolo de internet de próxima geração que fornece um espaço de endereço amplamente expandido em comparação ao IPv4. A implementação usa endpoints de pilha dupla que suportam IPv4 e IPv6 simultaneamente, garantindo compatibilidade com versões anteriores e permitindo conectividade pronta para o futuro. Para uma lista de endpoints, consulte [the section called “Endpoints globais”](#).

O Amazon Keyspaces implementa o suporte ao IPv6 por meio de uma arquitetura de pilha dupla que mantém total compatibilidade com versões anteriores e permite a conectividade IPv6.

Resolução de DNS no Amazon Keyspaces

Quando os aplicativos se conectam a endpoints de pilha dupla, o processo de resolução de DNS retorna os dois tipos de endereço:

Registros A (IPv4)

Endereços IPv4 tradicionais para compatibilidade com versões anteriores

Registros AAAA (IPv6)

Novos endereços IPv6 para conectividade moderna

O sistema operacional e a pilha de rede do cliente selecionam automaticamente o protocolo mais adequado com base na configuração local, na disponibilidade da rede e nas preferências do sistema.

O protocolo Cassandra Query Language (CQL) suporta perfeitamente a conectividade IPv6 sem exigir alterações no código do aplicativo.

Seleção automática de protocolos

- Os aplicativos especificam o endpoint de pilha dupla
- A pilha de rede escolhe IPv4 ou IPv6 com base na disponibilidade
- Nenhuma modificação de código é necessária para aplicativos CQL existentes

Compatibilidade de drivers

- Todos os principais drivers CQL suportam IPv6 de forma transparente
- DataStax os drivers lidam com endereços IPv6 de forma nativa
- Open-source os motoristas trabalham sem modificação

Consistência da conexão

- As tabelas do sistema refletem o protocolo de conexão usado
- As conexões IPv6 mostram endereços IPv6 em `system.peers`
- As conexões IPv4 continuam mostrando endereços IPv4

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.

Note

`Murmur3Partitioner` é o particionador recomendado para o Amazon Keyspaces e o `cqlsh-expansion`. O `cqlsh-expansion` não é compatível com o Amazon Keyspaces `DefaultPartitioner`. Para obter mais informações, consulte [the section called “Como trabalhar com particionadores”](#).

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, é possível executar o comando a seguir.

```
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 no PATH, 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 foi 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 arquivo de certificado combinado no `.cassandra` diretório. 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 obter mais informações sobre certificados, consulte [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

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 sua 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 `us-east-1`.

```
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”](#). 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 “Crie 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 exibir 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 “Crie 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 `cqlsh-expansion` utilitário que baixa automaticamente os certificados para você e instala um arquivo de `cqlshrc` configuração pré-configurado. Para obter mais informações sobre isso, consulte [the section called “Como usar o cqlsh-expansion”](#) nesta página.

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

1. Baixe os seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.
 - a. AmazonRootCA1
 - b. AmazonRootCA2
 - c. AmazonRootCA3
 - d. AmazonRootCA4
 - e. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para que confiem tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

2. Combine todos os certificados baixados em um único pem arquivo com o nome **keyspaces-bundle.pem** em nossos exemplos. Você pode fazer isso executando o seguinte comando : Anote o caminho para o arquivo, você precisará disso mais tarde.

```
cat AmazonRootCA1.pem \  
    AmazonRootCA2.pem \  
    AmazonRootCA3.pem \  
    AmazonRootCA4.pem \  
    sf-class2-root.crt \  
> keyspaces-bundle.pem
```

3. 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/keyspaces-bundle.pem
```

Usando o AWS CLI para se conectar ao Amazon Keyspaces

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

Antes de usar o 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 “Crie credenciais do IAM para autenticação AWS”](#).

Para obter uma lista completa de todos os comandos disponíveis para o Amazon Keyspaces no AWS CLI, consulte a Referência de [AWS CLI comandos](#).

Tópicos

- [Baixando e configurando o AWS CLI](#)
- [Usando o AWS CLI com o Amazon Keyspaces](#)

Baixando e configurando o AWS CLI

O AWS CLI está disponível em <https://aws.amazon.com/cli>. Ela é executada no Windows, macOS ou Linux. Depois de baixar o AWS CLI, siga estas etapas para instalá-lo e configurá-lo:

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](#)

Usando o 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 dos parâmetros, bem como para JSON. Os seguintes exemplos do Amazon Keyspaces usam sintaxe AWS CLI abreviada. Para obter mais informações, consulte [Usando a sintaxe abreviada com a CLI](#). AWS

O comando a seguir cria um espaço de chaves 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-1:111122223333:/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-1:111122223333:/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-1:111122223333:/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-1:111122223333:/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 Point-in-time a recuperação da 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 AWS KMS chave 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-1:111122223333:key/11111111-2222-3333-4444-555555555555'
                        --ttl 'status=ENABLED'
```

Usar a API para se conectar ao Amazon Keyspaces

Você pode usar o AWS SDK e o AWS Command Line Interface (AWS CLI) para trabalhar interativamente 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 o Terraform.

Antes de usar o 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 “Crie 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](#).

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 ajudá-lo a começar, você pode visualizar e baixar exemplos de código de ponta a ponta 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 obter um tutorial detalhado que mostra como se conectar ao Amazon Keyspaces a partir de um endpoint da VPC da Amazon, consulte [the section called “Conexão com endpoints da VPC”](#).

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](#)
- [Usando um driver de Node.js cliente Cassandra 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 obter 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 obter mais informações, consulte [the section called “Crie 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](#)
- [Step-by-step tutorial para se conectar ao Amazon Keyspaces usando o driver DataStax Java para Apache Cassandra usando credenciais específicas do serviço](#)
- [Step-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 dos seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
```

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para confiar tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

- b. Converta os certificados digitais em arquivos TrustStore e adicione-os ao keystore.

```
openssl x509 -outform der -in AmazonRootCA1.pem -out temp_file.der
keytool -import -alias amazon-root-ca-1 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA2.pem -out temp_file.der
keytool -import -alias amazon-root-ca-2 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA3.pem -out temp_file.der
keytool -import -alias amazon-root-ca-3 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA4.pem -out temp_file.der
keytool -import -alias amazon-root-ca-4 -keystore cassandra_truststore.jks -
file temp_file.der

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
```

Na última etapa, você precisa criar uma senha para o armazenamento de chaves e confiar em cada certificado. O comando interativo tem a aparência a seguir.

```
Enter keystore password:
Re-enter new password:
Owner: CN=Amazon Root CA 1, O=Amazon, C=US
```

```

Issuer: CN=Amazon Root CA 1, O=Amazon, C=US
Serial number: 66c9fcf99bf8c0a39e2f0788a43e696365bca
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sun Jan 17 00:00:00 UTC 2038
Certificate fingerprints:
    SHA1: 8D:A7:F9:65:EC:5E:FC:37:91:0F:1C:6E:59:FD:C1:CC:6A:6E:DE:16
    SHA256:
    8E:CD:E6:88:4F:3D:87:B1:12:5B:A3:1A:C3:FC:B1:3D:70:16:DE:7F:57:CC:90:4F:E1:CB:97:C6:AE
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
    CA:true
    PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
    DigitalSignature
    Key_CertSign
    Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 18 CC 85 34 EC BC 0C   94 94 2E 08 59 9C C7 B2   ....4.....Y...
0010: 10 4E 0A 08                                     .N..
]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 2, O=Amazon, C=US
Issuer: CN=Amazon Root CA 2, O=Amazon, C=US
Serial number: 66c9fd29635869f0a0fe58678f85b26bb8a37
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
    SHA1: 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A

```

```

    SHA256:
    1B:A5:B2:AA:8C:65:40:1A:82:96:01:18:F8:0B:EC:4F:62:30:4D:83:CE:C4:71:3A:19:C3:9C:01:1E
Signature algorithm name: SHA384withRSA
Subject Public Key Algorithm: 4096-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
    CA:true
    PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
    DigitalSignature
    Key_CertSign
    Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
    KeyIdentifier [
0000: B0 0C F0 4C 30 F4 05 58    02 48 FD 33 E5 52 AF 4B    ...L0..X.H.3.R.K
0010: 84 E3 66 52                                ..fR
    ]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 3, O=Amazon, C=US
Issuer: CN=Amazon Root CA 3, O=Amazon, C=US
Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
    SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E
    SHA256:
    18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4
Signature algorithm name: SHA256withECDSA
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
Version: 3

```

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true

BasicConstraints:[

CA:true

PathLen:2147483647

]

#2: ObjectId: 2.5.29.15 Criticality=true

KeyUsage [

DigitalSignature

Key_CertSign

Crl_Sign

]

#3: ObjectId: 2.5.29.14 Criticality=false

SubjectKeyIdentifier [

KeyIdentifier [

0000: AB B6 DB D7 06 9E 37 AC 30 86 07 91 70 C7 9C C4 7.0...p...

0010: 19 B1 78 C0 ..x.

]

]

Trust this certificate? [no]: yes

Certificate was added to keystore

Enter keystore password:

Owner: CN=Amazon Root CA 4, O=Amazon, C=US

Issuer: CN=Amazon Root CA 4, O=Amazon, C=US

Serial number: 66c9fd7c1bb104c2943e5717b7b2cc81ac10e

Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040

Certificate fingerprints:

SHA1: F6:10:84:07:D6:F8:BB:67:98:0C:C2:E2:44:C2:EB:AE:1C:EF:63:BE

SHA256:

E3:5D:28:41:9E:D0:20:25:CF:A6:90:38:CD:62:39:62:45:8D:A5:C6:95:FB:DE:A3:C2:2B:0B:FB:25

Signature algorithm name: SHA384withECDSA

Subject Public Key Algorithm: 384-bit EC (secp384r1) key

Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true

BasicConstraints:[

CA:true

PathLen:2147483647


```

]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
  KeyIdentifier [
    0000: D3 EC C7 3A 65 6E CC E1   DA 76 9A 56 FB 9C F3 86   ...:en...v.V....
    0010: 6D 57 E5 81                               mW..
  ]
]

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore 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:
    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 (weak)
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               ....
  ]
]

Warning:
The input uses the SHA1withRSA signature algorithm which is considered a
security risk. This algorithm will be disabled in a future update.

Trust this certificate? [no]: yes
Certificate was added to keystore

```

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

```

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

O tutorial passo a passo a seguir o ajuda a se conectar ao Amazon Keyspaces através do driver Java para Apache 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 “Crie 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”](#).

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. `ServiceUserName` e `ServicePassword` deve corresponder ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas em [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)

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 `SSLConnectionFactory` adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com `class = DefaultSslConnectionFactory`. 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 = DefaultSslConnectionFactory  
        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.

Step-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 [GitHub repositório](#).

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 obter mais informações, consulte [the section called “Crie 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 “Crie credenciais do IAM para autenticação AWS”](#). Este tutorial pressupõe que as chaves de acesso sejam armazenadas como variáveis de ambiente. Para obter mais informações, consulte [the section called “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”](#).

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`. `SigV4AuthProvider` 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 [???](#). Defina `slow-replica-avoidance = false` para balancear a carga em relação a todos os nós disponíveis.
3. Idempotência — Defina o padrão `idempotence` para que o aplicativo configure o driver `true` para sempre repetir solicitações com falha. `read/write prepare/execute` 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 `SSLConnectionFactory` adicionando uma seção no arquivo de configuração com uma única linha que especifica a classe com `class = DefaultSslConnectionFactory`. 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 configurando `local.size = 3`. 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, consulte [the section called “Como configurar conexões”](#).
6. Política de repetição: implemente a política de repetição do Amazon Keyspaces `AmazonKeyspacesExponentialRetryPolicy` em vez da `DefaultRetryPolicy` que vem com o driver Cassandra. Isso permite que você configure o número de tentativas de repetição para `AmazonKeyspacesExponentialRetryPolicy` que atenda às suas necessidades. Por padrão, o número de tentativas de repetição do `AmazonKeyspacesExponentialRetryPolicy` é definido como 3. Para obter mais informações, consulte [the section called “Como configurar políticas de repetição”](#).
7. Declarações preparadas: defina `prepare-on-all-nodes` como falso 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 obter mais informações, consulte [the section called “Crie 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 “Crie credenciais do IAM para autenticação AWS”](#). Este tutorial pressupõe que as chaves de acesso sejam armazenadas como variáveis de ambiente. Para obter mais informações, consulte [the section called “Gerenciar chaves de acesso”](#).
- Siga as etapas em [the section called “Antes de começar”](#) para baixar os certificados digitais, convertê-los em arquivos TrustStore e anexar o keystore 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”](#).

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 obter mais informações, consulte [the section called “Crie 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 os certificados digitais da Amazon e configurar o driver do Python para usar o TLS.

Faça o download dos seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
```

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para confiar tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

Combine todos os certificados baixados em um único pem arquivo com o nome *keyspaces-bundle.pem* em nossos exemplos. Você pode fazer isso executando o seguinte comando : Anote o caminho para o arquivo, você precisará disso mais tarde.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
    AmazonRootCA4.pem \
    sf-class2-root.crt \
    > keyspaces-bundle.pem
```

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/keyspaces-bundle.pem')

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. "*path_to_file/keyspaces-bundle.pem*" Substitua pelo caminho para o arquivo de certificado combinado salvo na primeira etapa.
2. Certifique-se de que *ServiceUserName* e *ServicePassword* corresponda ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)
3. Para obter uma lista de endpoints disponíveis, consulte [the section called “Service endpoints”](#).

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 “Crie 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 “Crie 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/keyspaces-bundle.pem')
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/keyspaces-bundle.pem*" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que o *aws_access_key_id*, *aws_secret_access_key*, e o *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 para Python (Boto3).
3. Para obter uma lista de endpoints disponíveis, consulte [the section called "Service endpoints"](#).

Usando um driver de Node.js cliente Cassandra para acessar o Amazon Keyspaces programaticamente

Esta seção mostra como se conectar ao Amazon Keyspaces usando um driver de Node.js cliente. 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 obter mais informações, consulte [the section called “Crie credenciais do IAM para autenticação AWS”](#).

Tópicos

- [Antes de começar](#)
- [Conecte-se ao Amazon Keyspaces usando o Node.js DataStax driver do Apache Cassandra e credenciais específicas do serviço](#)
- [Conecte-se ao Amazon Keyspaces usando o DataStax Node.js driver do 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.

Baixe os seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3

4. AmazonRootCA4

5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para confiar tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

Combine todos os certificados baixados em um único pem arquivo com o nome *keyspaces-bundle.pem* em nossos exemplos. Você pode fazer isso executando o seguinte comando : Anote o caminho para o arquivo, você precisará disso mais tarde.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
    AmazonRootCA4.pem \
    sf-class2-root.crt \
    > keyspaces-bundle.pem
```

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

Configure seu driver para usar o arquivo de certificado combinado *keyspaces-bundle.pem* 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/keyspaces-bundle.pem', '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. "*path_to_file/keyspaces-bundle.pem*" Substitua pelo caminho para o arquivo de certificado combinado salvo na primeira etapa.
2. Certifique-se de que *ServiceUserName* e *ServicePassword* corresponda ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)
3. Para obter uma lista de endpoints disponíveis, consulte [the section called "Service endpoints"](#).

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

A seção a seguir mostra como usar o plug-in de autenticação SigV4 para o DataStax Node.js driver 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 “Crie credenciais do IAM para autenticação AWS”](#).

[Adicione o plug-in de autenticação Node.js SigV4 ao seu aplicativo a partir do GitHub repositório](#). O plug-in é compatível com a versão 4.x do DataStax Node.js driver do 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 Region-specific instância de `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_file/keyspaces-bundle.pem', '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/keyspaces-bundle.pem*" pelo caminho para o certificado salvo na primeira etapa.
2. Certifique-se de que *accessKeyId* e *secretAccessKey* corresponda à chave de acesso e à chave de acesso secreta que você obteve usando `AWSCredentialsProvider`. Para obter mais informações, consulte [Como configurar credenciais Node.js no AWS SDK para JavaScript](#) entrar. Node.js
3. Para armazenar chaves de acesso fora do código, consulte as melhores práticas em [the section called “Gerenciar chaves de acesso”](#).
4. Para obter uma lista de endpoints disponíveis, consulte [the section called “Service endpoints”](#).

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 TLS, configure seu driver para usar o armazenamento confiável do sistema, que inclui as CAs 1-4 do Amazon Root.

1. Instale o `CassandraCSharpDriver` through nuget, usando o console do nuget.

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

2. O exemplo a seguir usa um projeto de console do .NET Core C# 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)
        {
            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:

1. Certifique-se de usar o armazenamento confiável padrão do sistema, que inclui as CAs 1-4 do Amazon Root.

2. Certifique-se de que *ServiceUserName* e *ServicePassword* corresponda ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)
3. Para obter uma lista de endpoints disponíveis, consulte [the section called “Service endpoints”](#).

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 Cassandra. 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 entidades principais do IAM que sejam 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 obter mais informações, consulte [the section called “Crie 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 Go para usar o TLS.

Baixe os seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

 Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para confiar tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

Combine todos os certificados baixados em um único pem arquivo com o nome *keyspaces-bundle.pem* em nossos exemplos. Você pode fazer isso executando o seguinte comando : Anote o caminho para o arquivo, você precisará disso mais tarde.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
    AmazonRootCA4.pem \
    sf-class2-root.crt \
    > keyspaces-bundle.pem
```


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 keyspaces-bundle.pem
    cluster.SslOpts = &gocql.SslOptions{
        CaPath: "path_to_file/keyspaces-bundle.pem",
```

```

        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. "*path_to_file/keyspaces-bundle.pem*" Substitua pelo caminho para o arquivo de certificado combinado salvo na primeira etapa.
 - b. Certifique-se de que *ServiceUserName* e *ServicePassword* corresponda ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para. [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)
 - c. Para obter uma lista de endpoints disponíveis, consulte [the section called "Service endpoints"](#).
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 a entidade principal do IAM seguindo as etapas em [the section called “Crie credenciais do IAM para autenticação AWS”](#). Se um aplicativo estiver sendo executado no Lambda ou em uma instância do Amazon EC2, seu aplicativo estará automaticamente usando as credenciais da instância. Para executar este tutorial localmente, você pode armazenar as credenciais como variáveis de ambiente local.

[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

    // the authenticator uses the default credential chain to find AWS credentials
    cluster.Authenticator = sigv4.NewAwsAuthenticator()

    cluster.SslOpts = &gocql.SslOptions{
```

```
        CaPath: "path_to_file/keyspaces-bundle.pem",
        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. "*path_to_file/keyspaces-bundle.pem*" Substitua pelo caminho para o arquivo de certificado salvo na primeira etapa.
2. Para que este exemplo seja executado localmente, você precisa definir as seguintes variáveis como variáveis de ambiente:
 - AWS_ACCESS_KEY_ID
 - AWS_SECRET_ACCESS_KEY
 - AWS_DEFAULT_REGION
3. Para armazenar chaves de acesso fora do código, consulte as melhores práticas em [the section called “Gerenciar chaves de acesso”](#).
4. Para obter uma lista de endpoints disponíveis, consulte [the section called “Service endpoints”](#).

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

Os certificados Amazon Keyspaces estão migrando para a hierarquia do Amazon Trust Services (ATS). Certifique-se de que seu ambiente confie no Amazon Root CAs 1—4 para evitar erros de conexão durante essa rotação. 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 *ServiceUserName* e *ServicePassword* corresponda ao nome de usuário e à senha que você obteve ao gerar as credenciais específicas do serviço seguindo as etapas para [Crie credenciais específicas do serviço para acesso programático ao Amazon Keyspaces](#)

Note

Para obter uma lista de endpoints disponíveis, consulte [the section called “Service endpoints”](#).

4. Execute a aplicação .

```
perl cqlapp.pl
```

Configure o acesso entre contas para Amazon Keyspaces e endpoints da VPC

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 da VPC privados, consulte [the section called “Usar VPC endpoints da interface do ”](#).

Tópicos

- [Configure o acesso entre contas para o Amazon Keyspaces usando endpoints da VPC em uma VPC compartilhada](#)
- [Como configurar o acesso entre contas para o Amazon Keyspaces sem uma VPC compartilhada](#)

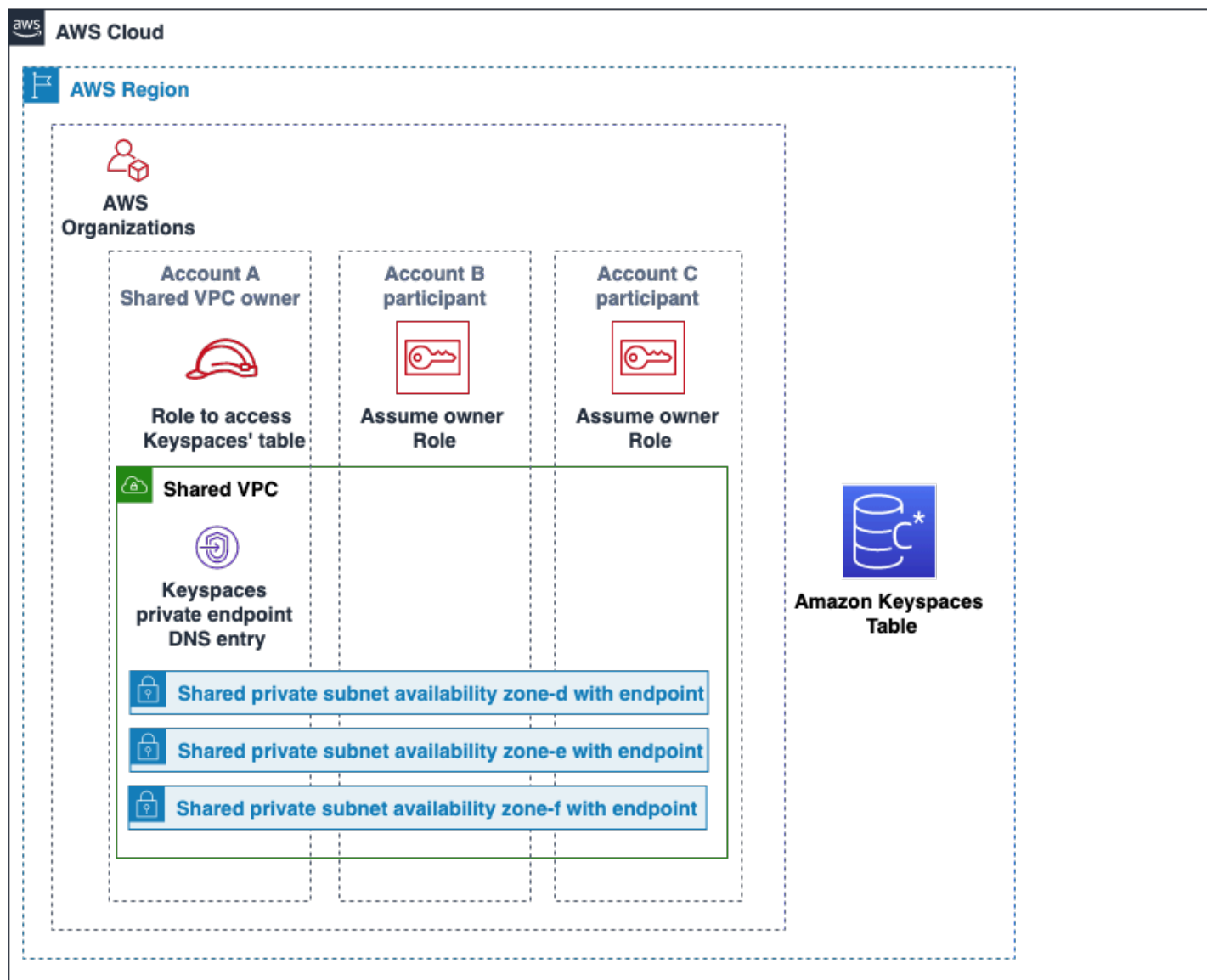
Configure o acesso entre contas para o Amazon Keyspaces usando endpoints da VPC em uma VPC compartilhada

Você pode criar recursos diferentes Contas da AWS ou separados 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:111111111111: essa conta contém infraestrutura, incluindo os endpoints da VPC, as sub-redes da VPC e as tabelas do Amazon Keyspaces.
- Account B:222222222222: essa conta contém um aplicativo em um ambiente de desenvolvimento que precisa se conectar à tabela do Amazon Keyspaces em Account A:111111111111.
- Account C:333333333333: essa conta contém um aplicativo em um ambiente de produção que precisa se conectar à tabela do Amazon Keyspaces em Account A:111111111111.



Account A:111111111111 é a conta que contém os recursos (uma tabela do Amazon Keyspaces) que Account B:222222222222 Account C:333333333333 você precisa acessar, assim como Account A:111111111111 a conta confiável. Account B:222222222222 e Account C:333333333333 são as contas com os diretores que precisam acessar os recursos (uma tabela do Amazon Keyspaces) Account A:111111111111, ou Account B:222222222222 seja, Account C:333333333333 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:111111111111.

Configuração para a **Account A:111111111111**

1. Use AWS Resource Access Manager para criar um compartilhamento de recursos para a sub-rede e compartilhar a sub-rede privada com e. Account B:222222222222 Account C:333333333333

Account B:222222222222 e Account C:333333333333 agora podem ver e criar recursos na sub-rede que foi compartilhada com eles.

2. Crie um endpoint 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 uma função do IAM Account A:111111111111 que tenha acesso total à tabela Amazon Keyspaces, leia as tabelas do sistema Amazon Keyspaces e seja capaz de descrever os recursos de 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 uma política de confiança para a função do IAM Account A:111111111111 para que ela Account B:222222222222 Account C:333333333333 possa assumir a função como contas confiáveis. Isso é mostrado no exemplo a seguir.

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

```

    "Effect": "Allow",
    "Principal": {
      "AWS": [
        "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
        "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
      ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {}
  }
]
}

```

Para obter mais informações sobre políticas de IAM entre contas, consulte [Cross-account as políticas](#) no Guia do usuário do IAM.

Configuração em **Account B:222222222222** e **Account C:333333333333**

1. Em Account B:222222222222 e Account C:333333333333, crie novos perfis e anexe a seguinte política que permite que a entidade principal assuma o perfil compartilhado criado em Account A:111111111111.

```

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

```

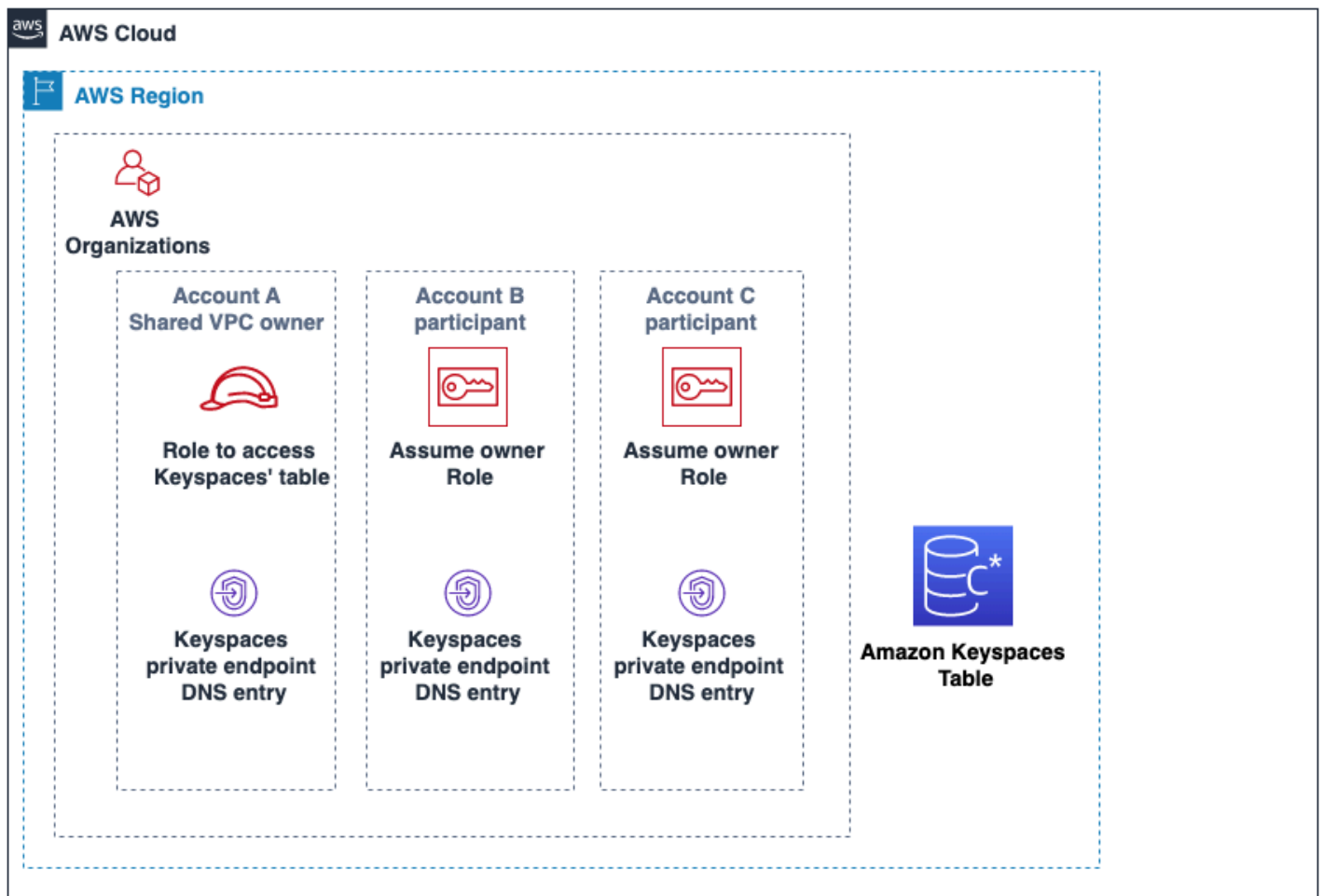
Permitir que o diretor assuma a função compartilhada é implementado usando a AssumeRole API do AWS Security Token Service (AWS STS). Para obter mais informações, consulte [Fornecer acesso a um usuário do IAM em outro Conta da AWS de sua propriedade](#) no Guia do usuário do IAM.

2. Em Account B:222222222222 e Account C:333333333333, 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:111111111111 por meio do endpoint da VPC na VPC compartilhada. Para obter mais informações sobre o plug-in de autenticação SIGV4, consulte [the section called “Crie credenciais programáticas de acesso”](#). Para obter mais informações sobre como configurar um aplicativo para assumir uma função em outra AWS conta, consulte [Autenticação e acesso](#) no Guia de referência de AWS SDKs e ferramentas.

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:111111111111, Account B:222222222222 e Account C:333333333333 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 VPC endpoint compartilhado usando os endpoints públicos ou implantando um endpoint VPC privado em cada conta. Quando não está usando uma VPC compartilhada, cada conta exige seu próprio endpoint da VPC. Neste exemplo, Account A:111111111111, Account B:222222222222 e Account C:333333333333 exigem seus próprios endpoints da VPC para acessar a tabela em Account A:111111111111. 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.



Account A:111111111111 é a conta que contém os recursos (uma tabela do Amazon Keyspaces) que Account B:222222222222 Account C:333333333333 você precisa acessar, assim como Account A:111111111111 a conta confiável. Account B:222222222222 e Account C:333333333333 são as contas com os diretores que precisam acessar os recursos (uma tabela do Amazon Keyspaces) Account A:111111111111, ou Account B:222222222222 seja, Account C:333333333333 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:111111111111.

Configuração para a **Account A:111111111111**

1. Crie um keyspace do Amazon Keyspaces e insira uma tabela. Account A:111111111111
2. Crie uma função do IAM Account A:111111111111 que tenha acesso total à tabela Amazon Keyspaces e acesso de leitura às tabelas do sistema Amazon Keyspaces.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select",
        "cassandra:Modify"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111111111111:/keyspace/mykeyspace/table/mytable",
        "arn:aws:cassandra:us-east-1:111111111111:/keyspace/system*"
      ]
    }
  ]
}
```

3. Configure uma política de confiança para a função do IAM em Account A:111111111111 para que os diretores Account C:333333333333 possam assumir a função como contas confiáveis. Account B:222222222222 Isso é mostrado no exemplo a seguir.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
          "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Para obter mais informações sobre políticas de IAM entre contas, consulte [Cross-account as políticas](#) no Guia do usuário do IAM.

- Configure o VPC endpoint Account A:111111111111 e anexe permissões ao endpoint que permitem que as funções de Account B:222222222222 e assumam Account C:333333333333 a função no uso Account A do VPC endpoint. Essas permissões são válidas para o VPC endpoint ao qual estão anexadas. Para obter mais informações sobre as políticas de endpoint da VPC, consulte [the section called “Como controlar o acesso aos endpoints da VPC de interface para o Amazon Keyspaces”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessfromSpecificIAMroles",
      "Effect": "Allow",
      "Action": "cassandra:*",
      "Resource": "*",
      "Principal": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
            "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
          ]
        }
      }
    }
  ]
}
```

Configuração em **Account B:222222222222** e **Account C:333333333333**

- Em Account B:222222222222 e Account C:333333333333, crie novos perfis e anexe a seguinte política que permite que a entidade principal assumo o perfil compartilhado criado em Account A:111111111111.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::111111111111:role/keyspaces_access"
```

```
}  
}
```

Permitir que o diretor assuma a função compartilhada 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 outro Conta da AWS de sua propriedade](#) no Guia do usuário do IAM.

2. Em Account B:222222222222 e Account C:333333333333, você pode criar aplicativos que utilizam o plug-in de autenticação SIGV4, que permite que um aplicativo assuma a função compartilhada para se conectar à tabela Amazon Keyspaces localizada em Account A:111111111111 Para obter mais informações sobre o plug-in de autenticação SIGV4, consulte [the section called “Crie credenciais programáticas de acesso”](#). Para obter mais informações sobre como configurar um aplicativo para assumir uma função em outra AWS conta, consulte [Autenticação e acesso](#) no Guia de referência de AWS SDKs e ferramentas.

Conceitos básicos do Amazon Keyspaces (para Apache Cassandra)

Se você não possui experiência com Apache Cassandra e Amazon Keyspaces, este tutorial orienta você na instalação dos programas e ferramentas necessários para usar o Amazon Keyspaces com sucesso. Você aprenderá a criar um keyspace e uma tabela usando a Cassandra Query Language (CQL) Console de gerenciamento da AWS, the ou the (). AWS Command Line Interface AWS CLI Em seguida, você usa Cassandra Query Language (CQL) para executar operações de criação, leitura, atualização e exclusão (CRUD) em dados na sua tabela Amazon Keyspaces.

Este tutorial inclui as seguintes etapas.

- Pré-requisitos — Antes de iniciar o tutorial, siga as instruções de AWS configuração para se cadastrar AWS e criar um usuário do IAM com acesso ao Amazon Keyspaces. Então você configura o `cqlsh-expansion` AWS CloudShell e. Como alternativa, você pode usar o AWS CLI para criar recursos no Amazon Keyspaces.
- Etapa 1: Crie um espaço de chave e uma tabela. Nesta seção, você criará um espaço chave chamado “catálogo” e uma tabela chamada “book_awards” dentro dele. Você especificará as colunas, os tipos de dados, a chave de partição e a coluna de agrupamento da tabela usando o Console de gerenciamento da AWS, CQL ou o. AWS CLI
- Etapa 2: Executar operações CRUD — Aqui, você usará o `cqlsh-expansion` in CloudShell para inserir, ler, atualizar e excluir dados na tabela “book_awards”. Você aprenderá a usar várias instruções CQL, como SELECT, INSERT, UPDATE e DELETE, e praticará a filtragem e a modificação de dados.
- Etapa 3: Limpe os recursos. Para evitar cobranças por recursos não utilizados, esta seção explica como excluir a tabela “book_awards” e o keyspace “catalog” usando o console, o CQL ou a AWS CLI.

Para obter tutoriais sobre como se conectar programaticamente ao Amazon Keyspaces usando diferentes drivers de cliente do Apache Cassandra, consulte [the section called “Uso de um driver de cliente Cassandra”](#). Para exemplos de código usando AWS SDKs diferentes, consulte [Exemplos de código para Amazon Keyspaces AWS](#) usando SDKs.

Tópicos

- [Pré-requisitos e considerações do tutorial](#)

- [Crie um keyspace no Amazon Keyspaces.](#)
- [Verifique o status de criação do keyspace no Amazon Keyspaces](#)
- [Crie uma tabela no Amazon Keyspaces.](#)
- [Verifique o status de criação da tabela no Amazon Keyspaces](#)
- [Crie, leia, atualize e exclua dados \(CRUD\) usando CQL no Amazon Keyspaces](#)
- [Exclua uma tabela do Amazon Keyspaces](#)
- [Excluir um keyspace no Amazon Keyspaces](#)

Pré-requisitos e considerações do tutorial

Antes de começar a usar o Amazon Keyspaces, siga as instruções de AWS configuração em. [Como acessar o Amazon Keyspaces \(para Apache Cassandra\)](#) Essas etapas incluem se inscrever AWS e criar um usuário AWS Identity and Access Management (IAM) com acesso ao Amazon Keyspaces.

Para concluir todas as etapas do tutorial, é necessário instalar o `cqlsh`. Você pode seguir as instruções de configuração em [Usar cqlsh para se conectar ao Amazon Keyspaces](#).

Para acessar o Amazon Keyspaces usando `cqlsh` ou o AWS CLI, recomendamos usar. AWS CloudShell CloudShell é um shell pré-autenticado baseado em navegador que você pode iniciar diretamente do. Console de gerenciamento da AWS Você pode executar comandos AWS Command Line Interface (AWS CLI) no Amazon Keyspaces usando seu shell preferido (Bash PowerShell ou Z shell). Para usar `cqlsh`, você deve instalar `cqlsh-expansion`. Para obter instruções de instalação de `cqlsh-expansion`, consulte [the section called “Como usar o cqlsh-expansion”](#). Para obter mais informações sobre, CloudShell consulte [the section called “Usando AWS CloudShell”](#).

Para usar o AWS CLI para criar, visualizar e excluir recursos no Amazon Keyspaces, siga as instruções de configuração em. [the section called “Baixando e configurando o AWS CLI”](#)

Depois de concluir as etapas de pré-requisito, vá para [Crie um keyspace no Amazon Keyspaces..](#)

Crie um keyspace no Amazon Keyspaces.

Nesta seção, você cria um keyspace usando o console, `cqlsh` ou AWS CLI.

Note

Antes de começar, certifique-se de que você configurou todos os [pré-requisitos do tutorial](#).

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:

- Instruções de Data Definition Language (DDL) na referência da linguagem CQL: [Keyspaces](#)
- [Cotas para Amazon Keyspaces \(para Apache Cassandra\)](#)

Neste tutorial, criamos um keyspace de região única, e a estratégia de replicação do keyspace é `SingleRegionStrategy`. Usando `SingleRegionStrategy`, o Amazon Keyspaces replica dados em três [zonas de disponibilidade](#) em uma Região da AWS. Para saber como criar chaves de réplica de várias regiões, consulte [the section called “Criar um espaço de chaves multirregional”](#).

Utilizar o console

Para criar um espaço de chave usando o console

1. Faça login no Console de gerenciamento da AWS 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 **catalog** como o nome do espaço de chaves.

Restrições de nome:

- Nome não pode estar vazio.
 - Caracteres permitidos: caracteres alfanuméricos e sublinhado (_).
 - O tamanho máximo é de 48 caracteres.
5. Em Regiões da AWS, confirme se a Single-Region replicação é a estratégia de replicação para o keyspace.
 6. Para criar o espaço de chaves, escolha Criar espaço de chaves.
 7. Verifique se o espaço de chave `catalog` foi criado fazendo o seguinte:
 - a. No painel de navegação, escolha Keyspaces.
 - b. Localize seu espaço de chave `catalog` 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 AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

A saída do comando deve ser algo parecido com isso.

```
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.
```

2. Crie seu espaço de chave usando o seguinte comando CQL.

```
CREATE KEYSPACE catalog 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.

3. Verifique se seu espaço de chave foi criado.

```
SELECT * from system_schema.keyspaces;
```

A saída deste comando deve ser semelhante a esta.

```
cqlsh> SELECT * from system_schema.keyspaces;

keyspace_name          | durable_writes | replication
```

```

-----+-----
+-----
      system_schema |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
      system_schema_mcs |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
              system |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system_multiregion_info |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
              catalog |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(5 rows)

```

Usando o AWS CLI

O procedimento a seguir cria um espaço de chave usando AWS CLI.

Para criar um espaço de teclas usando o AWS CLI

1. Para confirmar se seu ambiente está configurado, você pode executar o seguinte comando em CloudShell.

```
aws keyspaces help
```

2. Crie seu keyspace usando a seguinte AWS CLI declaração.

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

3. Verifique se o seu keyspace foi criado com a seguinte declaração AWS CLI

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

A saída deste comando deve ser semelhante a este exemplo.

```
{
  "keyspaceName": "catalog",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/",
  "replicationStrategy": "SINGLE_REGION"
}
```

```
}
```

Verifique o status de criação do keyspace 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 Console de gerenciamento da AWS, 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 chaves fica visível na tabela `system_schema_mcs keyspaces` 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 chaves que foi criado com sucesso, a saída da consulta é semelhante à seguinte.

```
keyspace_name | durable_writes | replication
-----+-----+-----
mykeyspace | true           | {...} 1 item
```

Crie uma tabela no Amazon Keyspaces.

Nesta etapa, você cria uma tabela usando o console, `cqlsh` ou AWS CLI.

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:

- Design de chave de partição: [the section called “Design de chave de partição”](#)
- Trabalhar com tabelas: [the section called “Verifique o status da criação da tabela”](#)
- Instruções DDL na referência da linguagem CQL: [Tabelas](#)
- Gerenciamento de recursos da tabela: [Como gerenciar 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, as chaves de partição e as colunas de clustering especificados.

Utilizar o console

O procedimento a seguir cria a tabela `book_awards` com essas colunas e tipos de dados.

<code>year</code>	<code>int</code>
<code>award</code>	<code>text</code>
<code>rank</code>	<code>int</code>
<code>category</code>	<code>text</code>
<code>book_title</code>	<code>text</code>
<code>author</code>	<code>text</code>
<code>publisher</code>	<code>text</code>

Para criar uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces.
3. Escolha `catalog` como o espaço de chave em que você deseja criar essa tabela.
4. Escolha Criar tabela.
5. Na caixa Nome da tabela, insira **book_awards** como nome da tabela.

Restrições de nome:

- 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.

year	int
award	text
rank	int
category	text
book_title	text
author	text
publisher	text

- a. Nome: insira o nome da coluna.

Restrições de nome:

- 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. Para adicionar outra coluna, selecione Add column.

7. Escolha `award` e `year` como chaves 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 `category` e `rank` como Colunas 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 de Colunas, escolha a categoria. Na lista Ordem, escolha ASC para classificar em ordem crescente nos valores dessa coluna. (Escolha DESC para ordem decrescente.)
 - c. Então, selecione Adicionar coluna de clustering e escolha classificação.
9. Na seção Configurações da tabela, escolha Configurações padrão.
10. Escolha Criar tabela.
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.

Usar SSL

Este procedimento cria uma tabela com as seguintes colunas e tipos de dados usando CQL. As colunas `year` e `award` são chaves de partição com `category` e `rank` como colunas de clustering. Juntas, elas formam a chave primária da tabela.

<code>year</code>	<code>int</code>
<code>award</code>	<code>text</code>
<code>rank</code>	<code>int</code>
<code>category</code>	<code>text</code>
<code>book_title</code>	<code>text</code>
<code>author</code>	<code>text</code>

<code>publisher</code>	<code>text</code>

Para criar uma tabela usando CQL

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

A saída do comando deve ser algo parecido com isso.

```
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.
```

2. No prompt do espaço de chave (`cqlsh:keyspace_name>`), crie sua tabela inserindo o código a seguir na janela de comando.

```
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)  
);
```

Note

ASC é a ordem de clustering padrão. Você também pode especificar DESC para ordem decrescente.

Observe que `year` e `award` são colunas de chave de partição. Em seguida, `category` e `rank` são as colunas de clustering ordenadas em ordem crescente (ASC). Juntas, essas colunas formam a chave primária da tabela.

3. Verifique se sua tabela foi criada.

```
SELECT * from system_schema.tables WHERE keyspace_name='catalog.book_awards' ;
```

A saída deve ser semelhante a esta.

```
keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | 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
-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+

(0 rows)
```

4. Verifique a estrutura da sua tabela.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'catalog' AND table_name
= 'book_awards';
```

A saída desta instrução deve ser semelhante a este exemplo.

```
keyspace_name | table_name | column_name | clustering_order | column_name_bytes
| kind      | position | type
-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
catalog | book_awards | year | none |
0x79656172 | partition_key | 0 | int
catalog | book_awards | award | none |
0x6177617264 | partition_key | 1 | text
catalog | book_awards | category | asc |
0x63617465676f7229 | clustering | 0 | text
catalog | book_awards | rank | asc |
0x72616e6b | clustering | 1 | int
catalog | book_awards | author | none |
0x617574686f72 | regular | -1 | text
catalog | book_awards | book_title | none |
0x6266666b5f7469746c65 | regular | -1 | text
```

```

catalog | book_awards | publisher | none |
0x7075626c6973686572 | regular | -1 | text

```

(7 rows)

Confirme se todas as colunas e tipos de dados estão conforme o esperado. A ordem das colunas pode ser diferente da instrução CREATE.

Usando o AWS CLI

Este procedimento cria uma tabela com as seguintes colunas e tipos de dados usando AWS CLI. As colunas `year` e `award` formam a chave de partição com `category` e `rank` como colunas de clustering.

```

year          int
award         text
rank          int
category      text
book_title    text
author        text
publisher     text

```

Para criar uma tabela usando o AWS CLI

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` e `rank`. As duas colunas de agrupamento usam a ordem de classificação crescente. (Para facilitar a leitura, o `schema-definition` do comando de criação de tabela nesta seção é dividido em linhas separadas.)

1. Você pode criar uma tabela usando a instrução a seguir.

```

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-1:111122223333:/keyspace/catalog/
table/book_awards"
}
```

2. 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-1:111122223333:/keyspace/catalog/
table/book_awards",
  "creationTimestamp": "2024-07-11T15:12:55.571000+00:00",
  "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": "2024-07-11T15:12:55.571000+00:00"
},
"encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
    "status": "DISABLED"
},
"defaultTimeToLive": 0,
"comment": {
    "message": ""
},
"replicaSpecifications": []
```

```
}
```

Para realizar operações CRUD (criar, ler, atualizar e excluir) nos dados da tabela, vá para [the section called “Operações de CRUD”](#).

Verifique o status de criação da tabela 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 Console de gerenciamento da AWS, 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
```

Crie, leia, atualize e exclua dados (CRUD) usando CQL no Amazon Keyspaces

Nesta etapa do tutorial, você aprenderá a inserir, ler, atualizar e excluir dados em uma tabela do Amazon Keyspaces usando instruções da linguagem de manipulação de dados (DML) CQL. No Amazon Keyspaces, você só pode criar declarações DML na linguagem CQL. Neste tutorial, você praticará a execução de instruções DML usando `cqlsh-expansion` com [AWS CloudShell](#) em Console de gerenciamento da AWS.

- **Inserção de dados:** esta seção aborda a inserção de registros únicos e múltiplos em uma tabela usando a instrução de `INSERT`. Você aprenderá a fazer upload de dados de um arquivo CSV e verificar inserções bem-sucedidas usando consultas de `SELECT`.
- **Leitura de dados:** aqui, você explorará diferentes variações da instrução de `SELECT` para recuperar dados de uma tabela. Os tópicos incluem selecionar todos os dados, selecionar colunas específicas, filtrar linhas com base nas condições usando a cláusula de `WHERE` e entender condições simples e compostas.
- **Atualização de dados:** nesta seção, você aprenderá a modificar os dados existentes em uma tabela usando a instrução de `UPDATE`. Você praticará a atualização de uma e várias colunas e, ao mesmo tempo, entenderá as restrições relacionadas à atualização de colunas de chave primária.
- **Exclusão de dados:** a seção final aborda a exclusão de dados de uma tabela usando a instrução de `DELETE`. Você aprenderá como excluir células específicas, linhas inteiras e as implicações da exclusão de dados versus a exclusão de toda a tabela ou do espaço chave.

Ao longo do tutorial, você encontrará exemplos, dicas e oportunidades para praticar a criação de suas próprias consultas CQL para vários cenários.

Tópicos

- [Inserção e carregamento de dados em uma tabela do Amazon Keyspaces](#)
- [Leia dados de uma tabela usando a instrução `SELECT` em CQL no Amazon Keyspaces](#)
- [Atualize dados em uma tabela do Amazon Keyspaces usando CQL](#)
- [Excluir dados de uma tabela usando a instrução `DELETE` em CQL](#)

Inserção e carregamento de dados em uma tabela do Amazon Keyspaces

Para criar dados em sua tabela `book_awards`, use a instrução `INSERT` para adicionar uma única linha.

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar `us-east-1` com sua própria região.

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

A saída do comando deve ser algo parecido com isso.

```
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.
```

2. 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 compatíveis, consulte [the section called “Níveis de consistência de gravação”](#). Observe que essa etapa não é necessária se você estiver usando o editor CQL no Console de gerenciamento da AWS.

```
CONSISTENCY LOCAL_QUORUM;
```

3. Para inserir um único registro, execute o comando a seguir no editor CQL.

```
INSERT INTO catalog.book_awards (award, year, category, rank, author, book_title,
publisher)
VALUES ('Wolf', 2023, 'Fiction', 3, 'Shirley Rodriguez', 'Mountain', 'AnyPublisher');
```

4. Verifique se os dados foram adicionados corretamente à tabela executando o comando a seguir.

```
SELECT * FROM catalog.book_awards;
```

A saída da instrução deve ter a aparência a seguir.

year	award	category	rank	author	book_title	publisher
2023	Wolf	Fiction	3	Shirley Rodriguez	Mountain	AnyPublisher

(1 rows)

Para inserir vários registros de um arquivo usando `cqlsh`

1. Baixe o arquivo CSV de amostra (`keyspaces_sample_table.csv`) contido no arquivo [samplemigration.zip](#). Descompacte o arquivo e anote o caminho até `keyspaces_sample_table.csv`.

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 Chateau	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 Ice	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

2. Abra AWS CloudShell Console de gerenciamento da AWS e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar `us-east-1` com sua própria região.

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

3. No prompt de `cqlsh` (`cqlsh>`), especifique um espaço de chave.

```
USE catalog ;
```

4. Defina a consistência de gravação como `LOCAL_QUORUM`. Para obter mais informações sobre os níveis de consistência compatíveis, consulte [the section called “Níveis de consistência de gravação”](#).

```
CONSISTENCY LOCAL_QUORUM;
```

5. Em Ações, no canto superior direito da tela, escolha Carregar arquivo para carregar o arquivo csv baixado anteriormente. AWS CloudShell Anote o caminho para o arquivo.
6. No prompt do espaço de chave (`cqlsh:catalog>`), execute a instrução a seguir.

```
COPY book_awards (award, year, category, rank, author, book_title, publisher) FROM
'/home/cloudshell-user/keyspaces_sample_table.csv' WITH header=TRUE ;
```

A saída da instrução deve ser semelhante a esta.

```
cqlsh:catalog> COPY book_awards (award, year, category, rank, author,
book_title, publisher)
FROM '/home/cloudshell-user/
keyspaces_sample_table.csv' WITH delimiter=',' AND header=TRUE ;
cqlsh current consistency level is LOCAL_QUORUM.
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy]:
{'numprocesses': '16', 'maxattempts': '1000'}
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy-from]:
{'ingestrate': '1500', 'maxparseerrors': '1000', 'maxinserterrors': '-1',
'maxbatchsize': '10', 'minbatchsize': '1', 'chunksize': '30'}
Reading options from the command line: {'delimiter': ',', 'header': 'TRUE'}
Using 16 child processes

Starting copy of catalog.book_awards with columns [award, year, category, rank,
author, book_title, publisher].
OSError: handle is closed      0 rows/s; Avg. rate:      0 rows/s
Processed: 9 rows; Rate:      0 rows/s; Avg. rate:      0 rows/s
9 rows imported from 1 files in 0 day, 0 hour, 0 minute, and 26.706 seconds (0
skipped).
```

7. Verifique se os dados foram adicionados corretamente à tabela executando a consulta a seguir.

```
SELECT * FROM book_awards ;
```

Você verá a saída a seguir.

year	award	category	rank	author	book_title
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas Example Books
2020	Wolf	Non-Fiction	2	Ana Carolina Silva	Science Today SomePublisher
2020	Wolf	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice AnyPublisher
2020	Kwesi Manu Prize	Fiction	1	Akua Mansa	Where did you go? SomePublisher
2020	Kwesi Manu Prize	Fiction	2	John Stiles	Yesterday Example Books

2020 Kwesi Manu Prize	Fiction	3	Nikki Wolf	Moving to the
Chateau AnyPublisher				
2020 Richard Roe	Fiction	1	Alejandro Rosalez	Long
Summer SomePublisher				
2020 Richard Roe	Fiction	2	Arnav Desai	
The Key Example Books				
2020 Richard Roe	Fiction	3	Mateo Jackson	Inside
the Whale AnyPublisher				

(9 rows)

Para saber mais sobre como usar `cqlsh COPY` para fazer upload de dados de arquivos csv para uma tabela do Amazon Keyspaces, consulte. [the section called “Como carregar dados usando cqlsh”](#)

Leia dados de uma tabela usando a instrução **SELECT** em CQL no Amazon Keyspaces

Na seção [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

- [Selecione todos os dados da tabela](#)
- [Selecione um subconjunto de colunas](#)
- [Selecione um subconjunto de linhas](#)

Selecione 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

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

2. Execute a seguinte consulta .

```
SELECT * FROM catalog.book_awards ;
```

Usar o caractere curinga (*) para o `column_list` selecionar todas as colunas. A saída da instrução deve ser semelhante ao seguinte exemplo.

year	award publisher	category	rank	author	book_title
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas
2020	Wolf	Non-Fiction	2	Ana Carolina Silva	Science Today
2020	Wolf	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice
2020	Kwesi Manu Prize	Fiction	1	Akua Mansa	Where did you go?
2020	Kwesi Manu Prize	Fiction	2	John Stiles	Yesterday
2020	Kwesi Manu Prize	Fiction	3	Nikki Wolf	Moving to the Chateau
2020	Richard Roe	Fiction	1	Alejandro Rosalez	Long Summer
2020	Richard Roe	Fiction	2	Arnav Desai	The Key

2020	Richard Roe	Fiction	3	Mateo Jackson	Inside
the Whale	AnyPublisher				

Selecione um subconjunto de colunas

Para consultar um subconjunto de colunas

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

2. Para recuperar apenas as colunas award, category e year, execute a seguinte consulta.

```
SELECT award, category, year FROM catalog.book_awards ;
```

A saída contém somente as colunas especificadas na ordem listada na instrução SELECT.

award	category	year
Wolf	Non-Fiction	2020
Wolf	Non-Fiction	2020
Wolf	Non-Fiction	2020
Kwesi Manu Prize	Fiction	2020
Kwesi Manu Prize	Fiction	2020
Kwesi Manu Prize	Fiction	2020
Richard Roe	Fiction	2020
Richard Roe	Fiction	2020
Richard Roe	Fiction	2020

Selecione um subconjunto de linhas

Ao consultar um grande conjunto de dados, escolha 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

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

2. Para recuperar somente os registros dos prêmios de um determinado ano, execute a consulta a seguir.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Wolf' ;
```

A instrução SELECT anterior retorna o seguinte resultado.

year	award	category	rank	author	book_title	publisher
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas	AnyPublisher
2020	Wolf	Non-Fiction	2	Ana Carolina Silva	Science Today	SomePublisher
2020	Wolf	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice	AnyPublisher

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 de chave de partição 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 que você pode usar na cláusula de `WHERE` precisam incluir todas ou um subconjunto das colunas na chave de partição da tabela. Se quiser usar somente um subconjunto das colunas na cláusula de `WHERE`, você deve incluir um conjunto contíguo de colunas de chave de partição da esquerda para a direita, começando com a coluna principal da chave de partição. Por exemplo, se as colunas da chave de partição forem `year`, `month` e `award`, você poderá usar as seguintes colunas na cláusula de `WHERE`:
 - `year`
 - `year E month`
 - `year E month E award`
- 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 `book_awards`:

- Encontre os vencedores dos prêmios Wolf 2020 e exiba os títulos e autores dos livros, ordenados por classificação.
- Mostre os vencedores do primeiro prêmio de todos os prêmios em 2020 e exiba os títulos dos livros e os nomes dos prêmios.

Atualize dados em uma tabela do Amazon Keyspaces usando CQL

Para atualizar os dados em sua tabela `book_awards`, 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 col3='1' ;
```

- 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 de `book_awards`, altere o nome de uma editora para vencedora do prêmio Wolf de não ficção em 2020.

```
UPDATE book_awards SET publisher='new Books' WHERE year = 2020 AND award='Wolf' AND category='Non-Fiction' AND rank=1;
```

Verifique se a editora está agora `new Books`.

```
SELECT * FROM book_awards WHERE year = 2020 AND award='Wolf' AND category='Non-Fiction' AND rank=1;
```

A instrução deve retornar a seguinte saída.

year	award	category	rank	author	book_title	publisher
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas	new Books

Experimente

Avançado: O vencedor do “Prêmio Kwezi Manu” de ficção de 2020 mudou de nome. Atualize esse registro para alterar o nome para 'Akua Mansa-House'.

Excluir dados de uma tabela usando a instrução **DELETE** em CQL

Para excluir dados na tabela `book_awards`, 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. Você pode usar a recuperação do Amazon Keyspaces Point-in-time (PITR) para ajudar a restaurar tabelas excluídas. Para saber mais, consulte [the section called “Backup e restauração com recuperação para um ponto no tempo”](#) Para saber como restaurar uma tabela excluída com o PITR ativado, consulte [the section called “Restaurar uma tabela excluída”](#).

Exclua as células

A exclusão de uma coluna de uma linha remove os dados da célula especificada. Quando você exibe essa coluna usando uma `SELECT` instrução, os dados são exibidos como `null`, embora um valor nulo não esteja 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 de `book_awards`, você pode ver que o título do livro que ganhou o primeiro preço do prêmio “Richard Roe” de 2020 é “Long Summer”. Imagine que esse título tenha sido recuperado e você precise excluir os dados dessa célula.

Como excluir uma célula específica

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar `us-east-1` com sua própria região.

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

2. Execute a seguinte consulta DELETE.

```
DELETE book_title FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe'
AND category='Fiction' AND rank=1;
```

3. Verifique se a solicitação de exclusão foi feita conforme o esperado.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
category='Fiction' AND rank=1;
```

A saída desta instrução tem a aparência a seguir.

year	award	category	rank	author	book_title	publisher
2020	Richard Roe	Fiction	1	Alejandro Rosalez	null	SomePublisher

Excluir linhas

Pode haver um momento em que você precise excluir uma linha inteira, por exemplo, para atender a uma solicitação de exclusão de dados. A sintaxe geral para excluir uma linha é a seguinte.

```
DELETE FROM table_name WHERE condition ;
```

Para excluir uma linha

1. Abra AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar *us-east-1* com sua própria região.

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

2. Execute a seguinte consulta DELETE.

```
DELETE FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
category='Fiction' AND rank=1;
```

3. Verifique se a exclusão foi feita conforme o esperado.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
category='Fiction' AND rank=1;
```

A saída dessa instrução fica assim depois que a linha foi excluída.

```
year | award | category | rank | author | book_title | publisher
-----+-----+-----+-----+-----+-----+-----
(0 rows)
```

Você pode excluir dados expirados automaticamente da sua tabela usando o Amazon Keyspaces Time to Live. Para obter mais informações, consulte [the section called “Dados expirados com vida útil”](#).

Exclua uma tabela do Amazon Keyspaces

Para evitar a cobrança por tabelas e dados desnecessários, exclua todas as tabelas 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.

É possível excluir uma tabela usando o console, o CQL ou AWS CLI. 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 Console de gerenciamento da AWS.

Para excluir uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS 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 `book_awards` 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 AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar `us-east-1` com sua própria região.

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

2. Exclua sua tabela inserindo a seguinte declaração.

```
DROP TABLE IF EXISTS catalog.book_awards ;
```

3. Verifique se sua tabela foi excluída.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = 'catalog' ;
```

A saída deve ser algo parecido com isso. Observe que isso pode levar algum tempo, portanto, execute novamente a instrução após um minuto, caso não veja esse resultado.

```
keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | 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
-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+

(0 rows)
```

Usando o AWS CLI

O procedimento a seguir exclui uma tabela e todos os seus dados que usam o AWS CLI.

Para excluir uma tabela usando o AWS CLI

1. Abra o CloudShell
2. Exclua sua tabela com a seguinte declaração.

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

3. Para verificar se sua tabela foi excluída, você pode listar todas as tabelas em um keyspace.

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

Você verá a saída a seguir. Observe que essa operação assíncrona pode levar algum tempo. Re-run o comando novamente após um curto período de tempo para confirmar que a tabela foi excluída.

```
{  
  "tables": []  
}
```

Excluir um keyspace no Amazon Keyspaces

Para evitar a cobrança por espaços de chave, exclua todos os espaços de chave que não estiver usando. 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.

Você pode excluir um espaço de chave usando o console, CQL ou AWS CLI.

Utilizar o console

O procedimento a seguir exclui um espaço de chave e todas as suas tabelas e dados usando o console.

Para excluir um espaço de chave usando o console

1. Faça login no Console de gerenciamento da AWS 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 `catalog` 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 `book_awards` 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 AWS CloudShell e conecte-se ao Amazon Keyspaces usando o comando a seguir. Certifique-se de atualizar `us-east-1` com sua própria região.

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

2. Exclua seu keyspace digitando a seguinte declaração.

```
DROP KEYSPACE IF EXISTS catalog ;
```

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 chaves seja excluído. Depois que o keyspace for excluído, a saída da instrução deverá ficar assim.

```
keyspace_name          | durable_writes | replication
```

```

-----+-----
+-----
      system_schema |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
      system_schema_mcs |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
              system |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system_multiregion_info |          True | {'class':
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(4 rows)

```

Usando o AWS CLI

O procedimento a seguir exclui um espaço de chave e todas as suas tabelas e dados usando o AWS CLI.

Para excluir um espaço de tecla usando o AWS CLI

1. Abra o AWS CloudShell
2. Exclua seu keyspace digitando a seguinte declaração.

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

3. Verifique se seu espaço de chave foi excluído.

```
aws keyspaces list-keyspaces
```

A saída dessa declaração deve ser semelhante a essa e listar apenas os espaços de teclas do sistema. Observe que, como essa é uma operação assíncrona, pode haver um atraso até que seu espaço de teclas seja excluído.

```

{
  "keyspaces": [
    {
      "keyspaceName": "system_schema",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_schema/",
      "replicationStrategy": "SINGLE_REGION"
    }
  ]
}

```

```
    },
    {
      "keyspaceName": "system_schema_mcs",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_schema_mcs/",
      "replicationStrategy": "SINGLE_REGION"
    },
    {
      "keyspaceName": "system",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system/",
      "replicationStrategy": "SINGLE_REGION"
    },
    {
      "keyspaceName": "system_multiregion_info",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_multiregion_info/",
      "replicationStrategy": "SINGLE_REGION"
    }
  ]
}
```


Tutoriais e soluções para Amazon Keyspaces

Este tópico inclui vários tutoriais que descrevem as etapas detalhadas necessárias para se conectar programaticamente ao Amazon Keyspaces ou usar o Amazon Keyspaces com outros serviços para criar soluções. AWS Os tutoriais disponíveis abrangem diferentes cenários populares de acesso entre serviços e demonstram como integrar o Amazon Keyspaces com a Amazon Virtual Private Cloud, o Amazon Elastic Kubernetes Service, o Amazon Simple Storage Service e/ou com tecnologias de código aberto, como o Apache Spark. AWS Glue

Para tutoriais passo a passo que mostram como se conectar ao Amazon Keyspaces usando drivers Apache Cassandra, consulte [the section called “Uso de um driver de cliente Cassandra”](#)

Tópicos

- [Tutorial: Conecte-se ao Amazon Keyspaces usando uma interface VPC endpoint](#)
- [Tutorial: Integre com o Apache Spark para importar ou exportar dados](#)
- [Tutorial: Conecte-se a partir de um aplicativo em contêiner hospedado no Amazon Elastic Kubernetes Service Amazon Elastic Kubernetes Service](#)
- [Tutorial: Exportar uma tabela do Amazon Keyspaces para o Amazon S3 usando AWS Glue](#)

Tutorial: Conecte-se ao Amazon Keyspaces usando uma interface VPC endpoint

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 obter 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 em [Como 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 [Introdução](#).

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:

- Escolha 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 conferir outras maneiras de se conectar, consulte [Conectar-se à instância do 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 Criar grupo de segurança. 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, vá para Grupos de segurança associados e insira 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 Salvar.
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 a permissão de conexões com a instância do Amazon EC2 e para obter instruções da AWS CLI, consulte [Authorize inbound traffic for your Linux instances](#) 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. Adicione regras ao grupo de segurança de suas VPCs que permitam acesso HTTP, HTTPS e SSH de entrada a partir de endereços IPv6.
7. Para confirmar que seu VPC endpoint foi configurado corretamente, você precisa usar uma conexão com o Amazon `cqlsh` Keyspaces. Se você usa seu ambiente local ou o editor Amazon Keyspaces CQL no Console de gerenciamento da AWS, 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 VPC de pilha dupla para o Amazon Keyspaces.

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

Nesta etapa, você cria um endpoint VPC de pilha dupla 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"
    "api.aws.us-east-1.cassandra-streams"
  ]
}

```

Se o Amazon Keyspaces for um dos serviços disponíveis na saída do comando, você poderá continuar com a criação de um VPC endpoint.

3. Para se conectar ao Amazon Keyspaces usando endpoints de pilha dupla habilitados para IPv6, confirme se sua VPC oferece suporte a IPv6 e configure sub-redes com suporte a IPv6. Para adicionar suporte a IPv6 a uma VPC existente que atualmente só oferece suporte a IPv4, consulte Suporte a IPv6 [para sua VPC no Guia do usuário da Amazon VPC](#);
4. Determine o identificador da VPC.

```

aws ec2 describe-vpcs

{
  "Vpcs": [
    {
      "OwnerId": "111122223333",
      "InstanceTenancy": "default",
      "Ipv6CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-0000aaa0a00a00aa0",
          "Ipv6CidrBlock": "2600:1f18:e19:7d00::/56",
          "Ipv6CidrBlockState": {

```



```

        "State": "associated"
      },
      "NetworkBorderGroup": "us-east-1",
      "Ipv6Pool": "Amazon",
      "Ipv6AddressAttribute": "public",
      "IpSource": "amazon"
    }
  ],
  "CidrBlockAssociationSet": [
    {
      "AssociationId": "vpc-cidr-assoc-00a0000a",
      "CidrBlock": "111.11.0.0/16",
      "CidrBlockState": {
        "State": "associated"
      }
    }
  ],
  "IsDefault": true,
  "BlockPublicAccessStates": {
    "InternetGatewayBlockMode": "off"
  },
  "VpcId": "vpc-a1234bcd",
  "State": "available",
  "CidrBlock": "111.11.0.0/16",
  "DhcpOptionsId": "dopt-a00aaaaa"
}
]
}

```

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

5. Use um filtro para coletar detalhes sobre as sub-redes da VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-a1234bcd"
```

```

{
  "Subnets": [
    {
      "AvailabilityZoneId": "use1-az1",
      "MapCustomerOwnedIpOnLaunch": false,
      "OwnerId": "111122223333",
      "AssignIpv6AddressOnCreation": false,
      "Ipv6CidrBlockAssociationSet": [
        {

```

```

        "AssociationId": "subnet-cidr-assoc-05d75732736740283",
        "Ipv6CidrBlock": "*****",
        "Ipv6CidrBlockState": {
            "State": "associated"
        },
        "Ipv6AddressAttribute": "public",
        "IpSource": "amazon"
    },
    "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-70b24b16",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
        "HostnameType": "ip-name",
        "EnableResourceNameDnsARecord": false,
        "EnableResourceNameDnsAAAARecord": false
    },
    "BlockPublicAccessStates": {
        "InternetGatewayBlockMode": "off"
    },
    "SubnetId": "subnet-70b24b16",
    "State": "available",
    "VpcId": "vpc-a1234bcd",
    "CidrBlock": "*****/20",
    "AvailableIpAddressCount": 4096,
    "AvailabilityZone": "us-east-1a",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true
},
{
    "AvailabilityZoneId": "use1-az2",
    "MapCustomerOwnedIpOnLaunch": false,
    "OwnerId": "111122223333",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [
        {
            "AssociationId": "subnet-cidr-assoc-0ec6fb253e05b17eb",
            "Ipv6CidrBlock": "*****",
            "Ipv6CidrBlockState": {
                "State": "associated"
            },
            "Ipv6AddressAttribute": "public",
            "IpSource": "amazon"
        }
    ]
}

```

```

        }
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/subnet-
c63ffbe7",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
        "HostnameType": "ip-name",
        "EnableResourceNameDnsARecord": false,
        "EnableResourceNameDnsAAAARecord": false
    },
    "BlockPublicAccessStates": {
        "InternetGatewayBlockMode": "off"
    },
    "SubnetId": "subnet-c63ffbe7",
    "State": "available",
    "VpcId": "vpc-a1234bcd",
    "CidrBlock": "*/20",
    "AvailableIpAddressCount": 4087,
    "AvailabilityZone": "us-east-1b",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true
    }
]
}

```

No exemplo de saída, há duas IDs de sub-rede disponíveis: subnet-70b24b16 e. subnet-c63ffbe7

6. Crie o endpoint da VPC. Para o parâmetro `--vpc-id`, especifique o ID da VPC da etapa anterior. Para o parâmetro `--subnet-ids`, especifique as IDs da sub-rede da etapa anterior. Use o parâmetro `--vpc-endpoint-type` para definir o endpoint como uma interface. Para criar um endpoint de pilha dupla, use. `--ip-address-type dualstack` 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 \
  --ip-address-type dualstack \
  --service-name com.amazonaws.us-east-1.cassandra \
  --subnet-ids subnet-70b24b16 subnet-c63ffbe7

```

```

{
  "VpcEndpoint": {
    "VpcEndpointId": "vpce-0000000abc111d2ef3",
    "VpcEndpointType": "Interface",
    "VpcId": "vpc-a1234bcd",
    "ServiceName": "com.amazonaws.us-east-1.cassandra",
    "State": "pending",
    "RouteTableIds": [],
    "SubnetIds": [
      "subnet-70b24b16",
      "subnet-c63ffbe7"
    ],
    "Groups": [
      {
        "GroupId": "sg-0123456789",
        "GroupName": "default"
      }
    ],
    "IpAddressType": "dualstack",
    "DnsOptions": {
      "DnsRecordIpType": "dualstack"
    },
    "PrivateDnsEnabled": true,
    "RequesterManaged": false,
    "NetworkInterfaceIds": [
      "eni-08cd525f72ea6f1fa",
      "eni-07b1f6c895169d8fb"
    ],
    "DnsEntries": [
      {
        "DnsName": "vpce-0000000000-1234567.cassandra.us-
east-1.vpce.amazonaws.com",
        "HostedZoneId": "Z7HUB22UULQXV"
      },
      {
        "DnsName": "vpce-0000000000-1234567-us-east-1a.cassandra.us-
east-1.vpce.amazonaws.com",
        "HostedZoneId": "Z7HUB22UULQXV"
      },
      {
        "DnsName": "cassandra.us-east-1.amazonaws.com",
        "HostedZoneId": "ZONEIDPENDING"
      }
    ]
  }
}

```

```
{
  {
    "DnsName": "cassandra.us-east-1.api.aws",
    "HostedZoneId": "ZONEIDPENDING"
  },
  "CreationTimestamp": "2025-09-19T15:19:19.266000+00:00",
  "OwnerId": "111122223333",
  "ServiceRegion": "us-east-1"
}
```

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 Gerenciar grupos de segurança e, em seguida, 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 definirá automaticamente o intervalo de portas 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 no Console de gerenciamento da AWS 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 Anexar políticas existentes diretamente.
 - e. Na lista de políticas, escolha e AmazonKeyspacesReadOnlyAccess, em seguida, escolha Avançar: Revisão.
 - f. Escolha 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 “Identity-based exemplos de políticas”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/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 no Console de gerenciamento da AWS 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 Anexar políticas existentes diretamente.
 - i. Na lista de políticas, escolha keyspacesVPCendpoint, e Próximo: Revisar.
 - j. Escolha 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 “Como usar o cqlsh-expansion”](#).

```
SELECT * FROM system.peers;
```

A saída retorna nós com endereços IP IPv6 privados, dependendo da configuração da VPC e da sub-rede na sua região. AWS

```
peer                                | data_center | host_id
| preferred_ip                      | rack       | release_version
| rpc_address                      | schema_version |
tokens
-----+-----
+-----+-----+-----+
+-----+-----+-----+
+-----+-----+-----+
+-----+-----+-----+
2600:1111:2222:3333:283b:8e6:d04f | us-east-1 | dddddd-7a22-3582-
a73d-49338a686a53 | 2600:1111:2222:3333:283b:8e6:d04f | us-east-1 |
3.11.2 | 2600:1111:2222:3333:283b:8e6:d04f | 05deae2d-6405-494d-a965-
c0e5836bcb3c | {'85070591730234615865843651857942052863'}
2600:1111:2222:4444:7d26:5a09:1b44 | us-east-1 | 66666666-035d-37ef-
a247-19a6a867ab09 | 2600:1111:2222:4444:7d26:5a09:1b44 | us-east-1 |
3.11.2 | 2600:1111:2222:4444:7d26:5a09:1b44 | 05deae2d-6405-494d-a965-
c0e5836bcb3c | {'170141183460469231731687303715884105726'}
```


 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 Console de gerenciamento da AWS, 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. É possível 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 em <https://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/PrivateLinkEndpoints.
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 Node.js driver. Para obter um exemplo completo de uma conexão de driver de Node.js cliente, consulte [the section called “Usando um driver de cliente Cassandra Node.js”](#).

O exemplo de código abaixo usa o *keyspaces-bundle.pem* arquivo do certificado digital. Consulte [the section called “Antes de começar”](#) para obter mais informações.

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/keyspaces-bundle.pem',  
      '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\":\"2000-00-00\",\"Statement\":[{\"Effect\":\"Allow\", \"Principal\": \"*\", \"Action\": \"*\", \"Resource\": \"*\"}]}\",
    "VpcId": "vpc-a1234bcd",
    "State": "available",
    "ServiceName": "com.amazonaws.us-east-1.cassandra",
    "RouteTableIds": [],
    "VpcEndpointId": "vpce-1a23b4c5",
    "CreationTimestamp": "2025-07-26T22:00:14Z"
  }
}
```

No exemplo de saída, o ID do VPC endpoint é **vpce-1a23b4c5**. Certifique-se de substituir esse valor pelo seu.

3. Exclua o endpoint da VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-1a23b4c5

{
  "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.

Tutorial: Integre com o Apache Spark para importar ou exportar dados

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 conector Spark Cassandra simplifica a execução de cargas de trabalho do Cassandra em pipelines de Spark-based análise usando um serviço de banco de dados totalmente gerenciado e 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 cargas de trabalho do Cassandra em Spark-based pipelines de análise.

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 read/write capacidade suficiente 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 read/write acesso à tabela de usuários e leia o acesso 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 melhores práticas para configurar capacidade de read/write transferência suficiente para sua tabela Amazon Keyspaces para suportar o tráfego do conector Spark Cassandra.
 - 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 ajuste de escala automático. Para obter mais informações, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).
 - 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 “Configurar modos read/write de capacidade”](#).

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 conector Spark Cassandra 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 Java. DataStax Você pode usar as credenciais específicas do serviço ou o plug-in SigV4 para se conectar.

Se ainda não tiver feito isso, você precisa converter o certificado digital usado para criar a conexão TLS 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 `application.conf` arquivo de exemplo que você pode usar ao se conectar com AWS credenciais 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 AWS configuração 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 “Crie 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 o `cqlsh-expansion`. Para obter instruções de instalação de `cqlsh-expansion`, consulte [the section called “Como usar o cqlsh-expansion”](#).

Substitua o endpoint do serviço no exemplo a seguir pelo seu próprio valor.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --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 “Verifique o status da criação do keyspace”](#).

```
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_table.csv")
```

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)
```

- 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()
```

- 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 função do IAM não tem as read/write permissões necessárias para a tabela de usuários e acesso de leitura às tabelas do 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 SSL/TLS conexão. Para obter 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 CloudWatch as métricas da Amazon para ajudá-lo a solucionar problemas com a configuração do seu conector Spark Cassandra no Amazon Keyspaces. Para saber mais sobre como usar o 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 emitirá uma `PerConnectionRequestRateExceeded` métrica em CloudWatch

Se você ver `PerConnectionRequestRateExceeded` eventos presentes 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 1.000 RCUs ou RRUs por segundo `second/3`, por partição. Se você estiver vendo

StoragePartitionThroughputCapacityExceeded CloudWatch eventos, isso pode indicar que os dados não são aleatórios 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.

Tutorial: Conecte-se a partir de um aplicativo em contêiner hospedado no Amazon Elastic Kubernetes Service Amazon Elastic Kubernetes Service

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

O Amazon EKS é um serviço gerenciado que elimina a necessidade de instalar, operar e manter um ambiente de gerenciamento do Kubernetes próprio. O [Kubernetes](#) é um sistema de código aberto que automatiza o gerenciamento, o ajuste de escala e a implantação de aplicações em contêineres.

O tutorial fornece orientação passo a passo 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 para conexão do Amazon EKS com o Amazon Keyspaces](#)
- [Etapa 1: configurar o cluster do Amazon EKS e configurar 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](#)
- [Etapa 5 \(opcional\): limpeza](#)

Pré-requisitos para conexão do Amazon EKS com o Amazon Keyspaces

Crie os seguintes AWS recursos antes de começar com o tutorial

1. Antes de começar este tutorial, siga as instruções AWS de configuração em [Como 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 espaço de chave foi criado, você pode usar o seguinte comando.

```
aws keyspaces list-keyspaces
```

Use o comando a seguir para criar a tabela:

```
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 a tabela deve estar listada, você pode usar o seguinte comando:

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

Para obter mais informações, consulte [criar espaço de chaves](#) e [criar tabela](#) na Referência de comandos AWS CLI .

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 confirmar que a tabela deve estar listada, você pode usar a instrução a seguir.

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

A tabela deve estar listada na saída desta instrução. Lembre-se que pode haver um atraso até que a tabela seja criada. Para obter 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 a implantação de pods do Kubernetes sem gerenciar instâncias do 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 deverá ver algo semelhante ao resultado a seguir.

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 instruções sobre a instalação do Docker em uma instância do Amazon EC2, consulte [Instalação do 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](#). É possível usar o seguinte comando 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 do Amazon EKS”](#).

Etapa 1: configurar o cluster do Amazon EKS e configurar 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 do Amazon EKS. Isso é necessário para usar perfis do IAM em 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 cluster `my-eks-cluster`. Se você tiver um cluster com um nome diferente, lembre-se de atualizar o nome em todos os comandos futuros.

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

- b. Confirme se o provedor de identidade OIDC foi registrado com IAM com o comando a seguir.

```
aws iam list-open-id-connect-providers --region us-east-1
```

A saída deve ser semelhante a esta. Anote o nome do recurso da Amazon (ARN) no OIDC, será necessário na próxima etapa ao criar uma política de confiança para a conta de serviço.

```
{
  "OpenIDConnectProviderList": [
    ..
    {
      "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
    }
  ]
}
```

2. Crie uma conta de serviço para o cluster do Amazon EKS. Contas de serviço fornecem uma identidade para processos 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 um perfil do IAM que a conta de serviço possa assumir para obter permissões para 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 de cluster criados para este tutorial. Você pode criar um namespace usando o seguinte comando:

```
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 o perfil do IAM a confiar em sua conta de serviço. Essa relação de confiança é necessária antes que uma entidade principal possa assumir uma função. Você precisa realizar as seguintes edições no arquivo:
- Para o `Principal`, insira o ARN que o IAM retornou ao comando `list-open-id-connect-providers`. 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.

É necessário anexar o arquivo de política de confiança na etapa a seguir ao criar o perfil do IAM.

```

cat >trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-namespace:my-eks-serviceaccount",
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}

```

```
    }  
  ]  
}  
EOF
```

Opcional: É possível também adicionar várias entradas nas condições `StringEquals` ou `StringLike` abaixo para permitir que várias contas de serviço ou namespaces assumam o perfil. Para permitir que sua conta de serviço assuma uma função do IAM em uma AWS conta diferente, consulte [as permissões Cross-account do IAM](#) no Guia do usuário do Amazon EKS.

3. Crie um perfil do IAM com o nome `my-iam-role` a ser assumido pela conta de serviço do Amazon EKS. Anexe o arquivo de política de confiança criado na etapa anterior ao perfil. A política de confiança especifica a conta de serviço e o provedor do OIDC em que o perfil 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 perfil do IAM ao Amazon Keyspaces anexando uma política de acesso.
 - a. Anexe uma política de acesso para definir as ações que o perfil 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 obter 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 ao perfil 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. Sinalize a conta de serviço com o nome do recurso da Amazon (ARN) do perfil do IAM que pode ser assumido. Certifique-se de atualizar o perfil ARN com seu ID da 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 o perfil do IAM e a conta de serviço estão configurados corretamente.
 - a. Confirme se a política de confiança do perfil do IAM está configurada corretamente com a seguinte instrução.

```
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.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount"
        }
      }
    }
  ]
}
```

```
}  
  }  
} ]  
}
```

- b. Confirme que a conta do serviço do Amazon EKS esteja anotada com o perfil 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, o perfil do IAM e configurar os relacionamentos e permissões necessários, prossiga para [the section called “Etapa 2: configurar o aplicativo”](#).

Etapa 2: configurar o aplicativo

Nesta etapa, você cria seu aplicativo para se conectar ao Amazon Keyspaces usando o plugin 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 diretório `resources` do arquivo `application.conf`.

- a. Configuração do aplicativo

Nesta etapa, você configura o plug-in de autenticação SigV4. Você poderá usar o seguinte exemplo 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 mais endpoints de serviço, consulte [the section called “Service endpoints”](#). 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.us-east-1.amazonaws.com:9142"]  
  basic.load-balancing-policy.local-datacenter = "us-east-1"  
  advanced.auth-provider {  
    class = software.aws.mcs.auth.SigV4AuthProvider  
    aws-region = "us-east-1"  
  }  
  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. É possível 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. Adicionar dependência de registro.

Sem registros, é impossível solucionar problemas de conexão. Neste tutorial, usamos `slf4j` como estrutura de registro e `logback.xml` para armazenar a saída do registro. Definimos o nível de registro em debug para estabelecer a conexão. É possível usar o seguinte comando para adicionar a dependência.

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

É possível usar o seguinte snippet de código para configurar o registro.

```
<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" />
  </root>
</configuration>
```

 Note

O nível debug 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. Nos exemplos, substitua as Regiões com suas próprias informações.

```
export CASSANDRA_HOST=cassandra.us-east-1.amazonaws.com:9142
export CASSANDRA_DC=us-east-1
```

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 arquivo JAR com todas as dependências incluídas no diretório target.

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 us-east-1
```

A saída deve ser como o exemplo a seguir:

```
"repositories": [
  {
```

```
"repositoryArn": "arn:aws:ecr:us-east-1:111122223333:repository/my-ecr-repository",
"registryId": "111122223333",
"repositoryName": "my-ecr-repository",
"repositoryUri": "111122223333.dkr.ecr.us-east-1.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.us-east-1.amazonaws.com/my-ecr-repository:latest .
```

5. Recupere um token de autenticação para inserir a imagem do Docker no Amazon ECR. Você pode fazer isso com o comando a seguir:

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 111122223333.dkr.ecr.us-east-1.amazonaws.com
```

6. Primeiro, verifique se há imagens existentes em seu repositório Amazon ECR. É possível usar o seguinte comando.

```
aws ecr describe-images --repository-name my-ecr-repository --region us-east-1
```

Então, insira a imagem do Docker no repositório. É possível usar o seguinte comando.

```
docker push 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository:latest
```


Etapa 4: implantar o aplicativo no Amazon EKS e gravar dados na sua tabela

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 informações a seguir:

- Nome do aplicativo: o nome do aplicativo. Para este tutorial, usamos `my-keyspaces-app`.
- Namespace do Kubernetes O namespace do cluster do Amazon EKS. Para este tutorial, usamos `my-eks-namespace`.
- Nome da conta de serviço Amazon EKS o nome da conta de serviço do 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 do Docker do Amazon ECR.
- AWS ID da conta sua ID AWS da conta.
- ARN do perfil do IAM o ARN do perfil do IAM criado para ser assumido pela conta de serviço. 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 executa o aplicativo que se conecta ao Amazon Keyspaces e grava dados em uma tabela.

1. Configure o arquivo `deployment.yaml`. Será necessário substituir os valores a seguir:

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

- `AWS_REGION`

Você pode usar o seguinte arquivo 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.us-east-1.amazonaws.com/my-ecr-repository:latest
          ports:
            - containerPort: 8080
          env:
            - name: CASSANDRA_HOST
              value: "cassandra.us-east-1.amazonaws.com:9142"
            - name: CASSANDRA_DC
              value: "us-east-1"
            - 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: "us-east-1"
```

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 mais detalhes, é possível usar o seguinte comando:

```
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://41ff7811d33ae4bc398755800abdc132335d51d74f218ba81da0700a6f8c67b
    Image:                             111122223333.dkr.ecr.us-east-1.amazonaws.com/
    my_eks_repository:latest
```

```

Image ID:      111122223333.dkr.ecr.us-east-1.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.us-east-1.amazonaws.com:9142
  CASSANDRA_DC:        us-east-1
  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:          us-east-1
  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.us-east-1.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.us-east-1.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.us-east-1.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 se 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 pelo 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.us-
east-1.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
$812/0x00000000801105e88@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 (opcional): limpeza

Siga essas etapas para remover os recursos criados neste tutorial.

Remova recursos criados neste tutorial

1. Exclua sua implantação. É possível usar o seguinte comando 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. É possível usar o seguinte comando para fazer isso:

```
eksctl delete cluster --name my-eks-cluster --region us-east-1
```

3. Exclua o perfil do IAM usado para a conta de serviço Amazon EKS com permissões de acesso ao Amazon Keyspaces. Primeiro, você precisa remover a política gerenciada anexada ao perfil.

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

Em seguida, você pode usar o seguinte comando para excluir o perfil.

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

Para obter mais informações, consulte [Exclusão de um perfil do IAM \(AWS CLI\)](#) no Guia do usuário do IAM.

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

```
aws ecr delete-repository \  
  --repository-name my-ecr-repository \  
  --force \  
  --region us-east-1
```

Note que a bandeira `force` é necessária para excluir um repositório que contém imagens. Para excluir sua imagem antes, você pode fazer isso executando o seguinte comando.

```
aws ecr batch-delete-image \  
  --repository-name my-ecr-repository \  
  --image-ids imageTag=latest \  
  --region us-east-1
```

Para obter mais informações, consulte [Excluir uma imagem](#) no Guia do usuário do Amazon Elastic Container Registry.

5. Exclua o espaço de chaves e a tabela do Amazon Keyspaces. A exclusão do espaço de chave exclui automaticamente todas as tabelas desse espaço de chave. Você pode usar uma das opções a seguir para fazê-lo:

AWS CLI

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

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

```
aws keyspaces list-keyspaces
```

Para excluir sua tabela antes, é possível usar o seguinte comando.

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

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

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

Para obter mais informações, consulte [excluir espaço de chave](#) e [excluir tabela](#) na Referência de comandos AWS CLI .

cqlsh

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

Para confirmar que o espaço de chave existe, você pode usar a instrução a seguir.

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

Seu espaço de chave não deve ser listado na saída desta instrução. Note que pode haver um atraso até a chave de espaço ser excluída. Para obter mais informações, consulte [the section called “DESCARTAR ESPAÇO DE CHAVES”](#).

Para excluir sua tabela antes, é possível usar o seguinte comando.

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

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

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

Sua tabela não deve estar listada na saída desta instrução. Note que pode haver um atraso até a tabela ser excluída. Para obter mais informações, consulte [the section called “DESCARTAR TABELA”](#).

Tutorial: Exportar uma tabela do Amazon Keyspaces para o Amazon S3 usando AWS Glue

Este tutorial mostra como exportar uma tabela do Amazon Keyspaces para um bucket do Amazon S3 usando AWS Glue. Neste tutorial, muitas etapas manuais são automatizadas usando scripts de shell.

disponíveis no [repositório Github](#) do Amazon Keyspaces. Usando esse processo, você pode exportar dados do Amazon Keyspaces para o Amazon S3 sem precisar configurar um cluster Spark.

Tópicos

- [Pré-requisitos para exportar dados do Amazon Keyspaces para o Amazon S3](#)
- [Etapa 1: Crie o bucket do Amazon S3, baixe as ferramentas necessárias e configure o ambiente](#)
- [Etapa 2: Configurar o AWS Glue trabalho que exporta a tabela Amazon Keyspaces](#)
- [Etapa 3: Execute o AWS Glue trabalho para exportar a tabela do Amazon Keyspaces para o bucket do Amazon S3 a partir do AWS CLI](#)
- [Etapa 4: \(Opcional\) Crie um gatilho para agendar o trabalho de exportação](#)
- [Etapa 5 \(opcional\): limpeza](#)

Pré-requisitos para exportar dados do Amazon Keyspaces para o Amazon S3

Confirme os seguintes pré-requisitos e crie os recursos do Amazon Keyspaces antes de começar com o tutorial

1. Antes de começar este tutorial, siga as instruções AWS de configuração em [Como 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. Os scripts deste tutorial usam suas credenciais e o padrão Região da AWS armazenados em um local conhecido. Para obter mais informações, consulte [the section called “Gerenciar chaves de acesso”](#).

O exemplo a seguir mostra como armazenar os valores necessários como variáveis de ambiente para o usuário padrão.

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-east-1
```

3. Para executar os scripts deste tutorial, você precisa dos seguintes softwares e ferramentas instalados em sua máquina:
 - [Java](#)

- [Apache Maven](#)
- [Git](#)
- [AWS CLI](#)

Este tutorial foi testado com AWS CLI 2, Java 17.0.13 e Apache Maven 3.8.7.

4. Você precisa de uma tabela do Amazon Keyspaces com dados de amostra para exportar posteriormente neste tutorial. Você pode usar sua própria tabela do Amazon Keyspaces ou criar uma tabela de amostra seguindo as etapas do tutorial. [Introdução](#)
 - a. Para instalar ocqlsh-expansion, siga as etapas em [the section called “Como usar o cqlsh-expansion”](#).
 - b. Confirme se o particionador Murmur3Partitioner é o particionador padrão da conta. Esse particionador é compatível com o conector Apache Spark Cassandra e com AWS Glue. Para obter mais informações sobre particionadores, consulte [the section called “Como trabalhar com particionadores”](#).

Para alterar o particionador da sua conta, você pode usar a seguinte declaração.

```
SELECT partitioner FROM system.local;  
  
UPDATE system.local set  
  partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

- c. Para criar um keyspace do Amazon Keyspaces, siga as etapas em [the section called “Criar um keyspace”](#)
- d. Para criar a tabela Amazon Keyspaces, siga as etapas em [the section called “Criar uma tabela”](#)
- e. Para carregar dados de amostra na tabela para exportar para o Amazon S3, siga as etapas em [the section called “Criar”](#)

Depois de concluir as etapas de pré-requisito, vá para [the section called “Etapa 1: criar o bucket do Amazon S3, baixar ferramentas e configurar o ambiente”](#).

Etapa 1: Crie o bucket do Amazon S3, baixe as ferramentas necessárias e configure o ambiente

Nesta etapa, você baixa as ferramentas externas e cria e configura os AWS recursos necessários para a solução automatizada de exportação de dados de uma tabela do Amazon Keyspaces para um bucket do Amazon S3 usando um trabalho. AWS Glue Para realizar todas essas tarefas de forma eficiente, executamos um script de shell com o nome `setup-connector.sh` disponível no [Github](#).

O script `setup-connector.sh` automatiza as etapas a seguir.

1. Cria um bucket do Amazon S3 usando CloudFormation. Esse bucket armazena os arquivos jar e de configuração baixados, bem como os dados da tabela exportada.
2. Cria uma função do IAM usando CloudFormation. AWS Glue os trabalhos usam essa função para acessar o Amazon Keyspaces e o Amazon S3.
3. Faz o download do [conector Apache Spark Cassandra](#) e o carrega no bucket do Amazon S3.
4. Faz o download do [plug-in de autenticação SigV4](#) e o carrega no bucket do Amazon S3.
5. Faz o download das [extensões do Apache Spark e as](#) carrega no bucket do Amazon S3.
6. Faz o download da [Keyspaces Retry Policy](#) do Github, compila o código usando o Maven e carrega a saída para o bucket do Amazon S3.
7. Carrega o **keyspaces-application.conf** arquivo no bucket do Amazon S3.

Use o script de **setup-connector.sh** shell para automatizar as etapas de instalação e configuração.

1. Copie os arquivos do repositório [aws-glue](#) no Github para sua máquina local. Esse diretório contém o script do shell, bem como outros arquivos necessários.
2. Execute o script do shell `setup-connector.sh`. Você pode especificar os três parâmetros opcionais a seguir.
 1. **SETUP_STACKNAME**— Esse é o nome da CloudFormation pilha usada para criar os AWS recursos.
 2. **S3_BUCKET_NAME**— Esse é o nome do bucket do Amazon S3.
 3. **GLUE_SERVICE_ROLE_NAME**— Esse é o nome da função de serviço do IAM AWS Glue usada para executar trabalhos que se conectam ao Amazon Keyspaces e ao Amazon S3.

Você pode usar o comando a seguir para executar o script de shell e fornecer os três parâmetros com os nomes a seguir.

```
./setup-connector.sh cfn-setup s3-keyspaces iam-export-role
```

Para confirmar que seu bucket foi criado, você pode usar o AWS CLI comando a seguir.

```
aws s3 ls s3://s3-keyspaces
```

A saída do comando deve ter a seguinte aparência.

```
PRE conf/  
PRE jars/
```

Para confirmar que a função do IAM foi criada e analisar os detalhes, você pode usar a seguinte AWS CLI declaração.

```
aws iam get-role --role-name "iam-export-role"
```

```
{  
  "Role": {  
    "Path": "/",  
    "RoleName": "iam-export-role",  
    "RoleId": "AKIAIOSFODNN7EXAMPLE",  
    "Arn": "arn:aws:iam::111122223333:role/iam-export-role",  
    "CreateDate": "2025-01-28T16:09:03+00:00",  
    "AssumeRolePolicyDocument": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Effect": "Allow",  
          "Principal": {  
            "Service": "glue.amazonaws.com"  
          },  
          "Action": "sts:AssumeRole"  
        }  
      ]  
    },  
  },  
}
```

```
    "Description": "AWS Glue service role to import and export data from Amazon  
    Keyspaces",  
    "MaxSessionDuration": 3600,  
    "RoleLastUsed": {  
        "LastUsedDate": "2025-01-29T12:03:54+00:00",  
        "Region": "us-east-1"  
    }  
}  
}
```

Se o processo de CloudFormation empilhamento falhar, você poderá revisar as informações de erro detalhadas sobre a pilha com falha no CloudFormation console.

Depois que o bucket do Amazon S3 contendo todos os scripts e ferramentas for criado e a função do IAM estiver configurada, prossiga para. [the section called “Etapa 2: Configurar o AWS Glue trabalho”](#)

Etapa 2: Configurar o AWS Glue trabalho que exporta a tabela Amazon Keyspaces

Na segunda etapa do tutorial, você usa o script `setup-export.sh` disponível no [Github](#) para criar e configurar o AWS Glue trabalho que se conecta ao Amazon Keyspaces usando o plug-in SigV4 e, em seguida, exporta a tabela especificada para o bucket do Amazon S3 criado na etapa anterior. O uso do script permite que você exporte dados do Amazon Keyspaces sem configurar um cluster do Apache Spark.

Crie um AWS Glue trabalho para exportar uma tabela do Amazon Keyspaces para um bucket do Amazon S3.

- Nesta etapa, você executa o script de `setup-export.sh` shell localizado no `export-to-s3/` diretório a ser usado CloudFormation para criar e configurar o trabalho de AWS Glue exportação. O script usa os seguintes parâmetros.

PARENT_STACK_NAME, EXPORT_STACK_NAME, KEYSPACE_NAME, TABLE_NAME, S3_URI, FORMAT

- PARENT_STACK_NAME— O nome da CloudFormation pilha criada na etapa anterior.
- EXPORT_STACK_NAME— O nome da CloudFormation pilha que cria o trabalho de AWS Glue exportação.

- **KEYSPACE_NAME** e **TABLE_NAME** — O nome totalmente qualificado do espaço de teclas e da tabela a serem exportados. Para este tutorial, usamos `catalog.book_awards`, mas você pode substituí-lo por seu próprio nome de tabela totalmente qualificado.
- **S3URI** — O URI opcional do bucket do Amazon S3. O padrão é o bucket Amazon S3 da pilha principal.
- **FORMAT** — O formato de dados opcional. O valor padrão é `parquet`. Neste tutorial, para facilitar o carregamento e a transformação de dados, usamos o padrão.

Você pode usar o seguinte comando como exemplo.

```
setup-export.sh cfn-setup cfn-glue catalog book_awards
```

Para confirmar que o trabalho foi criado, você pode usar a seguinte declaração.

```
aws glue list-jobs
```

A saída da instrução deve ser semelhante a esta.

```
{
  "JobNames": [
    "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue"
  ]
}
```

Para ver os detalhes do trabalho, você pode usar o comando a seguir.

```
aws glue get-job --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

A saída do comando mostra todos os detalhes do trabalho. Isso inclui os argumentos padrão que você pode substituir ao executar o trabalho.

```
{
  "Job": {
    "Name": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue",
    "JobMode": "SCRIPT",
    "JobRunQueuingEnabled": false,
    "Description": "export to s3",
```

```

    "Role": "iam-export-role",
    "CreatedOn": "2025-01-30T15:53:30.765000+00:00",
    "LastModifiedOn": "2025-01-30T15:53:30.765000+00:00",
    "ExecutionProperty": {
      "MaxConcurrentRuns": 1
    },
    "Command": {
      "Name": "glueetl",
      "ScriptLocation": "s3://s3-keyspaces/scripts/cfn-setup-cfn-glue-
export.scala",
      "PythonVersion": "3"
    },
    "DefaultArguments": {
      "--write-shuffle-spills-to-s3": "true",
      "--S3_URI": "s3://s3-keyspaces",
      "--TempDir": "s3://s3-keyspaces/shuffle-space/export-sample/",
      "--extra-jars": "s3://s3-keyspaces/jars/spark-cassandra-
connector-assembly_2.12-3.1.0.jar,s3://s3-keyspaces/jars/aws-sigv4-auth-
cassandra-java-driver-plugin-4.0.9-shaded.jar,s3://s3-keyspaces/jars/spark-
extension_2.12-2.8.0-3.4.jar,s3://s3-keyspaces/jars/amazon-keyspaces-helpers-1.0-
SNAPSHOT.jar",
      "--class": "GlueApp",
      "--user-jars-first": "true",
      "--enable-metrics": "true",
      "--enable-spark-ui": "true",
      "--KEYSPACE_NAME": "catalog",
      "--spark-event-logs-path": "s3://s3-keyspaces/spark-logs/",
      "--enable-continuous-cloudwatch-log": "true",
      "--write-shuffle-files-to-s3": "true",
      "--FORMAT": "parquet",
      "--TABLE_NAME": "book_awards",
      "--job-language": "scala",
      "--extra-files": "s3://s3-keyspaces/conf/keyspaces-application.conf",
      "--DRIVER_CONF": "keyspaces-application.conf"
    },
    "MaxRetries": 0,
    "AllocatedCapacity": 4,
    "Timeout": 2880,
    "MaxCapacity": 4.0,
    "WorkerType": "G.2X",
    "NumberOfWorkers": 2,
    "GlueVersion": "3.0"
  }
}

```

```
}
```

Se o processo de CloudFormation empilhamento falhar, você poderá revisar os erros da pilha com falha no CloudFormation console. Você pode revisar os detalhes do trabalho de exportação no AWS Glue console escolhendo trabalhos ETL no menu do lado esquerdo.

Depois de confirmar os detalhes do trabalho de AWS Glue exportação, execute o trabalho [the section called “Etapa 3: Executar o AWS Glue trabalho de exportação a partir do AWS CLI”](#) para exportar os dados da sua tabela do Amazon Keyspaces.

Etapa 3: Execute o AWS Glue trabalho para exportar a tabela do Amazon Keyspaces para o bucket do Amazon S3 a partir do AWS CLI

Nesta etapa, você usa o AWS CLI para executar o AWS Glue trabalho criado na etapa anterior para exportar uma tabela do Amazon Keyspaces para o seu bucket no Amazon S3.

Execute o trabalho de exportação a partir do AWS CLI

1. No exemplo a seguir, o AWS CLI comando executa o trabalho criado na etapa anterior.

```
aws glue start-job-run --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

- Você pode substituir qualquer um dos parâmetros do AWS Glue trabalho, incluindo os argumentos padrão no AWS CLI comando. Para substituir qualquer argumento padrão do trabalho, por exemplo, espaço de chave ou nome da tabela, você pode passá-los como argumentos. Para obter uma lista completa dos argumentos, consulte [start-job-run](#) na Referência da Linha de Comando AWS Glue .

O comando a seguir executa o trabalho de AWS Glue exportação, mas substitui o número de AWS Glue trabalhadores, o tipo de trabalhador e o nome da tabela.

```
aws glue start-job-run --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue \
    --number-of-workers 8 --worker-type G.2X \
    --arguments '{"--TABLE_NAME": "my_table"}'
```

2. Confirme se sua tabela foi exportada para seu bucket do Amazon S3. Com base no tamanho da tabela, isso pode levar algum tempo. Quando o trabalho de exportação estiver concluído, você poderá ver as seguintes pastas no bucket usando o comando de exemplo.


```
aws s3 ls s3://s3-keyspaces
```

A saída mostra a seguinte estrutura em seu bucket.

```
PRE conf/  
PRE export/  
PRE jars/  
PRE scripts/  
PRE spark-logs/
```

Seus arquivos estarão localizados na seguinte estrutura de pastas em `export`, `data/time` os valores mostrarão seus próprios valores.

```
\----- export  
      \----- keyspace_name  
            \----- table_name  
                  \----- snapshot  
                        \----- year=2025  
                              \----- month=01  
                                    \----- day=02  
                                          \----- hour=09  
                                                \----- minute=22  
                                                    \--- YOUR DATA HERE
```

Para agendar o AWS Glue trabalho que você acabou de executar manualmente, vá para [the section called “Etapa 4: \(Opcional\) Programar o trabalho de exportação”](#).

Etapa 4: (Opcional) Crie um gatilho para agendar o trabalho de exportação

Para executar regularmente o trabalho de exportação criado na etapa anterior, você pode criar um gatilho programado. Para obter mais informações, consulte [AWS Glue gatilhos](#) no Guia do AWS Glue desenvolvedor.

Agende um AWS Glue trabalho

1. O AWS CLI comando a seguir é um exemplo de um gatilho simples com o nome `KeyspacesExportWeeklyTrigger` que executa o AWS Glue trabalho com o nome

AmazonKeyspacesExportToS3-cfn-setup-cfn-glue uma vez por semana na segunda-feira às 12:00 UTC.

```
aws glue create-trigger \  
  --name KeyspacesExportWeeklyTrigger \  
  --type SCHEDULED \  
  --schedule "cron(0 12 ? * MON *)" \  
  --start-on-creation \  
  --actions '[  
    "JobName": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue"  
  ]'
```

- Para substituir qualquer uma das configurações padrão do trabalho agendado, você pode passá-las como argumentos. Neste exemplo, substituímos o nome do keyspace, o nome da tabela, o número de trabalhadores e o tipo de trabalhador passando-os como argumentos. O comando a seguir é um exemplo disso.

```
aws glue create-trigger \  
  --name KeyspacesExportWeeklyTrigger \  
  --type SCHEDULED \  
  --schedule "cron(0 12 ? * MON *)" \  
  --start-on-creation \  
  --actions '[  
    "JobName": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue",  
    "Arguments": {  
      "--number-of-workers": "8",  
      "--worker-type": "G.2X",  
      "--table_name": "my_table",  
      "--keyspace_name": "my_keyspace"  
    }  
  ]'
```

2. Para confirmar que o gatilho foi criado, use o comando a seguir.

```
aws glue list-triggers
```

A saída do comando deve ser semelhante a essa.

```
{  
  "TriggerNames": [  
    "KeyspacesExportWeeklyTrigger"  
  ]  
}
```

```
}
```

Para limpar os AWS recursos criados neste tutorial, vá para [the section called “Etapa 5 \(opcional\): limpeza”](#).

Etapa 5 (opcional): limpeza

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

Remova recursos criados neste tutorial

1. Exclua a segunda CloudFormation pilha criada neste tutorial. Isso remove a AWS Glue tarefa e o gatilho criados neste tutorial. É possível usar o comando da a seguir para fazer isso.

```
aws cloudformation delete-stack --stack-name cfn-glue
```

2. Exclua o bucket do Amazon S3 junto com todos os dados armazenados nele. É possível usar o seguinte comando para fazer isso:

```
aws s3 rm s3://s3-keyspaces --recursive
```

3. Exclua a primeira pilha criada neste tutorial. Isso exclui a função do IAM e as permissões associadas criadas neste tutorial. Você pode usar o seguinte comando como exemplo.

```
aws cloudformation delete-stack --stack-name cfn-setup
```

4. Exclua o espaço de chaves e a tabela do Amazon Keyspaces. A exclusão do espaço de chave exclui automaticamente todas as tabelas desse espaço de chave. Você pode usar uma das opções a seguir para fazê-lo:

AWS CLI

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

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

```
aws keyspaces list-keyspaces
```

Para excluir sua tabela antes, é possível usar o seguinte comando.

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

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

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

Para obter mais informações, consulte [excluir espaço de chave](#) e [excluir tabela](#) na Referência de comandos AWS CLI .

cqlsh

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

Para confirmar que o espaço de chave existe, você pode usar a instrução a seguir.

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

Seu espaço de chave não deve ser listado na saída desta instrução. Note que pode haver um atraso até a chave de espaço ser excluída. Para obter mais informações, consulte [the section called “DESCARTAR ESPAÇO DE CHAVES”](#).

Para excluir sua tabela antes, é possível usar o seguinte comando.

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

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

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

Sua tabela não deve estar listada na saída desta instrução. Note que pode haver um atraso até a tabela ser excluída. Para obter mais informações, consulte [the section called “DESCARTAR TABELA”](#).

Como gerenciar os 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 carga de trabalho por meio de nós em um cluster, o Amazon Keyspaces aloca recursos de armazenamento e taxa de transferência diretamente às tabelas. read/write

O Amazon Keyspaces provisiona o armazenamento automaticamente com base nos dados armazenados em suas tabelas. Ele escalona o armazenamento à medida que você grava, atualiza e exclui dados, e paga somente pelo armazenamento que usa. Os dados são replicados em várias [zonas de disponibilidade](#) para alta disponibilidade. 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 “Estimar o tamanho da linha”](#).

Este capítulo aborda os principais aspectos do gerenciamento de recursos no Amazon Keyspaces.

- **Estimar o tamanho da linha:** para estimar o tamanho codificado das linhas no Amazon Keyspaces, considere fatores como metadados de chave de partição, metadados de colunas em cluster, identificadores de coluna, tipos de dados e metadados de linha. Esse tamanho de linha codificado é usado para faturamento, gerenciamento de cotas e planejamento de capacidade de throughput provisionada.
- **Estimar o consumo de capacidade:** esta seção aborda exemplos de como estimar o consumo de capacidade de leitura e gravação para cenários comuns, como consultas de intervalo, consultas de limite, varreduras de tabelas, transações leves, colunas estáticas e tabelas multirregionais. Você pode usar CloudWatch a Amazon para monitorar a utilização real da capacidade. Para obter mais informações sobre o monitoramento com CloudWatch, consulte [the section called “Monitoramento com CloudWatch”](#).
- **Configurar modos de read/write capacidade** — Você pode escolher entre dois modos de capacidade para processar leituras e gravações em suas tabelas:
 - **On-demand modo (padrão)** — Pague por solicitação pela taxa de transferência de leitura e gravação. O Amazon Keyspaces pode escalar instantaneamente a capacidade para qualquer nível de tráfego alcançado previamente.

- Modo provisionado: especifique o número necessário de unidades de capacidade de leitura e gravação com antecedência. Esse modo ajuda a manter o desempenho previsível do throughput.
- Gerencie a capacidade de throughput com ajuste de escala automático: para tabelas provisionadas, você pode ativar o ajuste de escala automático para ajustar a capacidade de throughput automaticamente com base no tráfego real do aplicativo. O Amazon Keyspaces usa o rastreamento de metas para aumentar ou diminuir a capacidade provisionada, mantendo a utilização dentro da meta especificada.
- Usar a capacidade de expansão de forma eficaz: o Amazon Keyspaces fornece capacidade de expansão reservando uma parte do throughput não utilizado para lidar com picos de tráfego. Essa flexibilidade permite expansões ocasionais de atividade além do throughput provisionado.

Para solucionar erros de capacidade, consulte [the section called “Erros de capacidade de tecnologia sem servidor”](#).

Tópicos

- [Estimar o tamanho da linha no Amazon Keyspaces](#)
- [Estimar o consumo de capacidade do throughput de leitura e gravação no Amazon Keyspaces](#)
- [Configurar modos de read/write capacidade no Amazon Keyspaces](#)
- [Gerencia automaticamente a capacidade de throughput com o ajuste de escala automático do Amazon Keyspaces](#)
- [Use a capacidade de expansão de forma eficaz no Amazon Keyspaces](#)

Estimar 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.

Este tópico 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 pode usar o tamanho da linha codificada ao estimar os requisitos de capacidade de transferência provisionada para tabelas.

Para calcular o tamanho codificado das linhas no Amazon Keyspaces, é possível usar as diretrizes a seguir.

Tópicos

- [Estime o tamanho codificado das colunas](#)
- [Estime o tamanho codificado dos valores de dados com base no tipo de dados](#)
- [Considere o impacto dos recursos do Amazon Keyspaces no tamanho da linha](#)
- [Escolha a fórmula certa para calcular o tamanho codificado de uma linha](#)
- [Exemplo de cálculo do tamanho da linha](#)

Estime o tamanho codificado das colunas

Esta seção mostra como estimar o tamanho codificado das colunas no Amazon Keyspaces.

- Colunas regulares — Para colunas regulares, que não são chaves primárias, colunas de agrupamento ou STATIC colunas, use o tamanho bruto dos dados da célula com base no [tipo de dados](#) e adicione os metadados necessários. Os tipos de dados e algumas diferenças importantes na forma como o Amazon Keyspaces armazena valores de tipos de dados e metadados estão listados na próxima seção.
- Colunas de chave de partição — As chaves de partição podem conter até 2048 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.
- Colunas de agrupamento — As colunas de agrupamento 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.

Note

Para oferecer suporte a consultas eficientes e indexação integrada, o Amazon Keyspaces armazena o valor dos dados de cada chave de partição e coluna de chave de agrupamento duas vezes.

- Nomes de coluna — 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”](#).

Estime o tamanho codificado dos valores de dados com base no tipo de dados

Esta seção mostra como estimar o tamanho codificado de diferentes tipos de dados no Amazon Keyspaces.

- Tipos de string — Os tipos de dados Cassandra ASCIITEXT,, e VARCHAR string 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. UTF-8
- Tipos numéricos — CassandraINT,,, BIGINT SMALLINTTINYINT, e VARINT os tipos de dados 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.
- Tipo de blob — A BLOB no Amazon Keyspaces é armazenado com o comprimento bruto do byte do valor.
- Tipo booleano — O tamanho de um Boolean valor ou Null valor é de 1 byte.
- Tipos de coleção — Uma coluna que armazena tipos de dados de coleta 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.
- User-defined tipos — Um [tipo definido pelo usuário \(UDT\)](#) requer 3 bytes para metadados, independentemente de seu conteúdo. Para cada elemento UDT, o Amazon Keyspaces exige um byte adicional de metadados.

Para calcular o tamanho codificado de um UDT, comece com o `field name` e o `field value` para os campos de um UDT:

- nome do campo — Cada nome de campo do UDT de nível superior é armazenado usando um identificador. O valor de armazenamento do identificador depende do número geral de campos em seu UDT de nível superior e pode variar entre 1 e 3 bytes:
 - 1—62 campos: 1 byte
 - 63—124 campos: 2 bytes
 - 125— campos máximos: 3 bytes
- valor do campo — Os bytes necessários para armazenar os valores de campo do UDT de nível superior dependem do tipo de dados armazenado:
 - Tipo de dados escalar — Os bytes necessários para armazenamento são os mesmos do mesmo tipo de dados armazenado em uma coluna normal.
 - UDT congelada — Para cada UDT aninhada congelada, a UDT aninhada tem o mesmo tamanho que teria no protocolo binário CQL. Para um UDT aninhado, 4 bytes são armazenados para cada campo (incluindo campos vazios) e o valor do campo armazenado é o formato de serialização do protocolo binário CQL do valor do campo.
 - Coleções Frozen:
 - LIST e SET — Para um congelado aninhado LIST ou SET, 4 bytes são armazenados para cada elemento da coleção mais o formato de serialização do protocolo binário CQL do valor da coleção.
 - MAP — Para um congelado aninhado MAP, cada par de valores-chave tem os seguintes requisitos de armazenamento:
 - Para cada chave, aloque 4 bytes e, em seguida, adicione o formato de serialização do protocolo binário CQL da chave.
 - Para cada valor, aloque 4 bytes e, em seguida, adicione o formato de serialização do protocolo binário CQL do valor.
- Palavra-chave FROZEN — Para coleções congeladas aninhadas em coleções congeladas, o Amazon Keyspaces não exige nenhum byte adicional para metadados.
- Palavra-chave STATIC — os dados da STATIC coluna não contam para o tamanho máximo da linha de 1 MB. Para calcular o tamanho dos dados das colunas estáticas, consulte [the section called “Calcular o tamanho da coluna estática por partição lógica”](#).

Considere o impacto dos recursos do Amazon Keyspaces no tamanho da linha

Esta seção mostra como os recursos no Amazon Keyspaces afetam o tamanho codificado de uma linha.

- Client-side timestamps — os Client-side timestamps são armazenados para cada coluna em cada linha quando o recurso 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 sobre carimbos de data/hora do lado do cliente, consulte [the section called “Client-side carimbos de data/hora”](#)
- Time to Live (TTL) — Os metadados TTL ocupam aproximadamente 8 bytes por linha quando o recurso é ativado. Além disso, os metadados TTL são armazenados para cada coluna de cada linha. Os metadados TTL ocupam aproximadamente 8 bytes para cada coluna que armazena um tipo de dados escalar ou uma coleção congelada. Se a coluna armazenar um tipo de dados de coleta que não esteja congelado, para cada elemento da coleção, o TTL exigirá aproximadamente 8 bytes adicionais para metadados. Para uma coluna que armazena um tipo de dados de coleta quando o TTL está ativado, você pode usar a fórmula a seguir.

```
total encoded size of column = (column id) + sum (nested elements + collection  
metadata (1 byte) + TTL metadata (8 bytes)) + collection column metadata (3 bytes)
```

Os metadados TTL contribuem para o custo de armazenamento e taxa de transferência da linha. Para mais informações sobre TTL, consulte [the section called “Dados expirados com vida útil”](#).

Escolha a fórmula certa para calcular o tamanho codificado de uma linha

Esta seção mostra as diferentes fórmulas que você pode usar para estimar os requisitos de armazenamento ou capacidade de transferência para uma linha de dados no Amazon Keyspaces.

O tamanho total codificado de uma linha de dados pode ser estimado com base em uma das seguintes fórmulas, com base em sua meta:

- Capacidade de processamento — Para estimar o tamanho codificado de uma linha para avaliar as unidades de read/write solicitação (RRUs/WRUs) ou unidades de read/write capacidade () necessárias: RCUs/WCUs

```
total encoded size of row = partition key columns + clustering columns + regular columns
```

- Tamanho do armazenamento — Para estimar o tamanho codificado de uma linha para prever `aBillableTableSizeInBytes`, adicione os metadados necessários para o armazenamento da linha:

```
total encoded size of row = partition key columns + clustering columns + regular columns + row metadata (100 bytes)
```

Important

Todos os metadados da coluna, por exemplo, ids de coluna, metadados de chave de partição, metadados de colunas de agrupamento, bem como carimbos de data/hora do lado do cliente, TTL e metadados de linha, contam para o tamanho máximo de linha de 1 MB.

Exemplo de cálculo do tamanho da linha

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 de todas as colunas. Para estimar o tamanho faturável do armazenamento, 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 sem servidor com a Amazon CloudWatch, consulte [the section called “Monitoramento com CloudWatch”](#)

Estimar o consumo de capacidade do throughput de leitura e gravação no Amazon Keyspaces

Quando você lê ou grava dados no Amazon Keyspaces, a quantidade de unidades de read/write solicitação (RRUs/WRUs) ou unidades de read/write capacidade (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 a 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 “Estimar o tamanho da linha”](#).

Tópicos

- [Estimar o consumo de capacidade de consultas de intervalo no Amazon Keyspaces](#)
- [Estimar o consumo de capacidade de leitura de consultas limitadas](#)
- [Estimar o consumo da capacidade de leitura das varreduras de tabelas](#)
- [Estimar o consumo de capacidade de transações leves no Amazon Keyspaces](#)

- [Estimar o consumo de capacidade para colunas estáticas no Amazon Keyspaces](#)
- [Estimar e provisionar a capacidade para uma tabela multirregional no Amazon Keyspaces](#)
- [Estime o consumo da capacidade de leitura e gravação com a Amazon CloudWatch no Amazon Keyspaces](#)

Estimar o consumo de capacidade de consultas de intervalo no Amazon Keyspaces

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 executada 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` para processar a consulta. No entanto, o Amazon Keyspaces retorna somente a linha em que a condição `WHERE` 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.

Estimar o consumo de capacidade de leitura de consultas limitadas

Ao processar uma consulta que usa a cláusula `LIMIT`, 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 valor `LIMIT` na primeira página, podem ser necessárias uma ou mais chamadas paginadas. Para continuar as leituras na próxima página, você pode usar um token de paginação. O tamanho de página padrão é 1 MB. Para consumir menos capacidade de leitura ao usar cláusulas `LIMIT`, você pode reduzir o tamanho da página. Para obter mais informações sobre paginação, consulte [the section called “Paginar resultados”](#).

Por exemplo, vejamos a seguinte consulta.

```
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 lerá uma linha, desde que você não tenha linhas expiradas com base nas Time-to-live configurações ou nos timestamps do lado do cliente.

O `PAGE SIZE` parâmetro determina quantas linhas o Amazon Keyspaces escaneia do disco para cada solicitação, não quantas linhas o Amazon Keyspaces retorna ao cliente. O Amazon Keyspaces aplica os filtros que você fornece, por exemplo, desigualdade em colunas que não são chave ou `LIMIT` depois de escanear os dados em disco. Se você não definir explicitamente o `PAGE SIZE`, o Amazon Keyspaces lê até 1 MB de dados antes de aplicar os filtros. Por exemplo, se você estiver usando `LIMIT 1` sem especificar o `PAGE SIZE`, o Amazon Keyspaces poderá ler milhares de linhas do disco antes de aplicar a cláusula de limite e retornar somente uma única linha.

Para evitar leitura excessiva, reduza o, o `PAGE SIZE` que reduz o número de linhas que o Amazon Keyspaces escaneia para cada busca. Por exemplo, se você definir `LIMIT 5` em sua consulta, defina um valor entre 5 e 10 `PAGE SIZE` para que o Amazon Keyspaces escaneie somente de 5 a 10 linhas em cada chamada paginada. Você pode modificar esse número para reduzir o número de buscas. Para limites maiores que o tamanho da página, o Amazon Keyspaces mantém a contagem total de resultados com o estado da paginação. No caso `LIMIT` de 10.000 linhas, o Amazon Keyspaces pode buscar esses resultados em duas páginas de 5.000 linhas cada. O limite de 1 MB é o limite superior para qualquer tamanho de página definido.

Estimar o consumo da capacidade de leitura das varreduras de tabelas

As consultas que resultam em varreduras completas da tabela, como por exemplo, consultas usando a opção `ALLOW FILTERING`, 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 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>
```

Por padrão, o Amazon Keyspaces cria uma tabela no modo de capacidade sob demanda com quatro partições. 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 varre 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 varrer as três partições vazias da tabela. Como essa consulta resulta em uma varredura de tabela, o Amazon Keyspaces varre todas as partições na tabela, incluindo partições sem dados.

Estimar o consumo de capacidade de transações leves no Amazon Keyspaces

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 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, o `FALSE` Amazon Keyspaces consome unidades de capacidade de gravação (WCUs) ou unidades de solicitação de gravação (WRUs). O número de WCUs/WRUs consumidos depende do tamanho da linha.

Por exemplo, se o tamanho da linha for 2 KB, a falha na gravação condicional consumirá dois. WCUs/WRUs Se a linha não existir atualmente na tabela, a operação consumirá uma WCUs/WRUs.

Para determinar o número de solicitações que resultaram em falhas na verificação de condições, você pode monitorar a `ConditionalCheckFailed` métrica em CloudWatch.

Estime os custos de LWT para tabelas com Time to Live (TTL)

As LWTs podem exigir unidades de capacidade de leitura (RCUs) ou unidades de solicitação de leitura (RRUs) adicionais para tabelas configuradas com TTL que não usam timestamps do lado do cliente. Ao usar `IF EXISTS` nossos resultados de verificação de condição de `IF NOT EXISTS` palavras-chave `FALSE`, as seguintes unidades de capacidade são consumidas:

- RCUs/RRUs — Se a linha existir, os RCUs/RRUs consumidos serão baseados no tamanho da linha existente.
- RCUs/RRUs — Se a linha não existir, uma única RCU/RRU será consumida.

Se a condição avaliada resultar em uma operação de gravação bem-sucedida, WCUs/WRUs serão consumidas com base no tamanho da nova linha.

Estimar o consumo de capacidade para 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. Para diferenças funcionais com o Apache Cassandra ao usar colunas estáticas e resultados de leitura de intervalos paginados, consulte [the section called “Paginação”](#).

Tópicos

- [Calcular o tamanho da coluna estática por partição lógica no Amazon Keyspaces](#)
- [Estime os requisitos de capacidade e taxa de transferência para read/write operações com dados estáticos no Amazon Keyspaces](#)

Calcular o tamanho da coluna estática 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 “Estimar 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 “Estimar 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 “Estimar 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.

Estime os requisitos de capacidade e taxa de transferência para read/write operações com 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 sob demanda e provisionada. read/write

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:

$2 \times \text{WCUs required for nonstatic data} + 2 \times \text{WCUs required for static data}$

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 ($2 \times 26 \text{ WCUs} + 2 \times 2 \text{ WCUs}$).

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”](#)

Estimar e provisionar a capacidade para uma tabela multirregional no Amazon Keyspaces

Você pode configurar a capacidade de throughput de uma tabela multirregional de uma destas duas maneiras:

- On-demand modo de capacidade, medido em unidades de solicitação de gravação (WRUs)
- Modo de capacidade provisionada com ajuste de escala automático, 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. No modo sob demanda, não é necessário especificar o throughput de leitura e gravação que você espera que sua aplicação execute. O Amazon Keyspaces acomoda instantaneamente o crescimento e a redução das workloads para qualquer nível de tráfego previamente registrado. Se o nível de tráfego de uma workload atingir um novo pico, o Amazon Keyspaces faz adaptações rapidamente para acomodar a workload.

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 throughput de uma tabela multirregional, primeiro você deve 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 (N. da Virgínia), 10 gravações por segundo no Leste dos EUA (Ohio) e 5 gravações por segundo na Europa (Irlanda), você deve prever que a tabela consuma 20 WCUs em cada região: Leste dos EUA (N. da Virgínia), Leste dos EUA (Ohio) e Europa (Irlanda). Isso significa que, neste exemplo, é necessário provisionar 20 WCUs para cada réplica da tabela. Você pode monitorar o consumo de capacidade da sua mesa usando a Amazon CloudWatch. Para obter mais informações, consulte [the section called “Monitoramento com CloudWatch”](#).

Cada gravação é cobrada como 1 WCU, então você veria um total de 60 WCUs cobradas neste exemplo. Para obter mais informações sobre preços, consulte o [Amazon Keyspaces \(para Apache Cassandra\)](#).

Para obter mais informações sobre capacidade provisionada com o ajuste de escala automático do Amazon Keyspaces, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).

Note

Se uma tabela estiver sendo executada no modo de capacidade provisionada com ajuste de escala automático, a capacidade de gravação provisionada poderá flutuar dentro das configurações de ajuste de escala automático de cada região.

Estime o consumo da capacidade de leitura e gravação com a Amazon CloudWatch no Amazon Keyspaces

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. Criar uma 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 opção `ALLOW FILTERING` e verifique as unidades de capacidade de leitura consumidas para a verificação completa da tabela no painel.

Configurar modos de read/write capacidade no Amazon Keyspaces

O Amazon Keyspaces tem dois modos de read/write capacidade para processar leituras e gravações em suas tabelas:

- On-demand (padrão)
- Provisionada

O modo read/write de capacidade escolhido controla como você é cobrado pela taxa de transferência de leitura e gravação e como a capacidade de taxa de transferência da tabela é gerenciada.

Tópicos

- [Configurar o modo de capacidade sob demanda](#)
- [Configurar o modo de capacidade provisionada](#)
- [Exibir o modo de capacidade de uma tabela no Amazon Keyspaces](#)
- [Alterar o modo de capacidade de uma tabela no Amazon Keyspaces](#)
- [Configurar o preaquecimento para tabelas no Amazon Keyspaces](#)

Configurar o 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. Esta opção oferece preços de pagamento por solicitação para solicitações de leitura e gravação, para que você pague apenas 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.

On-demand o modo é uma boa opção se alguma das seguintes afirmações 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 obter 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 compatíveis com o Cassandra”](#).

Tráfego de pico e propriedades de dimensionamento

As tabelas do Amazon Keyspaces que usam o modo de capacidade sob demanda se adaptam automaticamente ao volume de tráfego do seu aplicativo. On-demand o modo de capacidade acomoda instantaneamente até o dobro do pico de tráfego anterior em uma mesa. 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 espere 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, consulte [the section called “Estimar 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.

Configurar o modo de capacidade 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 ajuda você a gerenciar o uso do Amazon Keyspaces para permanecer em ou abaixo de uma taxa de solicitação definida para manter a previsibilidade. Para saber mais sobre escalabilidade automática para throughput provisionada, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).

Modo de capacidade de throughput provisionada é uma boa opção se qualquer uma das declarações a seguir for verdadeira:

- Você tem tráfego de aplicações previsível.
- Você executa aplicativos cujo tráfego é consistente ou aumenta gradualmente.
- Você pode prever os requisitos de capacidade.

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 calcular o consumo da capacidade de leitura e gravação de uma tabela, consulte [the section called “Estimar 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 “Estimar 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 ($4 \text{ KB} \times 6 \text{ RCUs}$).
- Execute leituras `LOCAL_ONE` de até 48 KB por segundo (o dobro do throughput de leitura).
- Grave até 6 KB por segundo ($1 \text{ KB} \times 6 \text{ WCUs}$).

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 Console de gerenciamento da AWS 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](#).

Exibir o modo de capacidade de uma tabela no Amazon Keyspaces

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.

Você também pode usar o AWS CLI para visualizar o modo de capacidade de uma tabela.

Para alterar o throughput provisionado de uma tabela, use [the section called “Alterar modo de capacidade”](#).

Cassandra Query Language (CQL)

Exemplo

```
SELECT * from system_schema.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.

CLI

Visualize o modo de capacidade de taxa de transferência de uma tabela usando o AWS CLI

```
aws keyspaces get-table --keyspace-name myKeyspace --table-name myTable
```

A saída do comando pode ser semelhante à seguinte para uma tabela no modo de capacidade provisionada.

```
"capacitySpecification": {  
  "throughputMode": "PROVISIONED",  
  "readCapacityUnits": 4000,  
  "writeCapacityUnits": 2000  
}
```

A saída de uma tabela no modo sob demanda é semelhante a esta.

```
"capacitySpecification": {  
  "throughputMode": "PAY_PER_REQUEST",  
  "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"  
}
```

Alterar o modo de capacidade de uma tabela no Amazon Keyspaces

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.

Os períodos de espera a seguir se aplicam quando você alterna os modos de capacidade:

- É possível alternar uma tabela recém-criada no modo sob demanda para o modo de capacidade provisionada a qualquer momento. No entanto, só é possível voltar ao modo sob demanda 24 horas após o carimbo de data e hora de criação da tabela.
- É possível alternar uma tabela existente no modo sob demanda para o modo de capacidade provisionada a qualquer momento. No entanto, você pode alternar os modos de capacidade provisionada para sob demanda somente uma vez em um período de 24 horas.

Cassandra Query Language (CQL)

Alterar o modo de capacidade de throughput de uma tabela usando CQL

1. Para alterar o modo de capacidade de uma tabela para PROVISIONED, você precisa configurar as unidades de capacidade de leitura e de gravação com base nos valores de pico esperados da workload. A instrução a seguir é um exemplo disso. Você também pode executar essa instrução para ajustar a capacidade de leitura ou as unidades de capacidade de gravação da tabela.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':  
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 6000,  
'write_capacity_units': 3000}};
```

Para configurar o modo de capacidade provisionada com ajuste de escala automático, consulte [the section called “Configurar o ajuste de escala automático em uma tabela existente”](#).

2. Para alterar o modo de capacidade de uma tabela para o modo sob demanda, defina o modo de throughput como PAY_PER_REQUEST. A instrução a seguir é um exemplo disso.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':  
{'throughput_mode': 'PAY_PER_REQUEST'}};
```

3. Você pode usar a seguinte instrução para confirmar o modo de capacidade da tabela.

```
SELECT * from system_schema.tables where keyspace_name = 'catalog' and  
table_name = 'book_awards';
```

Uma tabela configurada com o modo de capacidade sob demanda retorna o seguinte.

```
{  
  "capacity_mode":{  
    "last_update_to_pay_per_request_timestamp":"1727952499092",  
    "throughput_mode":"PAY_PER_REQUEST"  
  }  
}
```

O valor `last_update_to_pay_per_request_timestamp` é medido em milissegundos.

CLI

Altere o modo de capacidade de taxa de transferência de uma tabela usando o AWS CLI

1. Para alterar o modo de capacidade da tabela para PROVISIONED, você precisa configurar as unidades de capacidade de leitura e de gravação com base nos valores de pico esperados da workload. O comando a seguir é um exemplo disso. Você também pode executar esse comando para ajustar a capacidade de leitura ou as unidades de capacidade de gravação da tabela.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards  
                           \--capacity-specification  
                           throughputMode=PROVISIONED,readCapacityUnits=6000,writeCapacityUnits=3000
```

Para configurar o modo de capacidade provisionada com ajuste de escala automático, consulte [the section called “Configurar o ajuste de escala automático em uma tabela existente”](#).

2. Para alterar o modo de capacidade de uma tabela para o modo sob demanda, você define o modo de throughput como PAY_PER_REQUEST. A instrução a seguir é um exemplo disso.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
                             \--capacity-specification
                             throughputMode=PAY_PER_REQUEST
```

3. Use os comandos a seguir para analisar o modo de capacidade configurado para uma tabela.

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

A saída de uma tabela no modo sob demanda é semelhante a esta.

```
"capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
}
```

Configurar o preaquecimento para tabelas no Amazon Keyspaces

O Amazon Keyspaces escala automaticamente as partições de armazenamento com base na taxa de transferência sob demanda ou provisionada, mas para novas tabelas ou picos repentinos de taxa de transferência, pode levar mais tempo para alocar as partições de armazenamento necessárias. Para garantir que uma mesa nova ou existente tenha capacidade suficiente para suportar o pico de produtividade previsto, você pode definir manualmente valores específicos de taxa de transferência quente para pré-aquecer sua mesa.

A taxa de transferência quente se refere ao número de operações de leitura e gravação que sua tabela Amazon Keyspaces pode suportar instantaneamente. Esses valores estão disponíveis por padrão para todas as tabelas novas e existentes. Se você estiver usando o modo sob demanda ou se atualizar sua taxa de transferência provisionada, o Amazon Keyspaces garante que seu aplicativo seja capaz de emitir solicitações de até esses valores instantaneamente.

O Amazon Keyspaces ajusta automaticamente os valores de taxa de transferência quente à medida que seu uso aumenta. Para ajustar a capacidade de taxa de transferência para eventos de pico futuros, por exemplo, ao migrar dados de outro banco de dados, o que pode exigir o carregamento de terabytes de dados em um curto período de tempo, você pode aumentar manualmente os valores

de taxa de transferência quente de suas tabelas. Isso é útil para eventos de pico planejados em que as taxas de solicitação podem aumentar em 10, 100 vezes ou mais. Primeiro, avalie se a taxa de transferência quente atual é suficiente para lidar com o tráfego esperado. [Então, se precisar pré-aquecer a mesa para o pico de carga de trabalho planejado, você pode aumentar o valor da taxa de transferência quente manualmente sem alterar as configurações de produção ou o modo de capacidade.](#)

Você pode pré-aquecer tabelas para operações de leitura, gravação ou ambas. Você pode aumentar esse valor para tabelas de região única e tabelas multirregionais novas e existentes, e as configurações de taxa de transferência quente definidas se aplicam automaticamente a todas as réplicas das tabelas multirregionais. Não há limite para o número de tabelas do Amazon Keyspaces que você pode pré-aquecer a qualquer momento. O tempo para concluir o pré-aquecimento depende dos valores definidos e do tamanho da tabela. Você pode enviar solicitações simultâneas de pré-aquecimento e essas solicitações não interferem em nenhuma operação da tabela. Você pode pré-aquecer sua mesa até o limite da cota de mesa da sua conta nessa região. Use o [console Service Quotas](#) para verificar suas cotas atuais e aumentá-las, se necessário.

Os valores de taxa de transferência quente que o Amazon Keyspaces ajusta com base no seu uso sob demanda ou na capacidade provisionada estão disponíveis por padrão para todas as tabelas sem custos adicionais. No entanto, se você aumentar manualmente os valores padrão de taxa de transferência de aquecimento para tabelas de pré-aquecimento para eventos de pico de tráfego, cobranças adicionais serão aplicadas. Para obter mais informações, consulte os [preços do Amazon Keyspaces](#).

Aqui estão alguns cenários e melhores práticas diferentes que você pode considerar ao pré-aquecer as tabelas do Amazon Keyspaces.

Throughput a quente e padrões de acesso desiguais

Uma tabela pode ter uma taxa de transferência quente de 30.000 unidades de leitura por segundo e 10.000 unidades de gravação por segundo, mas você ainda pode experimentar eventos de capacidade excedida em leituras ou gravações antes de atingir esses valores. Isso provavelmente se deve a uma partição sobrecarregada. Embora o Amazon Keyspaces possa continuar escalando para suportar uma taxa de transferência praticamente ilimitada, cada partição individual é limitada a 1.000 unidades de gravação por segundo e 3.000 unidades de leitura por segundo. Se seu aplicativo direcionar muito tráfego para uma pequena parte das partições da tabela, eventos de capacidade excedida podem ocorrer mesmo antes de você atingir os valores de taxa de transferência quente da tabela. Recomendamos seguir as [melhores práticas do Amazon Keyspaces](#) para garantir uma escalabilidade perfeita e evitar partições ativas.

Throughput a quente para uma tabela provisionada

Considere uma tabela provisionada que tenha uma taxa de transferência quente de 30.000 unidades de leitura por segundo e 10.000 unidades de gravação por segundo, mas que atualmente tenha uma taxa de transferência provisionada de 4.000 RCUs e 8.000 WCUs. Você pode escalar instantaneamente a taxa de transferência provisionada da tabela até 30.000 RCUs ou 10.000 WCUs atualizando suas configurações de taxa de transferência provisionada. À medida que você aumenta a taxa de transferência provisionada além desses valores, a taxa de transferência quente se ajusta automaticamente aos novos valores mais altos, porque você estabeleceu uma nova taxa de transferência máxima. Por exemplo, se você definir a taxa de transferência provisionada para 50.000 RCU, a taxa de transferência quente aumentará para 50.000 unidades de leitura por segundo.

```
"ProvisionedThroughput":
{
  "ReadCapacityUnits": 4000,
  "WriteCapacityUnits": 8000
}
"WarmThroughput":
{
  "ReadUnitsPerSecond": 30000,
  "WriteUnitsPerSecond": 10000
}
```

Throughput a quente para uma tabela sob demanda

Uma nova tabela sob demanda começa com um throughput a quente de 12 mil unidades de leitura por segundo e 4 mil unidades de gravação por segundo. Sua tabela pode acomodar instantaneamente o tráfego sustentado até esses níveis. Quando suas solicitações excedem 12.000 unidades de leitura por segundo ou 4.000 unidades de gravação por segundo, a taxa de transferência quente se ajusta automaticamente aos valores mais altos.

```
"WarmThroughput":
{
  "ReadUnitsPerSecond": 12000,
  "WriteUnitsPerSecond": 4000
}
```

Melhores práticas para pré-aquecer tabelas do Amazon Keyspaces

Siga estas melhores práticas ao implementar o pré-aquecimento para suas tabelas do Amazon Keyspaces:

Estime com precisão a capacidade necessária

Como o pré-aquecimento tem um custo único, calcule cuidadosamente a taxa de transferência necessária com base na carga de trabalho esperada para evitar o excesso de provisionamento.

Considere o esquema da tabela

Tabelas com linhas maiores podem exigir mais partições para a mesma taxa de transferência. Considere o tamanho médio da linha ao estimar os requisitos de pré-aquecimento.

Monitore o desempenho da mesa

Após o pré-aquecimento, use CloudWatch métricas para verificar se sua tabela está lidando com a carga conforme o esperado. Para obter mais informações, consulte [the section called “Monitore o pré-aquecimento com CloudWatch”](#).

Gerenciar cotas

Se seu aplicativo exigir uma taxa de transferência maior do que a permitida pelas cotas padrão (40.000 RCUs/WCUs ou 2.000 partições), a cota de solicitações aumentará bem antes do evento de alto tráfego. Para solicitar aumento de cota, use o [console do Service Quotas](#).

Otimizar custos

Para eventos temporários de alto tráfego, considere usar o pré-aquecimento em vez de mudar para o modo provisionado com alta capacidade, pois ele pode ser mais econômico para eventos de curta duração. Para obter mais informações sobre preços, consulte os preços [do Amazon Keyspaces](#).

Note

Monitore as métricas de desempenho do seu aplicativo durante a fase de teste para validar se sua configuração de pré-aquecimento suporta adequadamente seus requisitos de carga de trabalho.

Tópicos

- [Criar uma tabela do Amazon Keyspaces com um throughput a quente mais alto](#)
- [Aumentar o throughput a quente da tabela existente do Amazon Keyspaces](#)
- [Veja a taxa de transferência quente de uma tabela do Amazon Keyspaces](#)
- [Monitore o desempenho de uma mesa pré-aquecida usando a Amazon CloudWatch](#)

Criar uma tabela do Amazon Keyspaces com um throughput a quente mais alto

Você pode ajustar os valores de taxa de transferência quente ao criar sua tabela Amazon Keyspaces usando o console, o CQL ou o AWS CLI

Console

Como criar uma nova tabela com configurações de taxa de transferência a quente

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 clustering opcionais.
6. Na seção Configurações da tabela, selecione Personalizar configurações.
7. Continue com as configurações Read/write de capacidade.
8. Para o modo Capacidade, você pode escolher entre um On-demand ou outro Provisionado.
9. Na seção Pre-warming Para tabelas, você pode aumentar os valores de unidades de leitura por segundo e unidades de gravação por segundo conforme necessário para preparar sua tabela para lidar com eventos de pico planejados.

Os valores de taxa de transferência quente que o Amazon Keyspaces ajusta com base no seu uso sob demanda ou na capacidade provisionada estão disponíveis por padrão para todas as tabelas sem custos adicionais. Observe que, se você aumentar manualmente os valores padrão de taxa de transferência de aquecimento para pré-aquecer a tabela para eventos de pico de tráfego, cobranças adicionais serão aplicadas.

10. Configure outros recursos opcionais da tabela conforme necessário. Em seguida, escolha Criar tabela.

Cassandra Query Language (CQL)

- Crie uma tabela com taxa de transferência quente usando um dos seguintes métodos:
 - Para o modo provisionado, crie uma tabela e especifique a capacidade de pico esperada para leituras e gravações usando a seguinte sintaxe CQL:

```
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))  
WITH CUSTOM_PROPERTIES = {  
    'capacity_mode': {  
        'throughput_mode': 'PROVISIONED',  
        'read_capacity_units': 20000,  
        'write_capacity_units': 10000  
    },  
    'warm_throughput': {  
        'read_units_per_second': 40000,  
        'write_units_per_second': 20000  
    }  
};
```

- Para o modo sob demanda, crie uma tabela e especifique a capacidade de pico esperada para leituras e gravações usando a seguinte sintaxe CQL:

```
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))
WITH CUSTOM_PROPERTIES = {
  'capacity_mode': {
    'throughput_mode': 'PAY_PER_REQUEST'
  },
  'warm_throughput': {
    'read_units_per_second': 40000,
    'write_units_per_second': 20000
  }
};
```

Para confirmar as configurações de capacidade da tabela, consulte [the section called “Veja a produtividade quente da sua mesa”](#).

CLI

1. Crie uma tabela com taxa de transferência quente usando um dos métodos a seguir usando o AWS CLI
 - Crie uma nova tabela no modo provisionado e especifique os valores de capacidade de pico esperados para leituras e gravações para ela. A instrução a seguir é um exemplo disso.

```
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=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--capacity-specification
throughputMode=PROVISIONED,readCapacityUnits=20000,writeCapacityUnits=10000
\
--warm-throughput-specification
readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

- Crie uma nova tabela no modo sob demanda e especifique os valores de capacidade de pico esperados para leituras e gravações para a nova tabela. A instrução a seguir é um exemplo disso.

```
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=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--warmThroughputSpecification
readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

2. A saída do comando retorna o ARN da tabela, conforme mostrado no exemplo a seguir.

```
{
  "resourceArn": "arn:aws::cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards>"
}
```

Para confirmar as configurações de capacidade da tabela, consulte [the section called “Veja a produtividade quente da sua mesa”](#).

Java

Crie uma nova tabela usando o SDK for Java.

- Crie uma nova tabela no modo provisionado e especifique os valores de capacidade de pico esperados para leituras e gravações para ela. O exemplo de código a seguir é um exemplo disso.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class PreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Define schema
        List<ColumnDefinition> columns = Arrays.asList(
            ColumnDefinition.builder().name("year").type("int").build(),
```



```

        ColumnDefinition.builder().name("award").type("text").build(),
        ColumnDefinition.builder().name("rank").type("int").build(),
        ColumnDefinition.builder().name("category").type("text").build(),
        ColumnDefinition.builder().name("book_title").type("text").build(),
        ColumnDefinition.builder().name("author").type("text").build(),
        ColumnDefinition.builder().name("publisher").type("text").build()
    );

    List<PartitionKey> partitionKeys = Arrays.asList(
        PartitionKey.builder().name("year").build(),
        PartitionKey.builder().name("award").build()
    );

    List<ClusteringKey> clusteringKeys = Arrays.asList(
        ClusteringKey.builder().name("category").orderBy("ASC").build(),
        ClusteringKey.builder().name("rank").orderBy("ASC").build()
    );

    SchemaDefinition schema = SchemaDefinition.builder()
        .allColumns(columns)
        .partitionKeys(partitionKeys)
        .clusteringKeys(clusteringKeys)
        .build();

    // Define capacity specification
    CapacitySpecification capacitySpec = CapacitySpecification.builder()
        .throughputMode(ThroughputMode.PROVISIONED)
        .readCapacityUnits(20000)
        .writeCapacityUnits(10000)
        .build();

    // Define warm throughput specification
    WarmThroughputSpecification warmThroughput =
    WarmThroughputSpecification.builder()
        .readUnitsPerSecond(40000L)
        .writeUnitsPerSecond(20000L)
        .build();

    // Create table with PreWarming
    CreateTableRequest request = CreateTableRequest.builder()
        .keyspaceName("catalog")
        .tableName("book_awards")
        .schemaDefinition(schema)
        .capacitySpecification(capacitySpec)

```

```
        .warmThroughputSpecification(warmThroughput)
        .build();

    CreateTableResponse response = keyspacesClient.createTable(request);
    System.out.println("Table created with ARN: " + response.resourceArn());
}
}
```

Aumentar o throughput a quente da tabela existente do Amazon Keyspaces

Você pode aumentar os valores atuais de taxa de transferência quente da sua tabela Amazon Keyspaces usando o console, o CQL ou o AWS CLI

Console

Como aumentar as configurações de pré-aquecimento de uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha a tabela que você deseja atualizar.
3. Na guia Capacidade da tabela, continue até Pre-warming quatro tabelas.
4. Na seção Pre-warming Para tabelas, escolha Editar.
5. Na página Editar pré-aquecimento para tabelas, você pode atualizar os valores para unidades de leitura por segundo e para unidades de gravação por segundo.
6. Escolha Salvar alterações. Sua tabela está sendo atualizada com as configurações de pré-aquecimento especificadas.

Cassandra Query Language (CQL)

Aumente as configurações de taxa de transferência quente de uma tabela usando CQL

- Use a ALTER TABLE declaração para aumentar a taxa de aquecimento de uma tabela. A instrução a seguir é um exemplo disso.

```
ALTER TABLE catalog.book_awards
WITH CUSTOM_PROPERTIES = {
    'warm_throughput': {
```

```
        'read_units_per_second': 60000,  
        'write_units_per_second': 30000  
    }  
};
```

Para confirmar as configurações de capacidade atualizadas da tabela, consulte [the section called “Veja a produtividade quente da sua mesa”](#).

CLI

Aumente as configurações de pré-aquecimento de uma mesa usando o AWS CLI

- Para aumentar a taxa de aquecimento da tabela, você pode usar o comando. `update-table` A instrução a seguir é um exemplo disso.

```
aws keyspaces update-table \  
--keyspace-name 'catalog' \  
--table-name 'book_awards' \  
--warmThroughputSpecification readUnitsPerSecond=60000,writeUnitsPerSecond=30000
```

Para confirmar as configurações de capacidade atualizadas da tabela, consulte [the section called “Veja a produtividade quente da sua mesa”](#).

Java

Atualize as configurações de pré-aquecimento de uma tabela usando o SDK for Java.

- Atualize as configurações de taxa de transferência de aquecimento de uma tabela. O exemplo de código a seguir é um exemplo disso.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;  
import software.amazon.awssdk.services.keyspaces.model.*;  
  
public class UpdatePreWarmingExample {  
    public static void main(String[] args) {  
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();  
  
        // Define new warm throughput specification  
        WarmThroughputSpecification warmThroughput =  
            WarmThroughputSpecification.builder()
```

```
        .readUnitsPerSecond(60000L)
        .writeUnitsPerSecond(30000L)
        .build();

// Update table with new PreWarming settings
UpdateTableRequest request = UpdateTableRequest.builder()
    .keyspaceName("catalog")
    .tableName("book_awards")
    .warmThroughputSpecification(warmThroughput)
    .build();

UpdateTableResponse response = keyspacesClient.updateTable(request);
System.out.println("Table update requested: " + response.resourceArn());
    }
}
```

Veja a taxa de transferência quente de uma tabela do Amazon Keyspaces

Você pode visualizar os valores atuais de taxa de transferência quente da tabela do Amazon Keyspaces usando o console, o CQL ou o AWS CLI

Console

Como visualizar as configurações de pré-aquecimento da sua mesa usando o console.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha a tabela que você deseja revisar.
3. Na guia Capacidade da tabela, continue até Pre-warming quatro tabelas.

Cassandra Query Language (CQL)

Visualize as configurações de taxa de transferência de aquecimento de uma tabela usando CQL

- Para visualizar as configurações de taxa de transferência de aquecimento de uma tabela, você pode usar a seguinte instrução CQL.

```
SELECT custom_properties
FROM system_schema_mcs.tables
```

```
WHERE keyspace_name='catalog' and table_name='book_awards';

// Output:
...
custom_properties
-----
{
  'warm_throughput':
  {
    'read_units_per_second': '40000',
    'write_units_per_second': '20000',
    'status': 'AVAILABLE'
  }
}
...
```

CLI

Visualize as configurações de taxa de transferência de aquecimento de uma tabela usando o AWS CLI

- Você pode visualizar as configurações de warm-throughput de uma tabela usando o `get-table` comando, conforme mostrado no exemplo a seguir.

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

Veja a seguir o exemplo de saída do `get-table` comando para uma tabela de região única no modo provisionado.

```
{
  "keyspaceName": "catalog",
  "tableName": "book_awards",
  ... Existing Fields ...,
  "capacitySpecificationSummary": {
    "throughputMode": "PROVISIONED",
    "readCapacityUnits": 20000,
    "writeCapacityUnits": 10000
  },
  "warmThroughputSpecificationSummary": {
```

```
        "readUnitsPerSecond": 40000,  
        "writeUnitsPerSecond": 20000,  
        "status": "AVAILABLE"  
    }  
}
```

Veja a seguir o exemplo de saída de uma tabela de região única no modo sob demanda.

```
{  
  "keyspaceName": "catalog",  
  "tableName": "book_awards_ondemand",  
  ... Existing Fields ...,  
  "capacitySpecification": {  
    "throughputMode": "PAY_PER_REQUEST"  
  },  
  "warmThroughputSpecificationSummary": {  
    "readUnitsPerSecond": 40000,  
    "writeUnitsPerSecond": 20000,  
    "status": "AVAILABLE"  
  }  
}
```

Java

Leia as configurações de pré-aquecimento de uma tabela usando o SDK for Java.

- Leia os valores de taxa de transferência de aquecimento de uma tabela usando `get-table`. O exemplo de código a seguir é um exemplo disso.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;  
import software.amazon.awssdk.services.keyspaces.model.*;  
  
public class GetTableWithPreWarmingExample {  
    public static void main(String[] args) {  
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();  
  
        // Get table details including PreWarming specification  
        GetTableRequest request = GetTableRequest.builder()  
            .keyspaceName("catalog")  
            .tableName("book_awards")  
            .build();
```

```
GetTableResponse response = keyspacesClient.getTable(request);

// Access PreWarming details
if (response.warmThroughputSpecification() != null) {
    WarmThroughputSpecificationSummary warmThroughputSummary =
response.warmThroughputSpecification();
    System.out.println("PreWarming Status: " +
warmThroughputSummary.status());
    System.out.println("Read Units: " +
warmThroughputSummary.readUnitsPerSecond());
    System.out.println("Write Units: " +
warmThroughputSummary.writeUnitsPerSecond());

    // Check if PreWarming is active
    if (warmThroughputSummary.status().equals("AVAILABLE")) {
        System.out.println("Table is fully pre-warmed and ready for high
throughput");
    } else if (warmThroughputSummary.status().equals("UPDATING")) {
        System.out.println("Table PreWarming is currently being
updated");
    }
    } else {
        System.out.println("Table does not have PreWarming enabled");
    }
}
}
```

Monitore o desempenho de uma mesa pré-aquecida usando a Amazon CloudWatch

O pré-aquecimento do Amazon Keyspaces não introduz novas CloudWatch métricas, mas você pode monitorar o desempenho das tabelas pré-aquecidas usando as métricas existentes do Amazon Keyspaces:

SuccessfulRequestLatency

Monitore essa métrica para verificar se a tabela pré-aquecida está processando solicitações com a latência esperada.

WriteThrottleEvents and ReadThrottleEvents

Essas métricas devem permanecer baixas para uma mesa devidamente pré-aquecida. Se você observar erros de capacidade insuficientes apesar do pré-aquecimento, talvez seja necessário ajustar seus valores de taxa de transferência de aquecimento.

ConsumedReadCapacityUnits and ConsumedWriteCapacityUnits

Essas métricas mostram o consumo real da capacidade, o que pode ajudar a validar se sua configuração de pré-aquecimento é apropriada.

ProvisionedReadCapacityUnits and ProvisionedWriteCapacityUnits

Para tabelas provisionadas, essas métricas mostram a capacidade atualmente alocada.

Essas métricas podem ser visualizadas no CloudWatch console ou consultadas usando a CloudWatch API. Para obter mais informações, consulte [the section called “Monitoramento com CloudWatch”](#).

Gerencia automaticamente a capacidade de throughput com o ajuste de escala 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.

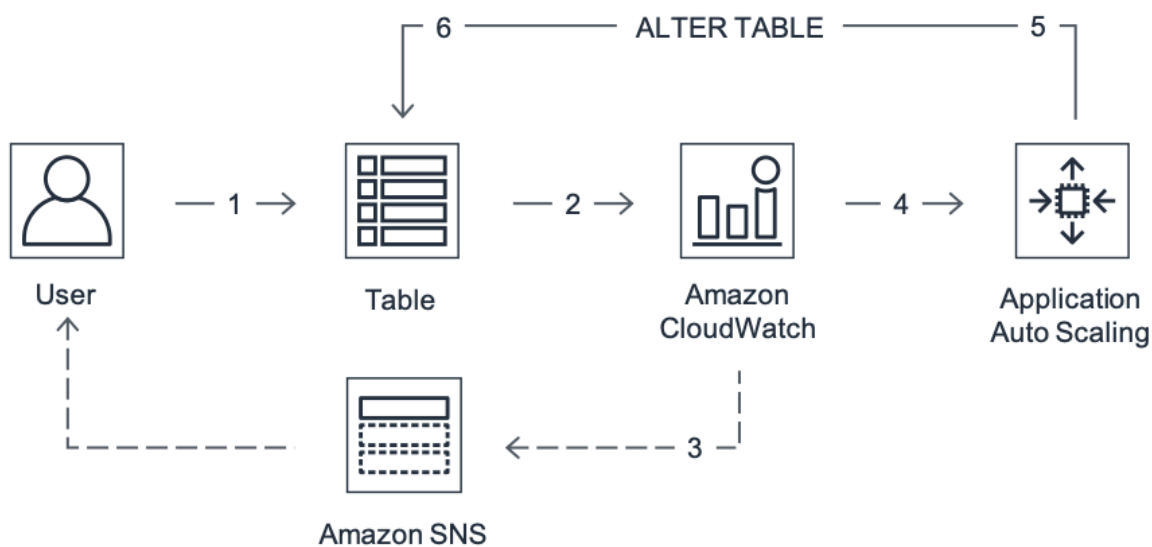
O Amazon Keyspaces (para Apache Cassandra) ajuda você a provisionar a capacidade de throughput de forma eficiente para workloads variáveis, ajustando automaticamente a capacidade de throughput em resposta ao tráfego real do aplicativo. O Amazon Keyspaces usa o serviço Application Auto Scaling para aumentar 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 “Configurar e atualizar políticas de ajuste de escala automático”](#).

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 ajuste de escala. A política de ajuste de escala 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 ajuste de escala 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 workloads 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 ajuste de escala, consulte [Target Tracking Scaling Policies for 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 ajuste de escala automático do Amazon Keyspaces modifica as configurações de throughput provisionado somente quando a workload real permanece elevada (ou baixa) por um período prolongado 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 ajuste de escala 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 ajuste de escala automático, você não pode desativar o ajuste de escala automático para uma única réplica de tabela. Mas você pode ajustar as configurações de ajuste de escala automático de leitura da tabela para diferentes regiões. Por exemplo, você pode especificar diferentes configurações de capacidade de leitura e ajuste de escala automático de leitura para cada região na qual a tabela é replicada.

As configurações de ajuste de escala automático de leitura que você define para uma réplica de tabela em uma região especificada substituem as configurações gerais de ajuste de escala 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 ajuste de escala automático do Amazon Keyspaces atualiza de forma independente a capacidade provisionada da tabela em cada 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 ajuste de escala automático 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 ajuste de escala automático para tabelas multirregionais, consulte [the section called “Atualize as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#).

Note

Se você usa o ajuste de escala automático para tabelas multirregionais, deve sempre usar as operações de API do Amazon Keyspaces para definir as configurações de ajuste de escala automático. 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:

- A escalabilidade automática do Amazon Keyspaces não está disponível na região Oriente Médio (EAU).
- 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 obter mais informações, consulte [the section called “Desativar o ajuste de escala automático do Amazon Keyspaces para uma tabela”](#).
- Se você estiver usando CloudFormation para criar políticas de escalabilidade, você deve gerenciar as políticas de escalabilidade 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 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.

Configure e atualize políticas de ajuste de escala automático do Amazon Keyspaces

Você pode usar o console, o CQL ou o AWS Command Line Interface (AWS CLI) para configurar a escalabilidade automática do Amazon Keyspaces para tabelas novas e existentes. Você também pode modificar as configurações de ajuste de escala automático ou desabilitá-lo.

Para recursos mais avançados, como definir tempos de espera de expansão e redução, recomendamos que você use o CQL ou o para gerenciar as políticas de escalabilidade do AWS CLI Amazon Keyspaces.

Tópicos

- [Configure permissões para ajuste de escala automático do Amazon Keyspaces](#)
- [Crie uma nova tabela com dimensionamento automático](#)
- [Configurar o ajuste de escala automático em uma tabela existente](#)
- [Veja a configuração de ajuste de escala automático do Amazon Keyspaces da sua tabela](#)
- [Desativar o ajuste de escala automático do Amazon Keyspaces para uma tabela](#)
- [Veja a atividade de auto scaling para uma tabela do Amazon Keyspaces na Amazon CloudWatch](#)

Configure permissões para ajuste de escala automático do Amazon Keyspaces

Para começar, confirme se a entidade principal 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. É necessário desativar o ajuste de escala automático de uma tabela antes de excluí-la.

Para configurar um usuário do IAM ou função 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 no Console de gerenciamento da AWS e abra o console do IAM em <https://console.aws.amazon.com/iam/>.
2. No painel do console do IAM, escolha Usuários e, em seguida, escolha seu usuário do IAM ou função na lista.
3. Na página Summary (Resumo), escolha Add permissions (Adicionar permissões).

4. Escolha Anexar políticas existentes diretamente.
5. Na lista de políticas, escolha e AmazonKeyspacesFullAccess, em seguida, escolha Avançar: Revisão.
6. Escolha Adicionar permissões.

Crie uma nova tabela com dimensionamento automático

Ao criar uma nova tabela do Amazon Keyspaces, você pode habilitar automaticamente o ajuste de escala automático para a capacidade de gravação ou leitura da tabela. 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 ajuste de escala automático para réplicas de tabela, consulte [the section called “Criar uma tabela multirregional no modo provisionado”](#).

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 obter mais informações, consulte [the section called “Uso de perfis vinculados ao serviço”](#).

Console

Como criar uma nova tabela com ajuste de escala automático habilitado no console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 clustering opcionais.

6. Na seção Configurações da tabela, selecione Personalizar configurações.
7. Continue com as configurações Read/write de capacidade.
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 configure os valores da capacidade manualmente.
11. Escolha Criar tabela. Sua tabela é criada com os parâmetros padrão de ajuste de escala automático.

Cassandra Query Language (CQL)

Crie uma nova tabela com ajuste de escala automático do Amazon Keyspaces usando CQL

Para definir as configurações de ajuste de escala automático para uma tabela de forma programática, você usa a instrução `AUTOSCALING_SETTINGS` que contém os parâmetros para

ajuste de escala automático do Amazon Keyspaces. Os parâmetros definem as condições que conduzem o Amazon Keyspaces a ajustar o throughput provisionado da tabela e as ações opcionais adicionais que serão executadas. Neste exemplo, você define as configurações de ajuste de escala automático para mytable.

A política contém os elementos a seguir:

- **AUTOSCALING_SETTINGS:** especifica se o Amazon Keyspaces tem permissão para ajustar a capacidade de throughput em seu nome. Os valores a seguir são necessários:
 - **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 ajuste de escala automático do Amazon Keyspaces garante que o índice de capacidade consumida para capacidade provisionada permaneça nesse valor ou próximo a ele. Você define **target_value** como uma porcentagem.
 - **disableScaleIn:** (Opcional) um boolean que especifica se **scale-in** está desativado ou ativado para a tabela. Por padrão, esse parâmetro é desabilitado. Para ativar **scale-in**, defina o valor de boolean 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 espera nas atividades de aumento da escala horizontalmente, especifique um valor, em segundos, para **scale_out_cooldown**. Se você não especificar um valor, o valor padrão é 0. Para obter mais informações sobre rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) 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 espera nas atividades de redução da escala horizontalmente, especifique um valor, em segundos, para **scale_in_cooldown**. Se você não especificar um valor, o valor padrão é 0. Para obter

mais informações sobre rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) 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 taxa consumida-para-provisionada agora é (150/200), ou 75%. Essa taxa excede o valor pretendido. Portanto, o ajuste de escala automático aumenta a capacidade de gravação provisionada para 215 de modo que a taxa seja (150/215) ou 69,77%, tão próxima ao seu `target_value` quanto possível, mas sem excedê-lo.

Para `mytable`, você define `TargetValue` para a capacidade de leitura e gravação em 50 por cento. O ajuste de escala automático do Amazon Keyspaces ajusta o throughput provisionado da tabela dentro do intervalo de 5 a 10 unidades de capacidade para que a taxa consumida-para-provisionada permaneça em 50% ou próxima a isso. Para capacidade de leitura, você define os valores de `ScaleOutCooldown` e `ScaleInCooldown` para 60 segundos.

Você pode usar a seguinte declaração para criar uma nova tabela do Amazon Keyspaces com o ajuste de escala automático 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
        }
      }
    }
  }
};
```

CLI

Crie uma nova tabela com a escalabilidade automática do Amazon Keyspaces usando o AWS CLI

Para definir as configurações de ajuste de escala automático para uma tabela de forma programática, você usa a ação `autoScalingSpecification` que define os parâmetros para o ajuste de escala automático do Amazon Keyspaces. Os parâmetros definem as condições que conduzem o Amazon Keyspaces a ajustar o throughput provisionado da tabela e as ações opcionais adicionais que serão executadas. Neste exemplo, você define as configurações de ajuste de escala automático para `mytable`.

A política contém os elementos a seguir:

- `autoScalingSpecification`: especifica se o Amazon Keyspaces tem permissão para ajustar a capacidade de throughput em seu nome. Você pode ativar o ajuste de escala automático para capacidade de leitura e gravação separadamente. Então, você deve especificar os 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 ajuste de escala automático do Amazon Keyspaces garante que o índice de capacidade consumida para capacidade provisionada permaneça nesse valor ou próximo a ele. Você define **targetValue** como uma porcentagem.
- **disableScaleIn**: (Opcional) um boolean que especifica se **scale-in** está desativado ou ativado para a tabela. Por padrão, esse parâmetro é desabilitado. Para ativar **scale-in**, defina o valor de boolean 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 espera nas atividades de aumento da escala horizontalmente, especifique um valor, em segundos, para **ScaleOutCooldown**. O valor padrão é 0. Para obter mais informações sobre rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) 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 espera nas atividades de redução da escala horizontalmente, especifique um valor, em segundos, para **ScaleInCooldown**. O valor padrão é 0. Para obter mais informações sobre rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) 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 taxa consumida-para-provisionada agora é $(150/200)$, ou 75%. Essa taxa excede o valor pretendido. Portanto, o ajuste de escala automático aumenta a capacidade de gravação provisionada para 215 de modo que a taxa seja $(150/215)$ ou 69,77%, tão próxima ao seu **TargetValue** quanto possível, mas sem excedê-lo.

Para mytable, você define TargetValue para a capacidade de leitura e gravação em 50 por cento. O ajuste de escala automático do Amazon Keyspaces ajusta o throughput provisionado da tabela dentro do intervalo de 5 a 10 unidades de capacidade para que a taxa consumida-para-provisionada permaneça em 50% ou próxima a isso. Para capacidade de leitura, você define os valores de ScaleOutCooldown e ScaleInCooldown para 60 segundos.

Ao criar tabelas com configurações de ajuste de escala automático complexas, é útil carregar as configurações de ajuste de escala automático a partir de um arquivo JSON. No 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
```

Configurar o ajuste de escala automático em uma tabela existente

Você pode atualizar uma tabela do Amazon Keyspaces existente para ativar o ajuste de escala automático para a capacidade de gravação ou leitura da tabela. Se você estiver atualizando uma tabela que está atualmente no modo de capacidade sob demanda, primeiro precisará alterar o modo de capacidade da tabela para o modo de capacidade provisionada.

Para obter mais informações sobre como atualizar as configurações de ajuste de escala automático para uma tabela multirregional, consulte [the section called “Atualize as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#).

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 obter mais informações, consulte [the section called “Uso de perfis vinculados ao serviço”](#).

Console

Configure o ajuste de escala automático do Amazon Keyspaces para uma tabela existente

1. Faça login no Console de gerenciamento da AWS 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 para 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 [the section called “Crie uma nova tabela com dimensionamento automático”](#) para editar a capacidade de leitura e gravação.
6. Quando as configurações de escala automática estiverem definidas, escolha Salvar.

Cassandra Query Language (CQL)

Configure uma tabela existente com o ajuste de escala automático do Amazon Keyspaces usando CQL

Você pode usar o comando `ALTER TABLE` para uma tabela do Amazon Keyspaces existente para configurar o ajuste de escala automático para a capacidade de gravação ou leitura da tabela. Se você estiver atualizando uma tabela que está atualmente no modo de capacidade sob demanda, precisará configurar `capacity_mode` como provisionada. Se sua tabela já estiver no modo de capacidade provisionada, esse campo poderá ser omitido.

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 ajuste de escala automático ativado.

A capacidade de gravação é configurada no intervalo de 5 a 10 unidades de capacidade com um valor de destino de 50%. A capacidade de leitura também é configurada no intervalo de 5 a 10 unidades de capacidade com um valor de destino de 50%. Para capacidade de leitura, você define os valores de `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
            }
        }
    }
}
};

```

CLI

Configure uma tabela existente com a escalabilidade automática do Amazon Keyspaces usando o AWS CLI

Para uma tabela do Amazon Keyspaces existente, você pode ativar o ajuste de escala automático para a capacidade de gravação ou leitura da tabela usando a operação `UpdateTable`.

Você pode usar o seguinte comando para habilitar o ajuste de escala automático do Amazon Keyspaces para tabelas existentes. As configurações de ajuste de escala automático da tabela são carregadas de um arquivo JSON. No 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 ajuste de escala automático usadas no exemplo a seguir, consulte [the section called “Crie uma nova tabela com dimensionamento automático”](#).

```
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
```

Veja a configuração de ajuste de escala automático do Amazon Keyspaces da sua tabela

Você pode usar o console, o CQL ou o AWS CLI para visualizar e atualizar as configurações de escalabilidade automática de uma tabela do Amazon Keyspaces.

Console

Visualização das configurações de ajuste de escala automático usando o console

1. Escolha a tabela que você deseja visualizar e vá para a guia Capacidade.
2. Na seção Configurações de capacidade, escolha Editar. Você pode agora modificar as configurações nas seções Capacidade de leitura ou Capacidade de gravação. Para ter mais informações sobre essas configurações, consulte [the section called “Crie uma nova tabela com dimensionamento automático”](#).

Cassandra Query Language (CQL)

Veja a política de ajuste de escala automático do Amazon Keyspaces da sua tabela usando CQL

Para visualizar os detalhes da configuração do ajuste de escala 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}}}

```

CLI

Veja a política de escalabilidade automática do Amazon Keyspaces da sua tabela usando o AWS CLI

Para visualizar a configuração de ajuste de escala automático de uma tabela, você pode usar a operação `get-table-auto-scaling-settings`. 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-1:111122223333:/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
        }
    }
}
}
}
}
}

```

Desativar o ajuste de escala automático do Amazon Keyspaces para uma tabela

Você pode desativar o ajuste de escala 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 desabilitar o ajuste de escala automático para que o Amazon Keyspaces não continue modificando as configurações de capacidade de leitura ou gravação da sua tabela. É possível atualizar a tabela usando o console, o CQL ou a AWS CLI.

Desativar o escalonamento automático também exclui os CloudWatch alarmes que foram criados em seu nome.

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 usada pelo Application Auto Scaling, você deve desativar o ajuste de escala automático em todas as tabelas da conta em todas as Regiões da AWS.

Console

Desative o ajuste de escala automático do Amazon Keyspaces para sua tabela usando o console

Uso do console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. Escolha a tabela que você deseja atualizar e vá para a guia Capacidade.
3. Na seção Configurações de capacidade, escolha Editar.
4. Para desativar o ajuste de escala automático do Amazon Keyspaces, desmarque a opção Escalar automaticamente. A desativação do ajuste de escala automático cancela o registro da tabela como um destino escalável com o Application Auto Scaling.

Cassandra Query Language (CQL)

Desative o ajuste de escala automático do Amazon Keyspaces para sua tabela usando o CQL

A instrução a seguir desativa o ajuste de escala automático para a capacidade de gravação da tabela mytable.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

CLI

Desative a escalabilidade automática do Amazon Keyspaces para sua tabela usando o AWS CLI

O comando a seguir desativa o ajuste de escala 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}
```

Veja a atividade de auto scaling para uma tabela do Amazon Keyspaces na Amazon CloudWatch

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.

Use a capacidade de expansão 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 trabalhar com os recursos do Amazon Keyspaces (para Apache Cassandra)

Este capítulo fornece detalhes sobre como trabalhar com o Amazon Keyspaces e vários recursos de banco de dados, por exemplo, backup e restauração, Time to Live e replicação multirregional.

- **Vida útil (TTL):** o Amazon Keyspaces expira os dados das tabelas automaticamente com base no valor Vida útil que você definiu. Saiba como configurar a TTL e como usá-la em suas tabelas.
- **PITR:** proteja as tabelas do Amazon Keyspaces contra operações acidentais de gravação ou exclusão criando backups contínuos dos dados da tabela. Saiba como configurar a PITR em suas tabelas e como restaurar uma tabela em um momento específico ou quando ela foi excluída acidentalmente.
- **Trabalhando com tabelas multirregionais** — as Multi-Region tabelas no Amazon Keyspaces devem ter capacidade de taxa de transferência de gravação configurada no modo de capacidade sob demanda ou provisionada com escalabilidade automática. Planeje as necessidades de capacidade de throughput estimando as unidades de capacidade de gravação (WCUs) necessárias para cada região e provisione a soma das gravações de todas as regiões para garantir capacidade suficiente para gravações replicadas.
- **Captura de dados de alterações do Amazon Keyspaces (CDC)** — Os streams CDC do Amazon Keyspaces registram eventos de alteração em nível de linha da sua tabela quase em tempo real. Saiba como usar a Kinesis Client Library (KCL) para consumir e processar dados dos streams CDC do Amazon Keyspaces.
- **Consultas e paginação:** o Amazon Keyspaces oferece suporte a recursos avançados de consulta, como utilizar o operador `IN` com declarações `SELECT`, ordenar resultados com `ORDER BY` e paginação automática de conjuntos volumosos de resultados. Esta seção explica como o Amazon Keyspaces processa essas consultas e fornece exemplos.
- **Particionadores:** o Amazon Keyspaces fornece três particionadores: `Murmur3Partitioner` (padrão), `RandomPartitioner` e `DefaultPartitioner`. Você pode alterar o particionador por região no nível da conta usando o Cassandra Query Language (CQL) Console de gerenciamento da AWS ou o Cassandra Query Language.
- **Client-side timestamps** — timestamps são Client-side timestamps Cassandra-compatible que o Amazon Keyspaces persiste para cada célula em sua tabela. Use carimbos de data/hora do lado do cliente para resolução de conflitos, permitindo que seu aplicativo cliente determine a ordem das gravações.

- **User-defined tipos (UDTs)** — Com os UDTs, você pode definir estruturas de dados em seus aplicativos que representam hierarquias de dados do mundo real.
- **Recursos de marcação:** você pode rotular recursos do Amazon Keyspaces, como espaços de chave e tabelas, usando tags. As tags ajudam a categorizar recursos, permitem o controle de custos e que você configure o controle de acesso com base nas tags. Esta seção aborda restrições de marcação, operações e melhores práticas para o Amazon Keyspaces.
- **CloudFormation modelos** — CloudFormation ajudam você a modelar e configurar seus espaços de chave e tabelas do Amazon Keyspaces para que você possa gastar menos tempo criando e gerenciando seus recursos e infraestrutura.

Tópicos

- [Espaços de chaves do sistema no Amazon Keyspaces](#)
- [User-defined tipos \(UDTs\) no Amazon Keyspaces](#)
- [Como trabalhar com consultas CQL no Amazon Keyspaces](#)
- [Trabalhando com fluxos de captura de dados de alteração \(CDC\) no Amazon Keyspaces](#)
- [Como trabalhar com particionadores no Amazon Keyspaces](#)
- [Client-side timestamps no Amazon Keyspaces](#)
- [Multi-Region replicação para Amazon Keyspaces \(para Apache Cassandra\)](#)
- [Backup e restauração de dados com a recuperação para um ponto no tempo para o Amazon Keyspaces](#)
- [Dados expirados com vida útil \(TTL\) para Amazon Keyspaces \(para Apache Cassandra\)](#)
- [Usando esse serviço com um AWS SDK](#)
- [Como trabalhar com tags e rótulos para recursos do Amazon Keyspaces](#)
- [Crie recursos do Amazon Keyspaces com AWS CloudFormation](#)
- [Uso do NoSQL Workbench com o Amazon Keyspaces \(para Apache Cassandra\)](#)

Espaços de chaves do sistema no Amazon Keyspaces

Espaços-chave do sistema e tabelas do sistema no Amazon Keyspaces (para Apache Cassandra) são recursos somente para leitura que armazenam metadados sobre seus recursos do Amazon Keyspaces. Os espaços de chave do sistema estão presentes em todos Conta da AWS, independentemente de você ter criado algum espaço-chave ou tabela. Eles são um recurso de compatibilidade com o Apache Cassandra e são fornecidos sem custo adicional.

Você não pode modificar ou excluir espaços de teclas do sistema. O console do Amazon Keyspaces exibe somente espaços de chave criados pelo usuário. Os espaços de teclas do sistema são acessíveis programaticamente por meio do CQL e aparecem em serviços como e. AWS CloudFormation AWS Config

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 chaves do sistema e as tabelas do sistema 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
<code>local</code>	<code>key</code> , <code>bootstrap</code> , <code>ped</code> , <code>broadcast</code> , <code>_address</code> , <code>cluster_name</code> , <code>cql_version</code> , <code>data_center</code> , <code>gossip_generation</code> , <code>host_id</code> , <code>listen_address</code> , <code>native_protocol_version</code> , <code>partitioner</code> , <code>rack</code> , <code>release_version</code> , <code>rpc_address</code> , <code>schema_version</code> , <code>thrift_version</code> , <code>tokens</code> , <code>truncated_at</code>	Informações sobre o espaço de chaves local.

Nomes das tabelas	Nomes de colunas	Comentários
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, verá a lista de endereços IP que você configurou. Para obter 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 informações 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_filter_fp_chance, caching, comment, compaction, compression, crc_check_chance, dclocal_read_repair_chance, default_time_to_live, extensions, flags, gc_grace_seconds, id, max_index_interval, memtable_flush_period_in_ms, min_index_interval, read_repair_chance, speculative_retry	Informações sobre uma tabela específica.
types	keyspace_name, type_name, field_names, field_types	Informações sobre um tipo específico definido pelo usuário (UDT).
columns	keyspace_name, table_name, column_name, clustering_order, column_name	Informações sobre uma coluna específica.

Nomes das tabelas	Nomes de colunas	Comentários
	me_bytes, kind, position, type	

system_schema_mcs

Esse é um keyspace do Amazon Keyspaces que armazena informações 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 obter mais informações, consulte the section called “Verifique o status da criação do keyspace” .
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, extension	Consulte essa tabela para descobrir o status de uma tabela específica. Para obter mais informações, consulte the section called “Verifique o status da criação da tabela”. Você também pode consultar essa tabela para listar as configurações específicas do Amazon Keyspaces que são armazenadas como <code>custom_properties</code>. Por exemplo: • <code>capacity_mode</code>

Nomes das tabelas	Nomes de colunas	Comentários
	s, compaction, comment, id, compression, memtable_flush_period_in_ms, cdc_specification, latest_stream_arn status	• cdc_specification • client_side_timestamps • encryption_specification • point_in_time_recovery • 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.
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 obter mais informações, consulte the section called “Exibir tags da tabela” .

Nomes das tabelas	Nomes de colunas	Comentários
types	keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types	Consulte essa tabela para descobrir informações sobre tipos definidos pelo usuário (UDTs). Por exemplo, você pode consultar essa tabela para listar todos os UDTs de um determinado espaço de chave. Para obter mais informações, consulte the section called “User-defined tipos (UDTs)” .
autoscaling	keyspace_name, table_name, provisioned_read_capacity, autoscaling_update, provisioned_write_capacity, autoscaling_update	Consulte essa tabela para obter as configurações de ajuste de escala 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 obter mais informações, consulte the section called “Veja a configuração de ajuste de escala automático do Amazon Keyspaces da sua tabela” .

system_multiregion_info

Esse é um keyspace 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 informações sobre tabelas multirregionais — 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 <code>custom_properties</code>. Por exemplo: • <code>capacity_mode</code> Para consultar essa tabela, você precisa especificar <code>keyspace_name</code> e <code>table_name</code> na cláusula <code>WHERE</code>. Para obter mais informações, consulte the section called “Criar um espaço de chaves multirregional”.
keyspaces	keyspace_name, region, status, tables_replication _progress	Essa tabela contém informações sobre o progresso de uma <code>ALTER KEYSPACE</code> operação que adiciona uma réplica a um keyspace — por exemplo, quantas tabelas já foram criadas na nova região e quantas tabelas ainda estão

Nomes das tabelas	Nomes de colunas	Comentários
		em andamento. Para obter exemplos, consulte the section called “Verifique o progresso da replicação” .
autoscaling	keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update, region	Consulte essa tabela para obter as configurações de ajuste de escala automático de uma tabela provisionada multirregional. Observe que essas configurações não estarão disponíveis até que a tabela esteja ativa. Para consultar essa tabela, você precisa especificar <code>keyspace_name</code> e <code>table_name</code> na cláusula <code>WHERE</code> . Para obter mais informações, consulte the section called “Atualize as configurações de capacidade e provisionada e ajuste de escala automático para uma tabela multirregional” .

Nomes das tabelas	Nomes de colunas	Comentários
types	keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types, region	Consulte esta tabela para descobrir informações sobre tipos definidos pelo usuário (UDTs) em espaços de chave multirregionais. Por exemplo, você pode consultar essa tabela para listar todas as réplicas da tabela e suas respectivas AWS regiões que usam UDTs para um determinado espaço de chave. Para obter mais informações, consulte the section called “User-defined tipos (UDTs)” .

User-defined tipos (UDTs) no Amazon Keyspaces

Um tipo definido pelo usuário (UDT) é um agrupamento de campos e tipos de dados que você pode usar para definir uma única coluna no Amazon Keyspaces. Os tipos de dados válidos para UDTs são todos os tipos de dados do Cassandra compatíveis, incluindo coleções e outros UDTs que você já criou no mesmo espaço de chaves. Para obter mais informações sobre os tipos de dados do Cassandra compatíveis, consulte [the section called “Suporte ao tipo de dados do Cassandra”](#).

Você pode usar tipos definidos pelo usuário (UDTs) no Amazon Keyspaces para organizar os dados de forma mais eficiente. Por exemplo, você pode criar UDTs com coleções aninhadas, o que permite implementar uma modelagem de dados mais complexa em seus aplicativos. Você também pode usar a palavra-chave frozen para definir UDTs.

Os UDTs estão vinculados a um espaço de chave e estão disponíveis para todas as tabelas e UDTs no mesmo espaço de chave. Você pode criar UDTs em espaços de chave de região única e multirregião.

Você pode criar novas tabelas ou alterar tabelas existentes e adicionar novas colunas que usam um UDT. Para criar uma UDT com uma UDT aninhada, a UDT aninhada precisa ser congelada.

Para analisar quantos UDTs são suportados por espaço de chave, níveis de aninhamento suportados e outros valores e cotas padrão relacionados aos UDTs, consulte [the section called “Cotas e valores padrão para tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces”](#)

Para obter informações sobre como calcular o tamanho codificado dos UDTs, consulte [the section called “Estime o tamanho codificado dos valores de dados com base no tipo de dados”](#)

Para obter mais informações sobre sintaxe de CQL, consulte [the section called “Tipos”](#).

Para saber mais sobre UDTs e restauração pontual, consulte [the section called “PITR e UDTs”](#)

Tópicos

- [Configure permissões para trabalhar com tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces](#)
- [Crie um tipo definido pelo usuário \(UDT\) no Amazon Keyspaces](#)
- [Visualize tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces](#)
- [Excluir um tipo definido pelo usuário \(UDT\) no Amazon Keyspaces](#)

Configure permissões para trabalhar com tipos definidos pelo usuário (UDTs) no Amazon Keyspaces

Assim como as tabelas, as UDTs estão vinculadas a um espaço de teclas específico. Mas, diferentemente das tabelas, você não pode definir permissões diretamente para UDTs. Os UDTs não são considerados recursos AWS e não têm identificadores exclusivos no formato de um Amazon Resource Name (ARN). Em vez disso, para dar a um IAM permissões principais para realizar ações específicas em um UDT, você precisa definir permissões para o keyspace ao qual o UDT está vinculado. Para trabalhar com UDTs em espaços de chave multirregionais, são necessárias permissões adicionais.

Para poder criar, visualizar ou excluir UDTs, o principal, por exemplo, um usuário ou função do IAM, precisa das mesmas permissões necessárias para realizar a mesma ação no espaço de chaves ao qual o UDT está vinculado.

Para obter mais informações sobre AWS Identity and Access Management, consulte [the section called “AWS Identity and Access Management”](#).

Permissões para criar um UDT

Para criar um UDT em um espaço de chave de região única, o principal precisa de Create permissões para o espaço de chave.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Create",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Para criar um UDT em um espaço de chave multirregional, além das Create permissões, o diretor também precisa de permissões para a ação no espaço de chave CreateMultiRegionResource especificado.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "cassandra:Create", "cassandra:CreateMultiRegionResource" ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```


Permissões para visualizar um UDT

Para visualizar ou listar UDTs em um espaço de chave de região única, o principal precisa de permissões de leitura para o espaço de chaves do sistema. Para obter mais informações, consulte [the section called “system_schema_mcs”](#).

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Select",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

Para visualizar ou listar UDTs para um espaço de chave multirregional, o principal precisa de permissões para as ações SELECT e SelectMultiRegionResource para o espaço de teclas do sistema. Para obter mais informações, consulte [the section called “system_multiregion_info”](#).

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["cassandra:Select", "cassandra:SelectMultiRegionResource"],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

Permissões para excluir um UDT

Para excluir um UDT de um espaço de chave de região única, o principal precisa de permissões para a Drop ação no espaço de chave especificado.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Drop",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Para excluir um UDT de um espaço de chave multirregional, o principal precisa de permissões para a Drop ação e para a DropMultiRegionResource ação do espaço de chave especificado.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "cassandra:Drop", "cassandra:DropMultiRegionResource" ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Crie um tipo definido pelo usuário (UDT) no Amazon Keyspaces

Para criar um UDT em um keyspace de região única, você pode usar a CREATE TYPE instrução na CQL, o create-type comando com o ou o console AWS CLI.

Os nomes do UDT devem conter 48 caracteres ou menos, devem começar com um caractere alfabético e só podem conter caracteres alfanuméricos e sublinhados. O Amazon Keyspaces converte automaticamente caracteres maiúsculos em caracteres minúsculos.

Como alternativa, você pode declarar um nome UDT entre aspas duplas. Ao declarar um nome UDT entre aspas duplas, o Amazon Keyspaces preserva maiúsculas e permite caracteres especiais.

Você também pode usar aspas duplas como parte do nome ao criar o UDT, mas deve escapar de cada caractere de aspas duplas com um caractere adicional de aspas duplas.

A tabela a seguir mostra exemplos de nomes de UDT permitidos. A primeira coluna mostra como inserir o nome ao criar o tipo, a segunda coluna mostra como o Amazon Keyspaces formata o nome internamente. O Amazon Keyspaces espera o nome formatado para operações como. `GetType`

Nome inserido	Nome formatado	Observação
MY_UDT	my_udt	Sem aspas duplas, o Amazon Keyspaces converte todos os caracteres maiúsculos em minúsculas.
"MY_UDT"	MY_UDT	Com aspas duplas, o Amazon Keyspaces respeita os caracteres maiúsculos e remove as aspas duplas do nome formatado.
"1234"	1234	Com aspas duplas, o nome pode começar com um número, e o Amazon Keyspaces remove as aspas duplas do nome formatado.
"Special_Ch@r@cter s<>!!"	Special_Ch@r@cter s<>!!	Com aspas duplas, o nome pode conter caracteres especiais, e o Amazon Keyspaces remove as aspas duplas do nome formatado.
"nested""""quote s"	nested""""quotes	O Amazon Keyspaces remove as aspas duplas externas e as aspas duplas de escape do nome formatado.

Console

Crie um tipo definido pelo usuário (UDT) com o console Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces e, em seguida, escolha um keyspace na lista.
3. Escolha a guia UDTs.
4. Escolha Criar UDT
5. Em Detalhes do UDT, insira o nome do UDT. Nos campos do UDT, você define o esquema do UDT.
6. Para finalizar, escolha Criar UDT.

Cassandra Query Language (CQL)

Crie um tipo definido pelo usuário (UDT) com CQL

Neste exemplo, criamos uma nova versão da tabela de prêmios de livros usada em [the section called “Criar uma tabela”](#). Nesta tabela, armazenamos todos os prêmios que um autor recebe por um determinado livro. Criamos dois UDTs que estão aninhados e contêm informações sobre o livro que recebeu um prêmio.

1. Crie um keyspace com o nome `catalog`.

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

2. Crie o primeiro tipo. Esse tipo armazena códigos BISAC, que são usados para definir o gênero dos livros. Um código BISAC consiste em um código alfanumérico e até quatro áreas temáticas.

```
CREATE TYPE catalog.bisac (  
    bisac_code text,  
    subject1 text,  
    subject2 text,  
    subject3 text,  
    subject4 text  
);
```

3. Crie um segundo tipo para prêmios de livros que use o primeiro UDT. O UDT aninhado deve ser congelado.

```
CREATE TYPE catalog.book (
    award_title text,
    book_title text,
    publication_date date,
    page_count int,
    ISBN text,
    genre FROZEN <bisac>
);
```

4. Crie uma tabela com uma coluna para o nome do autor e use um tipo de lista para os prêmios do livro. Observe que o UDT usado na lista precisa ser congelado.

```
CREATE TABLE catalog.authors (
    author_name text PRIMARY KEY,
    awards list <FROZEN <book>>
);
```

5. Nesta etapa, inserimos uma linha de dados na nova tabela.

```
CONSISTENCY LOCAL_QUORUM;
```

```
INSERT INTO catalog.authors (author_name, awards) VALUES (
    'John Stiles' ,
    [{
        award_title: 'Wolf',
        book_title: 'Yesterday',
        publication_date: '2020-10-10',
        page_count: 345,
        ISBN: '026204630X',
        genre: { bisac_code: 'FIC014090', subject1: 'FICTION', subject2:
        'Historical', subject3: '20th Century', subject4: 'Post-World War II' }
    },
    {award_title: 'Richard Roe',
    book_title: 'Who ate the cake?',
    publication_date: '2019-05-13',
    page_count: 193,
    ISBN: '9780262046305',
    genre: { bisac_code: 'FIC022130', subject1: 'FICTION', subject2: 'Mystery &
    Detective', subject3: 'Cozy', subject4: 'Culinary' }
```

```
    ]]  
);
```

- Na última etapa, lemos os dados da tabela.

```
SELECT * FROM catalog.authors;
```

A saída do comando deve ter essa aparência.

```
author_name | awards  
-----  
+-----  
John Stiles | [{award_title: 'Wolf', book_title: 'Yesterday', publication_date:  
2020-10-10, page_count: 345, isbn: '026204630X', genre: {bisac_code:  
'FIC014090', subject1: 'FICTION', subject2: 'Historical', subject3: '20th  
Century', subject4: 'Post-World War II'}}, {award_title: 'Richard Roe',  
book_title: 'Who ate the cake?', publication_date: 2019-05-13, page_count: 193,  
isbn: '9780262046305', genre: {bisac_code: 'FIC022130', subject1: 'FICTION',  
subject2: 'Mystery & Detective', subject3: 'Cozy', subject4: 'Culinary'}}]  
  
(1 rows)
```

Para obter mais informações sobre sintaxe de CQL, consulte [the section called “CREATE TYPE”](#).

CLI

Crie um tipo definido pelo usuário (UDT) com o AWS CLI

- Para criar um tipo, você pode usar a seguinte sintaxe.

```
aws keyspaces create-type  
--keyspace-name 'my_keyspace'  
--type-name 'my_udt'  
--field-definitions  
  '['  
    {"name" : "field1", "type" : "int"},  
    {"name" : "field2", "type" : "text"}  
  ]'
```

2. A saída desse comando é semelhante a este exemplo. Observe que `typeName` retorna o nome formatado do UDT.

```
{
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "typeName": "my_udt"
}
```

Visualize tipos definidos pelo usuário (UDTs) no Amazon Keyspaces

Para visualizar ou listar todos os UDTs em um espaço de chave de região única, você pode consultar a tabela `system_schema_mcs.types` no espaço de chaves do sistema usando uma instrução em CQL ou usar os `list-type` comandos `get-type` e com o AWS CLI console.

Para qualquer uma das opções, o diretor do IAM precisa de permissões de leitura no espaço de chaves do sistema. Para obter mais informações, consulte [the section called “Configurar permissões do”](#).

Console

Visualize tipos definidos pelo usuário (UDT) com o console Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces e, em seguida, escolha um keyspace na lista.
3. Escolha a guia UDTs para revisar a lista de todos os UDTs no keyspace.
4. Para revisar um UDT em detalhes, escolha um UDT na lista.
5. Na guia Esquema, você pode revisar o esquema. Na guia Usado em, você pode ver se esse UDT é usado em tabelas ou outros UDTs. Observe que você só pode excluir UDTs que não estejam em uso por tabelas ou outros UDTs.

Cassandra Query Language (CQL)

Visualize os tipos definidos pelo usuário (UDTs) de um keyspace de região única com CQL

1. Para ver os tipos que estão disponíveis em um determinado espaço de teclas, você pode usar a seguinte declaração.

```
SELECT type_name
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace';
```

2. Para ver os detalhes sobre um tipo específico, você pode usar a seguinte declaração.

```
SELECT
    keyspace_name,
    type_name,
    field_names,
    field_types,
    max_nesting_depth,
    last_modified_timestamp,
    status,
    direct_referring_tables,
    direct_parent_types
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace' AND type_name = 'my_udt';
```

3. Você pode listar todos os UDTs que existem na conta usando `DESC TYPE`.

```
DESC TYPES;

Keyspace my_keyspace
-----
my_udt1  my_udt2

Keyspace my_keyspace2
-----
my_udt1
```

4. Você pode listar todos os UDTs no espaço de teclas selecionado atualmente usando `DESC TYPE`

```
USE my_keyspace;
my_keyspace DESC TYPES;

my_udt1  my_udt2
```


5. Para listar todos os UDTs em um espaço de chave multirregional, você pode consultar a tabela do sistema `types` no espaço de chaves. `system_multiregion_info` A consulta a seguir é um exemplo disso.

```
SELECT keyspace_name, type_name, region, status FROM
system_multiregion_info.types WHERE keyspace_name = 'mykeyspace' AND table_name
= 'mytable';
```

A saída desse comando é semelhante a essa.

keyspace_name	table_name	region	status
mykeyspace	mytable	us-east-1	ACTIVE
mykeyspace	mytable	ap-southeast-1	ACTIVE
mykeyspace	mytable	eu-west-1	ACTIVE

CLI

Exibir tipos definidos pelo usuário (UDTs) com o AWS CLI

1. Para listar os tipos disponíveis em um keyspace, você pode usar o `list-types` comando.

```
aws keyspaces list-types
--keyspace-name 'my_keyspace'
```

A saída desse comando é semelhante a este exemplo.

```
{
  "types": [
    "my_udt",
    "parent_udt"
  ]
}
```

2. Para ver os detalhes sobre um determinado tipo, você pode usar o `get-type` comando.

```
aws keyspaces get-type
--type-name 'my_udt'
--keyspace-name 'my_keyspace'
```

A saída desse comando é semelhante a este exemplo.

```
{
  "keyspaceName": "my_keyspace",
  "typeName": "my_udt",
  "fieldDefinitions": [
    {
      "name": "a",
      "type": "int"
    },
    {
      "name": "b",
      "type": "text"
    }
  ],
  "lastModifiedTimestamp": 1721328225776,
  "maxNestingDepth": 3
  "status": "ACTIVE",
  "directReferringTables": [],
  "directParentTypes": [
    "parent_udt"
  ],
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/"
}
```

Excluir um tipo definido pelo usuário (UDT) no Amazon Keyspaces

Para excluir um UDT em um keyspace, você pode usar a `DROP TYPE` instrução em CQL, o `delete-type` comando com o `o.` ou o AWS CLI console.

Console

Exclua um tipo definido pelo usuário (UDT) com o console Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces e, em seguida, escolha um keyspace na lista.
3. Escolha a guia UDTs.

4. Escolha o UDT que você deseja excluir. Em Usado em, você pode confirmar que o tipo que você deseja excluir não é usado atualmente por uma tabela ou outro UDT.
5. Escolha Excluir acima do Resumo.
6. Digite Delete na caixa de diálogo exibida e escolha Excluir UDT.

Cassandra Query Language (CQL)

Exclua um tipo definido pelo usuário (UDT) com CQL

- Para excluir um tipo, você pode usar a seguinte declaração.

```
DROP TYPE my_keyspace.my_udt;
```

Para obter mais informações sobre sintaxe de CQL, consulte [the section called “DROP TYPE”](#).

CLI

Exclua um tipo definido pelo usuário (UDT) com o AWS CLI

1. Para excluir um tipo, você pode usar o comando a seguir.

```
aws keyspaces delete-type  
--keyspace-name 'my_keyspace'  
--type-name 'my_udt'
```

2. A saída do comando é semelhante a este exemplo.

```
{  
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
my_keyspace/",  
  "typeName": "my_udt"  
}
```

Como trabalhar com consultas CQL 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 em uma consulta no Amazon Keyspaces](#)
- [Use declarações em lote no Amazon Keyspaces](#)
- [Ordenar resultados com ORDER BY no Amazon Keyspaces](#)
- [Pagar resultados no Amazon Keyspaces](#)

Como usar o operador **IN** com a Instrução **SELECT** em uma consulta 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 “Pagar 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

Use declarações em lote no Amazon Keyspaces

Você pode combinar várias DELETE operações INSERTUPDATE, e em uma BATCH declaração. LOGGEDos lotes são o padrão.

```
batch_statement ::=      BEGIN [ UNLOGGED ] BATCH
                        [ USING update_parameter( AND update_parameter)* ]
                        modification_statement ( ';' modification_statement )*
                        APPLY BATCH
modification_statement ::= insert_statement | update_statement | delete_statement
```

Quando você executa uma instrução em lote, o driver combina todas as instruções no lote em uma única operação em lote.

Para decidir qual tipo de operação em lote usar, você pode considerar as diretrizes a seguir.

Use lotes registrados quando:

- Você precisa de garantias de transação atômica.
- Latências um pouco mais altas são uma compensação aceitável.

Use lotes não registrados quando:

- Você precisa otimizar as operações de partição única.
- Você quer reduzir a sobrecarga da rede.
- Você tem requisitos de alto rendimento.

Para obter informações sobre cotas do extrato em lote, consulte [Cotas](#).

Lotes não registrados

Com lotes não registrados, o Amazon Keyspaces processa várias operações como uma única solicitação sem manter um registro de lotes. Com uma operação em lote não registrada, é possível que algumas ações sejam bem-sucedidas e outras falhem. Lotes não registrados são úteis quando você deseja:

- Otimize as operações em uma única partição.
- Reduza o tráfego de rede agrupando solicitações relacionadas.

A sintaxe de um lote não registrado é semelhante à de um lote registrado, com a adição da palavra-chave. UNLOGGED

```
BEGIN UNLOGGED BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Lotes registrados

Um lote registrado combina várias ações de gravação em uma única operação atômica. Quando você executa um lote registrado:

- Todas as ações são bem-sucedidas juntas ou falham juntas.
- A operação é síncrona e idempotente.
- Você pode gravar em várias tabelas do Amazon Keyspaces, desde que estejam na mesma AWS conta e. Região da AWS

Os lotes registrados podem ter latências um pouco maiores. Para aplicações de alto rendimento, considere o uso de lotes não registrados.

Não há custo adicional para usar lotes registrados no Amazon Keyspaces. Você paga somente pelas gravações que fazem parte de suas operações em lote. O Amazon Keyspaces executa duas gravações subjacentes de cada linha do lote: uma para preparar a linha para o lote e outra para confirmar o lote. Ao planejar a capacidade de tabelas que usam lotes registrados, lembre-se de que cada linha em um lote exige o dobro da capacidade de uma operação de gravação padrão. Por exemplo, se seu aplicativo executa um lote registrado por segundo com três linhas de 1 KB, você precisará provisionar seis unidades de capacidade de gravação (WCUs) em comparação com apenas três WCUs para gravações individuais ou lotes não registrados.

Para obter informações sobre preços, consulte o [Amazon Keyspaces \(para Apache Cassandra\)](#).

Práticas recomendadas para operações em lote

Considere as seguintes práticas recomendadas ao usar operações em lote do Amazon Keyspaces.

- Ative o escalonamento automático para que você tenha capacidade de processamento suficiente para que suas tabelas lidem com operações em lote e com os requisitos adicionais de produtividade dos lotes registrados.
- Use operações individuais ou lotes não registrados quando as operações podem ser executadas de forma independente sem afetar a exatidão do aplicativo.
- Projete seu aplicativo para minimizar as atualizações simultâneas nas mesmas linhas, pois as operações simultâneas em lote podem entrar em conflito e falhar.
- Para ingestão de dados em massa de alto rendimento sem requisitos de atomicidade, use operações de gravação individuais ou lotes não registrados.

Consistência e simultaneidade

O Amazon Keyspaces impõe as seguintes regras de consistência e simultaneidade para lotes registrados:

- Todas as operações em lote usam o nível de LOCAL_QUORUM consistência.
- Lotes simultâneos que afetam linhas diferentes podem ser executados simultaneamente.
- DELETEOperações INSERT UPDATE simultâneas ou em linhas envolvidas em um lote em andamento falham devido a um conflito.

Operadores e condições suportados

Operadores de WHERE cláusulas compatíveis:

- Igualdade (=)

Operadores não suportados:

- Operadores de faixa (>, <, >=, <=)
- Operador IN
- Operador LIKE
- Operador BETWEEN

Não suportado em lotes registrados:

- Várias declarações afetando a mesma linha
- Operações de balcão
- Exclusões de intervalo

Condições de falha das declarações de lote registradas

Uma operação em lote registrada pode falhar em qualquer um dos seguintes casos:

- Expressões condicionais (como IF NOT EXISTS ou IF) são avaliadas como falsas.
- Uma ou mais operações contêm parâmetros inválidos.
- A solicitação está em conflito com outra operação em lote em execução nas mesmas linhas.
- A tabela não tem capacidade provisionada suficiente.
- Uma linha excede o limite máximo de tamanho.
- O formato dos dados de entrada é inválido.

Declarações em lote e replicação multirregional

Em implantações multirregionais:

- As operações da região de origem são síncronas e atômicas.
- As operações da região de destino são assíncronas.
- Todas as operações em lote são replicadas para as regiões de destino, mas podem não manter o isolamento durante a aplicação.

Monitore as operações em lote

Você pode monitorar operações em lote usando CloudWatch métricas da Amazon para monitorar desempenho, erros e padrões de uso. O Amazon Keyspaces fornece as seguintes CloudWatch métricas para monitorar operações em lote por tabela:

- **SuccessfulRequestCount**— Rastreie operações em lote bem-sucedidas.
- **Latency**— Meça o desempenho da operação em lote.
- **ConsumedWriteCapacityUnits**— Monitore o consumo de capacidade das operações em lote.

Para obter mais informações, consulte [the section called “Métricas do Amazon Keyspaces”](#).

Além das CloudWatch métricas, você pode usar AWS CloudTrail para registrar todas as ações da API do Amazon Keyspaces. Cada ação de API no lote é registrada, CloudTrail facilitando o rastreamento e a auditoria de operações em lote nas suas tabelas do Amazon Keyspaces.

Exemplos de operação em lote

Veja a seguir um exemplo de uma declaração básica de lote registrada.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Esse é um exemplo de um lote que inclui DELETE instruções INSERTUPDATE, e.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  UPDATE users SET firstname = 'Johnny' WHERE id = 2;
  DELETE FROM users WHERE id = 3;
APPLY BATCH;
```

Este é um exemplo de um lote usando carimbos de data/hora do lado do cliente.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Stiles') USING
  TIMESTAMP 1669069624;
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Doe') USING
  TIMESTAMP 1669069624;
APPLY BATCH;
```

```
BEGIN BATCH
  UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Carlos' WHERE id = 1;
  UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Diego' WHERE id = 2;
APPLY BATCH;
```

Esse é um exemplo de lote condicional.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'Jane', 'Doe') IF NOT
  EXISTS;
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'John', 'Doe') IF NOT
  EXISTS;
APPLY BATCH;
```

```
BEGIN BATCH
  UPDATE users SET lastname = 'Stiles' WHERE id = 1 IF lastname = 'Doe';
  UPDATE users SET lastname = 'Stiles' WHERE id = 2 IF lastname = 'Doe';
APPLY BATCH;
```

Este é um exemplo de um lote usando Time to Live (TTL).

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe') USING TTL
  3600;
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith') USING TTL
  7200;
APPLY BATCH;
```

Esse é um exemplo de uma instrução em lote que atualiza várias tabelas.

```
BEGIN BATCH
  INSERT INTO users (id, firstname) VALUES (1, 'John');
  INSERT INTO user_emails (user_id, email) VALUES (1, 'john@example.com');
APPLY BATCH;
```

Esse é um exemplo de uma operação em lote usando tipos definidos pelo usuário (UDTs). O exemplo pressupõe que o UDT `address` exista.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, address)
```

```
VALUES (1, 'John', {street: '123 Main St', city: 'NYC', zip: '10001'});
INSERT INTO users (id, firstname, address)
VALUES (2, 'Jane', {street: '456 Oak Ave', city: 'LA', zip: '90210'});
APPLY BATCH;

BEGIN BATCH
  UPDATE users SET address.zip = '10002' WHERE id = 1;
  UPDATE users SET address.city = 'Boston' WHERE id = 2;
APPLY BATCH;
```

Ordenar resultados com **ORDER BY** 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”](#).

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 o `PAGE SIZE` das consultas `LIMIT` podem afetar a capacidade de leitura, consulte [the section called “Estimar o consumo de capacidade de leitura de consultas limitadas”](#).

Para ver uma comparação com a paginação do Apache Cassandra, consulte [the section called “Paginação”](#).

Trabalhando com fluxos de captura de dados de alteração (CDC) no Amazon Keyspaces

A captura de dados de alteração (CDC) do Amazon Keyspaces registra eventos de alteração em nível de linha de uma tabela do Amazon Keyspaces quase em tempo real.

O Amazon Keyspaces CDC permite casos de uso orientados por eventos, como IoT industrial e detecção de fraudes, bem como casos de uso de processamento de dados, como pesquisa de texto completo e arquivamento de dados. Os eventos de mudança que o Amazon Keyspaces CDC captura em fluxos podem ser consumidos por aplicativos downstream que executam funções essenciais aos negócios, como análise de dados, pesquisa de texto, ML training/inference e backups contínuos de dados para arquivamento. Por exemplo, você pode transferir dados de stream para serviços de AWS análise e armazenamento, como Amazon OpenSearch Service, Amazon Redshift e Amazon S3, para processamento adicional.

O Amazon Keyspaces CDC oferece registros de alterações ordenados por tempo e sem duplicação para tabelas, com escalabilidade automática da taxa de transferência de dados e tempo de retenção de até 24 horas.

Os streams CDC do Amazon Keyspaces são totalmente sem servidor, e você não precisa gerenciar a infraestrutura de dados para capturar eventos de mudança. Além disso, o Amazon Keyspaces CDC não consome nenhuma capacidade de tabela para computação ou armazenamento. Para obter mais informações, consulte [the section called “Como funcionam os streams do CDC”](#).

Você pode usar a API Amazon Keyspaces Streams para criar aplicativos que consomem streams CDC do Amazon Keyspaces e agir com base no conteúdo. Para ver os endpoints disponíveis, consulte [the section called “Acesse endpoints de fluxo CDC”](#).

Para obter uma lista completa de todas as operações disponíveis para Amazon Keyspaces na API Streams, consulte Amazon [Keyspaces](#) Streams API Reference.

Tópicos

- [Como os fluxos de captura de dados de alteração \(CDC\) funcionam no Amazon Keyspaces](#)

- [Como usar fluxos de captura de dados de alteração \(CDC\) no Amazon Keyspaces](#)

Como os fluxos de captura de dados de alteração (CDC) funcionam no Amazon Keyspaces

Esta seção fornece uma visão geral de como os fluxos de captura de dados de alteração (CDC) funcionam no Amazon Keyspaces.

A captura de dados de alterações (CDC) do Amazon Keyspaces registra uma sequência ordenada de modificações em nível de linha nas tabelas do Amazon Keyspaces e armazena essas informações em um log chamado stream por até 24 horas. Cada modificação em nível de linha gera um novo registro CDC que contém as informações da coluna-chave primária, bem como os estados “antes” e “depois” da linha, incluindo todas as colunas. Os aplicativos podem acessar o stream e visualizar as mutações quase em tempo real.

Quando você habilita o CDC em sua tabela, o Amazon Keyspaces cria um novo stream de CDC e começa a capturar informações sobre cada modificação na tabela. O stream do CDC tem um Amazon Resource Name (ARN) com o seguinte formato:

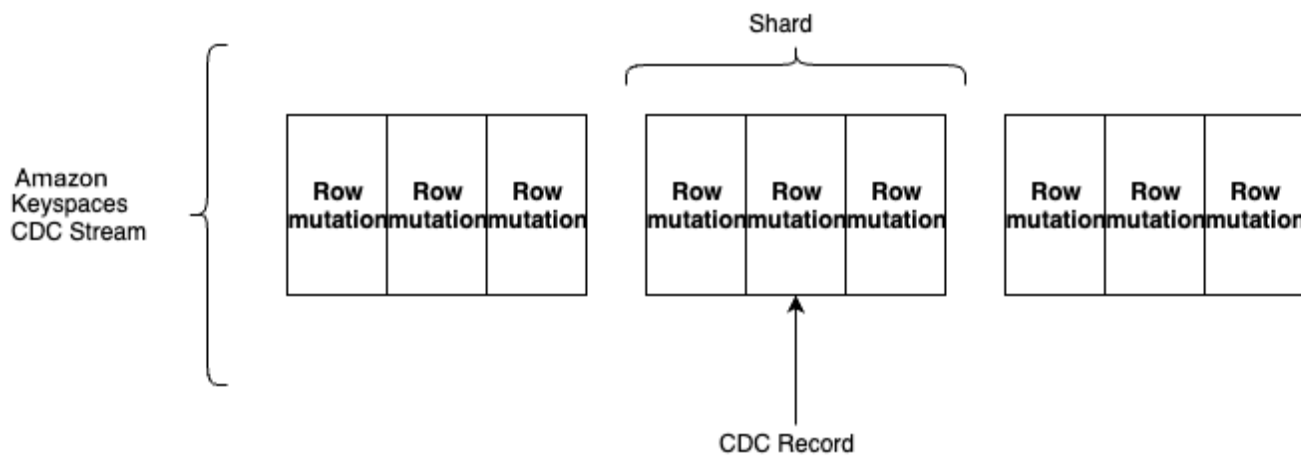
```
arn:${Partition}:cassandra:{Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Você pode selecionar o tipo de informação ou o tipo de visualização que o fluxo CDC coleta para cada registro quando você ativa o fluxo CDC pela primeira vez. Você não pode alterar o tipo de visualização do stream posteriormente. O Amazon Keyspaces oferece suporte aos seguintes tipos de visualização:

- **NEW_AND_OLD_IMAGES**— Captura as versões da linha antes e depois da mutação. Esse é o padrão.
- **NEW_IMAGE**— Captura a versão da linha após a mutação.
- **OLD_IMAGE**— Captura a versão da linha antes da mutação.
- **KEYS_ONLY**— Captura a partição e as chaves de agrupamento da linha que sofreu mutação.

Cada fluxo do CDC consiste em registros. Cada registro representa uma modificação de uma única linha em uma tabela do Amazon Keyspaces. Os registros são organizados logicamente em grupos conhecidos como fragmentos. Esses grupos são organizados logicamente por intervalos da chave primária (combinação de chave de partição, intervalos de chaves de agrupamento) e são uma

construção interna do Amazon Keyspaces. Cada fragmento atua como um contêiner para vários registros e contém as informações necessárias para acessar e iterar esses registros.



Cada registro do CDC recebe um número de sequência, refletindo a ordem em que o registro foi publicado no fragmento. É garantido que o número de sequência seja crescente e exclusivo em cada fragmento.

O Amazon Keyspaces cria e exclui fragmentos automaticamente. Com base nas cargas de tráfego, o Amazon Keyspaces também pode dividir ou mesclar fragmentos ao longo do tempo. Por exemplo, o Amazon Keyspaces pode dividir um fragmento em vários novos fragmentos ou mesclar fragmentos em um novo fragmento único. As APIs do Amazon Keyspaces publicam as informações do fragmento e do fluxo do CDC para permitir que aplicativos consumidores processem registros na ordem certa acessando todo o gráfico de linhagem de um fragmento.

O Amazon Keyspaces CDC é baseado nos seguintes princípios nos quais você pode confiar ao criar seu aplicativo:

- Cada registro de mutação em nível de linha aparece exatamente uma vez no fluxo do CDC.
- Quando você consome fragmentos em ordem de linhagem, cada registro de mutação em nível de linha aparece na mesma sequência da ordem real de mutação na chave primária.

Tópicos

- [Como a retenção de dados funciona para fluxos de CDC no Amazon Keyspaces](#)
- [Como a expiração de dados Time to Live \(TTL\) funciona com fluxos de CDC no Amazon Keyspaces](#)
- [Como as operações em lote funcionam para fluxos de CDC no Amazon Keyspaces](#)

- [Como as colunas estáticas funcionam em fluxos de CDC no Amazon Keyspaces](#)
- [Como a criptografia em repouso funciona para fluxos de CDC no Amazon Keyspaces](#)
- [Como a replicação multirregional funciona para fluxos de CDC no Amazon Keyspaces](#)
- [Streams do CDC e integração com serviços AWS](#)

Como a retenção de dados funciona para fluxos de CDC no Amazon Keyspaces

O Amazon Keyspaces retém os registros no stream do CDC por um período de 24 horas. Você não pode alterar o período de retenção. Se você desativar o CDC em uma tabela, os dados no fluxo continuarão legíveis por 24 horas. Após esse período, os dados expiram e os registros são excluídos automaticamente.

Como a expiração de dados Time to Live (TTL) funciona com fluxos de CDC no Amazon Keyspaces

O Amazon Keyspaces mostra o tempo de expiração no nível e no column/cell nível da linha em um campo de metadados chamado `expirationTime` nos registros de alteração do CDC. Quando o Amazon Keyspaces TTL detecta a expiração de uma célula, o CDC cria um novo registro de alteração que mostra o TTL como a origem da alteração. Para mais informações sobre TTL, consulte [the section called “Dados expirados com vida útil”](#).

Como as operações em lote funcionam para fluxos de CDC no Amazon Keyspaces

As operações em lote são divididas internamente em modificações individuais em nível de linha. O Amazon Keyspaces retém todos os registros nos fluxos do CDC no nível da linha, mesmo que a modificação tenha ocorrido em uma operação em lote. O Amazon Keyspaces mantém a ordem dos registros no stream do CDC na mesma sequência da ordem de mutação que ocorreu no nível da linha ou na chave primária.

Como as colunas estáticas funcionam em fluxos de CDC no Amazon Keyspaces

Os valores das colunas estáticas são compartilhados entre todas as linhas em uma partição no Cassandra. Devido a esse comportamento, o Amazon Keyspaces captura todas as atualizações em uma coluna estática como um registro separado no stream do CDC. Os exemplos a seguir resumem o comportamento das mutações estáticas da coluna:

- Quando somente a coluna estática é atualizada, o fluxo CDC contém uma modificação de linha para a coluna estática como a única coluna na linha.

- Quando uma linha é atualizada sem nenhuma alteração na coluna estática, o fluxo do CDC contém uma modificação de linha que contém todas as colunas, exceto a coluna estática.
- Quando uma linha é atualizada junto com a coluna estática, o fluxo CDC contém duas modificações de linha separadas, uma para a coluna estática e outra para o resto da linha.

Como a criptografia em repouso funciona para fluxos de CDC no Amazon Keyspaces

Para criptografar os dados em repouso no registro ordenado do CDC, o Amazon Keyspaces usa a mesma chave de criptografia que já é usada para a tabela. Para obter mais informações sobre criptografia em repouso, consulte [the section called “Criptografia em repouso”](#).

Como a replicação multirregional funciona para fluxos de CDC no Amazon Keyspaces

Você pode ativar e desativar fluxos CDC para réplicas individuais de uma tabela multirregional usando a `update-table` API ou o comando CQL. `ALTER TABLE` Devido à replicação assíncrona e à resolução de conflitos, os fluxos de CDC para tabelas multirregionais não são consistentes entre si. Regiões da AWS Portanto, os registros que o Amazon Keyspaces captura no stream podem aparecer em uma ordem diferente em diferentes regiões.

Para obter mais informações sobre a replicação multirregional, consulte [the section called “Multi-Region replicação”](#)

Streams do CDC e integração com serviços AWS

Como trabalhar com VPC endpoints para streams de CDC no Amazon Keyspaces

Você pode usar VPC endpoints para acessar os streams CDC do Amazon Keyspaces. Para obter informações sobre como criar e acessar VPC endpoints para streams, consulte [the section called “Usando endpoints VPC de interface com fluxos”](#)

Como CloudWatch funciona o monitoramento com fluxos de CDC no Amazon Keyspaces

Você pode usar CloudWatch a Amazon para monitorar chamadas de API feitas para o endpoint CDC do Amazon Keyspaces. Para obter mais informações sobre as métricas disponíveis, consulte [the section called “Métricas da captura de dados de alteração \(CDC\) do Amazon Keyspaces”](#).

Como CloudTrail funciona o login com streams de CDC no Amazon Keyspaces

O Amazon Keyspaces CDC está integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou serviço AWS 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.

Para obter mais informações sobre os eventos do CDC capturados por CloudTrail, consulte [the section called “Fazendo login com CloudTrail”](#).

Como a marcação funciona para fluxos de CDC no Amazon Keyspaces

Os streams CDC do Amazon Keyspaces são um recurso etiquetável. Você pode marcar um stream ao criar uma tabela programaticamente usando o CQL, o AWS SDK ou o AWS CLI. Você também pode marcar streams existentes, excluir tags ou visualizar tags de um stream. Para obter mais informações, consulte [the section called “Marque espaços de teclas, tabelas e fluxos”](#).

Como usar fluxos de captura de dados de alteração (CDC) no Amazon Keyspaces

Tópicos

- [Configure permissões para trabalhar com fluxos de CDC no Amazon Keyspaces](#)
- [Como acessar endpoints de stream do CDC no Amazon Keyspaces](#)
- [Ative um stream do CDC ao criar uma nova tabela no Amazon Keyspaces](#)
- [Habilite um stream CDC para uma tabela existente no Amazon Keyspaces](#)
- [Desativar um stream CDC no Amazon Keyspaces](#)
- [Veja os streams do CDC no Amazon Keyspaces](#)
- [Acesse registros em fluxos de CDC no Amazon Keyspaces](#)
- [Use a Kinesis Client Library \(KCL\) para processar streams do Amazon Keyspaces](#)

Configure permissões para trabalhar com fluxos de CDC no Amazon Keyspaces

Para habilitar fluxos de CDC, o diretor, por exemplo, um usuário ou uma função do IAM, precisa das seguintes permissões.

Para obter mais informações sobre AWS Identity and Access Management, consulte [the section called “AWS Identity and Access Management”](#).

Permissões para habilitar um fluxo CDC para uma tabela

Para habilitar um stream do CDC para uma tabela do Amazon Keyspaces, o principal primeiro precisa de permissões para criar ou alterar uma tabela e, em segundo lugar, das permissões para criar a função vinculada ao serviço. [AWSServiceRoleForAmazonKeyspacesCDC](#) O Amazon Keyspaces usa a função vinculada ao serviço para publicar CloudWatch métricas em sua conta em seu nome.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Create",
        "cassandra:CreateMultiRegionResource",
        "cassandra:Alter",
        "cassandra:AlterMultiRegionResource"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/*",
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    },
    {
      "Sid": "KeyspacesCDCServiceLinkedRole",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-streams.amazonaws.com/AWSServiceRoleForAmazonKeyspacesCDC",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
        }
      }
    }
  ]
}
```

Para desativar um stream, somente ALTER TABLE permissões são necessárias.

Permissões para visualizar um stream do CDC

Para visualizar ou listar fluxos do CDC, o diretor precisa de permissões de leitura para o espaço de teclas do sistema. Para obter mais informações, consulte [the section called “system_schema_mcs”](#).

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Select",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

Para visualizar ou listar streams do CDC com a API AWS CLI ou Amazon Keyspaces, o diretor precisa de permissões adicionais para as ações `cassandra:ListStreams` e `cassandra:GetStream`.

A política do IAM a seguir é um exemplo disso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select",
        "cassandra:ListStreams",
        "cassandra:GetStream"
      ],
      "Resource": "*"
    }
  ]
}
```

Permissões para ler um fluxo do CDC

Para ler fluxos do CDC, o diretor precisa das seguintes permissões.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:GetStream",
        "cassandra:GetShardIterator",
        "cassandra:GetRecords"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/stream_label"
      ]
    }
  ]
}
```

Permissões para processar streams CDC do Amazon Keyspaces com a Kinesis Client Library (KCL)

Para processar streams CDC do Amazon Keyspaces com a KCL, o diretor do IAM precisa das seguintes permissões.

- Amazon Keyspaces— Read-only acesso a um stream especificado do Amazon Keyspaces CDC.
- DynamoDB— Permissões para criar shard lease tabelas, acesso de leitura e gravação às tabelas e acesso de leitura ao índice, conforme necessário para o processamento do fluxo KCL.
- CloudWatch— Permissões para publicar dados métricos do processamento de streams CDC do Amazon Keyspaces com KCL no namespace do seu aplicativo cliente KCL em sua conta. CloudWatch Para obter mais informações sobre monitoramento, consulte [Monitorar a biblioteca cliente do Kinesis com a Amazon](#). CloudWatch

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



```

    "Action":[
        "cassandra:GetStream",
        "cassandra:GetShardIterator",
        "cassandra:GetRecords"
    ],
    "Resource":[
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/stream_label"
    ]
},
{
    "Effect":"Allow",
    "Action":[
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb:UpdateTable",
        "dynamodb:GetItem",
        "dynamodb:UpdateItem",
        "dynamodb:PutItem",
        "dynamodb>DeleteItem",
        "dynamodb:Scan"
    ],
    "Resource":[
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME"
    ]
},
{
    "Effect":"Allow",
    "Action":[
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb:GetItem",
        "dynamodb:UpdateItem",
        "dynamodb:PutItem",
        "dynamodb>DeleteItem",
        "dynamodb:Scan"
    ],
    "Resource":[
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-
WorkerMetricStats",
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-
CoordinatorState"
    ]
},

```

```
{
  "Effect": "Allow",
  "Action": [
    "dynamodb:Query"
  ],
  "Resource": [
    "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME/index/*"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:PutMetricData"
  ],
  "Resource": "*"
}
]
```

Como acessar endpoints de stream do CDC no Amazon Keyspaces

O Amazon Keyspaces mantém [endpoints](#) separados para keyspaces/tables e para fluxos de CDC em cada um onde o Região da AWS Amazon Keyspaces está disponível. Para acessar um fluxo CDC, selecione a região da tabela e substitua o cassandra prefixo pelo nome do endpoint, conforme mostrado no exemplo a seguir: `cassandra-streams`

Nome da região	Região	Endpoint para espaços de chave e tabelas	Endpoint para streams
Leste dos EUA (Norte da Virgínia)	us-east-1	<code>cassandra</code> .us-east-1.api.aws	<code>cassandra-streams</code> .us-east-1.api.aws

A tabela a seguir contém uma lista completa dos endpoints públicos disponíveis para Amazon Keyspaces change data capture streams. Amazon Keyspaces CDC streams suporta IPv4 e IPv6. Todos os endpoints públicos, por exemplo `cassandra-streams.us-east-1.api.aws`, são endpoints de pilha dupla que podem ser configurados para IPv4 e IPv6.

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	
Leste dos EUA (Norte da Virgínia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
Oeste dos EUA (Norte da Califórnia)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
Oeste dos EUA (Oregon)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
África (Cidade do Cabo)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Ásia-Pacífico (Hong Kong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Ásia-Pacífico (Malásia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Ásia-Pacífico (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Ásia-Pacífico (Seul)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Ásia-Pacífico (Singapura)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	
Ásia-Pacífico (Sydney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS	
Ásia-Pacífico (Tailândia)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS	
Ásia-Pacífico (Tóquio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS	
Canadá (Central)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS	
Europa (Frankfurt)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS	
Europa (Irlanda)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS	
Europa (Londres)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Europa (Paris)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Europa (Estocolmo)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Oriente Médio (Barém)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Oriente Médio (Emirados Árabes Unidos)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
América do Sul (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Ative um stream do CDC ao criar uma nova tabela no Amazon Keyspaces

Para habilitar um stream CDC ao criar uma tabela, você pode usar a `CREATE TABLE` instrução em CQL ou o `create-table` comando com o AWS CLI

Para cada linha alterada na tabela, o Amazon Keyspaces pode capturar as seguintes alterações com base nas `view_type` `cdc_specification` que você selecionar:

- `NEW_AND_OLD_IMAGES`— ambas as versões da linha, antes e depois da alteração. Esse é o padrão.
- `NEW_IMAGE`— a versão da linha após a alteração.
- `OLD_IMAGE`— a versão da linha antes da alteração.
- `KEYS_ONLY`— as chaves de partição e agrupamento da linha que foi alterada.

Para obter informações sobre como marcar um stream, consulte [the section called “Crie um stream com tags com a nova tabela”](#).

Note

O Amazon Keyspaces CDC exige a presença de uma função vinculada ao serviço (`AWSServiceRoleForAmazonKeyspacesCDC`) que publique dados métricos dos streams do Amazon Keyspaces CDC na sua conta em seu nome. `"cloudwatch:namespace": "AWS/Cassandra"` CloudWatch Esta função é criada automaticamente para você. Para obter mais informações, consulte [the section called “Transmissões CDC”](#).

Cassandra Query Language (CQL)

Ative um fluxo CDC ao criar uma tabela com CQL

1.

```
CREATE TABLE mykeyspace.mytable (a text, b text, PRIMARY KEY(a))
WITH CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'NEW_IMAGE'}} AND CDC
= TRUE;
```
2. Para confirmar as configurações de transmissão, você pode usar a seguinte declaração.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

A saída dessa declaração deve ser semelhante a essa.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';keyspace_name | table_name | cdc | custom_properties
-----+-----+-----
+-----+-----+-----+-----
mykeyspace | mytable | True | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741383893782',
'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
{'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T21:44:53.783', 'status': 'ENABLED',
'view_type': 'NEW_IMAGE'}, 'encryption_specification': {'encryption_type':
'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}>
```

CLI

Ative um fluxo CDC ao criar uma tabela com o AWS CLI

1. Para criar um stream, você pode usar a seguinte sintaxe.

```
aws keyspaces create-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--schema-definition 'allColumns=[{name=a,type=text},
{name=b,type=text}],partitionKeys=[{name=a}]' \
--cdc-specification status=ENABLED,viewType=NEW_IMAGE
```

2. A saída desse comando mostra a create-table resposta padrão e é semelhante a este exemplo.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

Habilite um stream CDC para uma tabela existente no Amazon Keyspaces

Para habilitar um stream CDC para uma tabela existente, você pode usar a `ALTER TABLE` instrução em CQL, o `update-table` comando com o AWS CLI ou você pode usar o console.

Para cada linha alterada na tabela, o Amazon Keyspaces pode capturar as seguintes alterações com base nas `view_type` `cdc_specification` que você selecionar:

- `NEW_AND_OLD_IMAGES`— ambas as versões da linha, antes e depois da alteração. Esse é o padrão.
- `NEW_IMAGE`— a versão da linha após a alteração.
- `OLD_IMAGE`— a versão da linha antes da alteração.
- `KEYS_ONLY`— as chaves de partição e agrupamento da linha que foi alterada.

Para obter informações sobre como marcar um stream, consulte [the section called “Adicionar novas tags a um stream”](#).

Note

O Amazon Keyspaces CDC exige a presença de uma função vinculada ao serviço (`AWSServiceRoleForAmazonKeyspacesCDC`) que publique dados métricos dos streams do Amazon Keyspaces CDC na sua conta em seu nome. `"cloudwatch:namespace": "AWS/Cassandra"` CloudWatch Esta função é criada automaticamente para você. Para obter mais informações, consulte [the section called “Transmissões CDC”](#).

Cassandra Query Language (CQL)

Ativar um fluxo (fluxo CDC) com CQL

Você pode usar `ALTER TABLE` para habilitar um stream para uma tabela existente.

1. O exemplo a seguir cria um fluxo que captura somente as alterações na partição e nas chaves de agrupamento de uma linha alterada.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = TRUE
AND CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'KEYS_ONLY'}};
```


2. Para verificar as configurações do stream, você pode usar a seguinte declaração.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

A saída da declaração é semelhante a essa.

```
keyspace_name | table_name | cdc | custom_properties
-----+-----+-----+-----
+-----+-----+-----+-----
mykeyspace | mytable | True | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741385897045',
'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
{'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T22:20:10.454', 'status': 'ENABLED',
'view_type': 'KEYS_ONLY'}, 'encryption_specification': {'encryption_type':
'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Crie um fluxo CDC com o AWS CLI

1. Para criar um fluxo para uma tabela existente, você pode usar a sintaxe a seguir.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=ENABLED,viewType=NEW_AND_OLD_IMAGES
```

2. A saída desse comando mostra a create-table resposta padrão e é semelhante a este exemplo.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

Console

Habilite um stream do CDC com o console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha uma tabela na lista.
3. Escolha a guia Streams.
4. Escolha Editar para ativar um stream.
5. Selecione Ativar transmissões.
6. Escolha o tipo de exibição do stream. As seguintes opções estão disponíveis: Observe que você não pode alterar o tipo de visualização de um stream depois que ele foi criado.
 - Imagens novas e antigas — o Amazon Keyspaces captura as duas versões da linha, antes e depois da alteração. Esse é o padrão.
 - Nova imagem — O Amazon Keyspaces captura somente a versão da linha após a alteração.
 - Imagem antiga — O Amazon Keyspaces captura somente a versão da linha antes da alteração.
 - Somente chave primária — O Amazon Keyspaces captura somente as colunas de partição e chave de agrupamento da linha que foi alterada.
7. Para finalizar, escolha Salvar alterações.

Desativar um stream CDC no Amazon Keyspaces

Para desativar um stream CDC em um keyspace, você pode usar a `ALTER TABLE` instrução em CQL, o `update-table` comando com o. ou o AWS CLI console.

Cassandra Query Language (CQL)

Desativar um fluxo (fluxo CDC) com CQL

1. Para desativar um stream, você pode usar a seguinte declaração.

```
ALTER TABLE mykeyspace.mytable  
WITH cdc = FALSE;
```

2. Para confirmar se o stream está desativado, você pode usar a seguinte declaração.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

A saída dessa declaração é semelhante a essa.

```
keyspace_name | table_name | cdc    | custom_properties
-----+-----+-----+-----
+-----+-----+-----+-----
mykeyspace   | mytable   | False  | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741385668642',
'throughput_mode': 'PAY_PER_REQUEST'}, 'encryption_specification':
{'encryption_type': 'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status':
'disabled'}}
```

CLI

Desative um fluxo (fluxo CDC) com o AWS CLI

1. Para desativar um stream, você pode usar o comando a seguir.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=DISABLED
```

2. A saída do comando é semelhante a este exemplo.

```
{
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "streamName": "my_stream"
}
```

Console

Desative um stream (stream CDC) com o console Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha uma tabela na lista.
3. Escolha a guia Streams.
4. Escolha Editar.
5. Desmarque Ativar transmissões.
6. Escolha Salvar alterações para desativar o stream.

Veja os streams do CDC no Amazon Keyspaces

Para visualizar ou listar todos os fluxos no keyspace, você pode consultar a tabela `system_schema_mcs.streams` no keyspace do sistema usando uma instrução em CQL ou usar os `list-stream` comandos `get-stream` and com o AWS CLI, ou o console.

Para ver as permissões necessárias, consulte [the section called “Configurar permissões do ”](#).

Cassandra Query Language (CQL)

Visualize streams do CDC com CQL

- Para monitorar o status do CDC da sua tabela, você pode usar a seguinte declaração.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='my_keyspace' and table_name='my_table';
```

A saída do comando é semelhante a essa.

```
...
custom_properties
-----
{'cdc_specification':{'status': 'Enabled', 'view_type': 'NEW_IMAGE',
  'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table/stream/stream_label'}}
...
```

CLI

Veja os streams do CDC com o AWS CLI

1. Este exemplo mostra como ver as informações de fluxo de uma tabela.

```
aws keyspaces get-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table'
```

A saída do comando tem a seguinte aparência.

```
{  
  "keyspaceName": "my_keyspace",  
  "tableName": "my_table",  
  ... Other fields ...,  
  "latestStreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
my_keyspace/table/my_table/stream/stream_label",  
  "cdcSpecification": {  
    "status": "ENABLED",  
    "viewType": "NEW_AND_OLD_IMAGES"  
  }  
}
```

2. Você pode listar todos os streams em sua conta em um determinado Região da AWS local. O comando a seguir é um exemplo disso.

```
aws keyspacesstreams list-streams --region us-east-1
```

A saída do comando pode ser semelhante a essa.

```
{  
  "Streams": [  
    {  
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
ks_1/table/t1/stream/2023-05-11T21:21:33.291",  
      "StreamLabel": "2023-05-11T21:21:33.291",  
      "KeyspaceName": "ks_1",  
      "TableName": "t1",  
    },  
    {
```

```

    "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
    "StreamLabel": "2023-05-11T21:21:33.291",
    "KeyspaceName": "ks_1"Create a keyspace with the name catalog. Note
                        that streams are not supported in multi-Region
keyspaces.
    "TableName": "t2",
  },
  {
    "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_2/table/t1/stream/2023-05-11T21:21:33.291",
    "StreamLabel": "2023-05-11T21:21:33.291",
    "KeyspaceName": "ks_3"
    "TableName": "t1",
  }
]
}

```

3. Você também pode listar os fluxos CDC para um determinado espaço de teclas usando os seguintes parâmetros.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --region us-east-1
```

A saída do comando é semelhante a essa.

```

{
  "Streams": [
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t1/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t1",
    },
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t2",
    }
  ]
}

```

```
}
```

4. Você também pode listar os fluxos CDC de uma determinada tabela usando os parâmetros a seguir.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --table-name t2 --  
region us-east-1
```

A saída do comando é semelhante a essa.

```
{  
  "Streams": [  
    {  
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
ks_1/table/t2/stream/2023-05-11T21:21:33.291",  
      "StreamLabel": "2023-05-11T21:21:33.291",  
      "KeyspaceName": "ks_1",  
      "TableName": "t2",  
    }  
  ]  
}
```

Console

Veja os streams do CDC no console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha uma tabela na lista.
3. Escolha a guia Streams para revisar os detalhes do stream.

Acesse registros em fluxos do CDC no Amazon Keyspaces

Para acessar os registros em um stream, você usa a API [Amazon Keyspaces Streams](#). A seção a seguir contém exemplos de como acessar registros usando AWS CLI o.

Para ver as permissões necessárias, consulte [the section called “Configurar permissões do ”](#).

Acesse registros em um stream usando o AWS CLI

1. Você pode usar a API Amazon Keyspaces Streams para acessar os registros de alterações do stream. Para obter mais informações, consulte [Amazon Keyspaces Streams API Reference](#). Para recuperar os fragmentos no stream, você pode usar a `get-stream` API conforme mostrado no exemplo a seguir.

```
aws keyspacesstreams get-stream \  
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/  
mytable/stream/STREAM_LABEL '
```

A seguir, veja um exemplo de saída.

```
{  
  "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/  
table/mytable/stream/2023-05-11T21:21:33.291",  
  "StreamStatus": "ENABLED",  
  "StreamViewType": "NEW_AND_OLD_IMAGES",  
  "CreationRequestDateTime": "<CREATION_TIME>",  
  "KeyspaceName": "mykeyspace",  
  "TableName": "mytable",  
  "StreamLabel": "2023-05-11T21:21:33.291",  
  "Shards": [  
    {  
      "SequenceNumberRange": {  
        "EndingSequenceNumber": "<END_SEQUENCE_NUMBER>",  
        "StartingSequenceNumber": "<START_SEQUENCE_NUMBER>"  
      },  
      "ShardId": "<SHARD_ID>"  
    },  
  ]  
}
```

2. Para recuperar registros do stream, você começa obtendo um iterador que fornece o ponto de partida para acessar os registros. Para fazer isso, você pode usar os fragmentos no stream do CDC retornado pela API na etapa anterior. Para reunir o iterador, você pode usar a `get-shard-iterator` API. Neste exemplo, você usa um iterador do tipo `TRIM_HORIZON` que recupera do último ponto recortado (ou início) do fragmento.

```
aws keyspacesstreams get-shard-iterator \  

```



```
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable/stream/STREAM_LABEL' \
--shard-id 'SHARD_ID' \
--shard-iterator-type 'TRIM_HORIZON'
```

A saída do comando se parece com o exemplo a seguir.

```
{
  "ShardIterator": "<SHARD_ITERATOR>"
}
```

3. Para recuperar os registros do CDC usando a `get-records` API, você pode usar o iterador retornado na última etapa. O comando a seguir é um exemplo disso.

```
aws keyspacesstreams get-records \
--shard-iterator 'SHARD_ITERATOR' \
--limit 100
```

Use a Kinesis Client Library (KCL) para processar streams do Amazon Keyspaces

Este tópico descreve como usar a Kinesis Client Library (KCL) para consumir e processar dados dos streams de captura de dados de alteração (CDC) do Amazon Keyspaces.

Em vez de trabalhar diretamente com a API Amazon Keyspaces Streams, trabalhar com a Kinesis Client Library (KCL) oferece muitos benefícios, por exemplo:

- Rastreamento de linhagem de fragmentos e manuseio de iteradores integrados.
- Balanceamento automático de carga entre os trabalhadores.
- Tolerância a falhas e recuperação de falhas de trabalhadores.
- Ponto de verificação para acompanhar o progresso do processamento.
- Adaptação às mudanças na capacidade do fluxo.
- Computação distribuída simplificada para processamento de registros CDC.

A seção a seguir descreve por que e como usar a Kinesis Client Library (KCL) para processar streams e fornece um exemplo para processar um stream CDC do Amazon Keyspaces com a KCL.

Para obter informações sobre preços, consulte o [Amazon Keyspaces \(para Apache Cassandra\)](#).

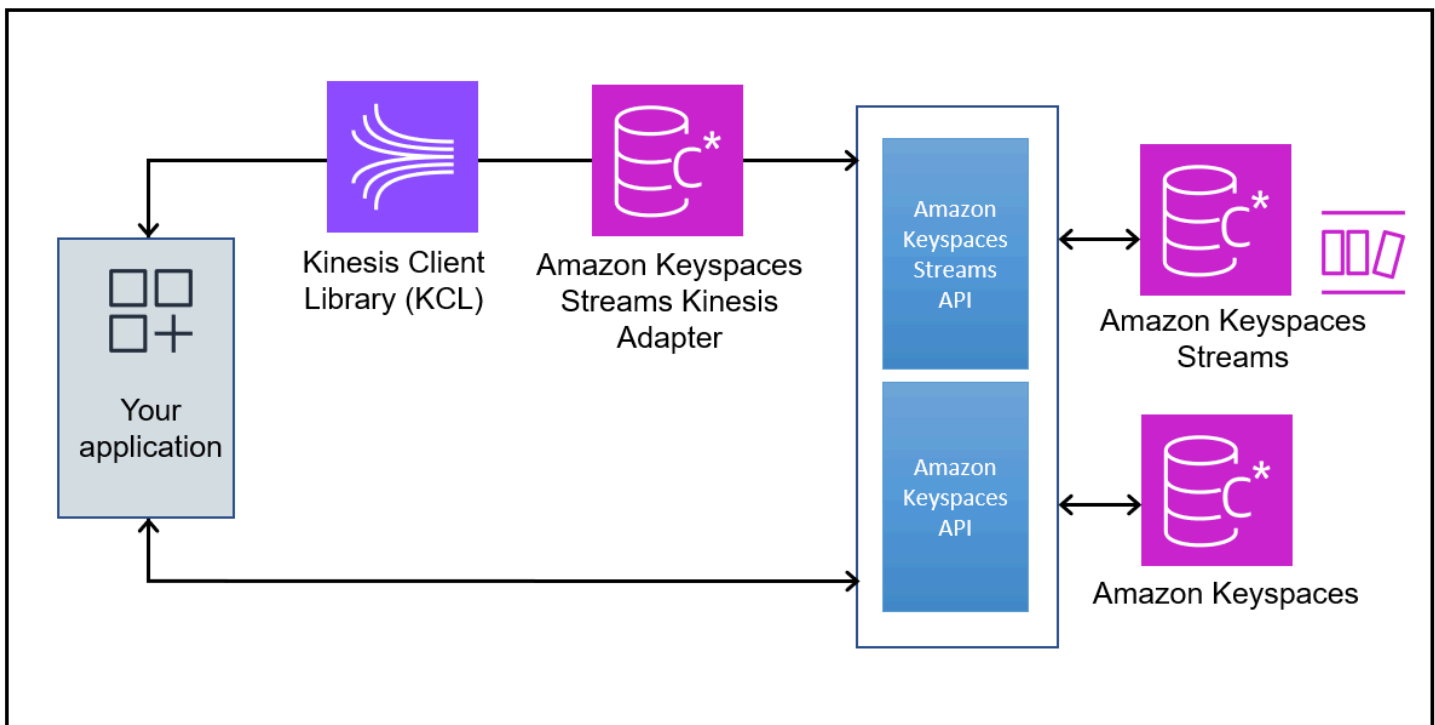
O que é a Kinesis Client Library?

A Kinesis Client Library (KCL) é uma biblioteca de software Java independente projetada para simplificar o processo de consumo e processamento de dados de streams. A KCL lida com muitas das tarefas complexas associadas à computação distribuída, permitindo que você se concentre na implementação de sua lógica de negócios ao processar dados de fluxo. A KCL gerencia atividades como balanceamento de carga entre vários trabalhadores, resposta às falhas dos trabalhadores, verificação de registros processados e resposta às mudanças no número de fragmentos no fluxo.

Para processar streams CDC do Amazon Keyspaces, você pode usar os padrões de design encontrados na KCL para trabalhar com fragmentos de stream e registros de stream. A KCL simplifica a codificação fornecendo abstrações úteis acima da API de baixo nível do Kinesis Data Streams. Para obter mais informações sobre a KCL, consulte [Desenvolver consumidores com a KCL no Guia](#) do desenvolvedor do Amazon Kinesis Data Streams.

Para escrever aplicativos usando a KCL, você usa o Amazon Keyspaces Streams Kinesis Adapter. O Kinesis Adapter implementa a interface do Kinesis Data Streams para que você possa usar a KCL para consumir e processar registros dos streams do Amazon Keyspaces. Para obter instruções sobre como configurar e instalar o adaptador Kinesis de streams do Amazon Keyspaces, visite o repositório. [GitHub](#)

O diagrama a seguir mostra como essas bibliotecas interagem entre si.



A KCL é atualizada com frequência para incorporar as versões mais recentes das bibliotecas subjacentes, promover melhorias de segurança e realizar correções de bugs. É recomendável que você use a versão mais recente da KCL para evitar os problemas já conhecidos e se beneficiar de todas as melhorias mais recentes. Para encontrar a versão mais recente do KCL, consulte o repositório [KCL GitHub](#).

Conceitos da KCL

Antes de implementar um aplicativo de consumidor usando o KCL, você deve entender os seguintes conceitos:

Aplicativo para consumidores KCL

Um aplicativo de consumidor da KCL é um programa que processa dados de um stream CDC do Amazon Keyspaces. A KCL atua como intermediária entre o código do aplicativo do consumidor e o stream CDC do Amazon Keyspaces.

Trabalhador

Um trabalhador é uma unidade de execução do seu aplicativo consumidor da KCL que processa dados do stream CDC do Amazon Keyspaces. Seu aplicativo pode executar vários trabalhadores distribuídos em várias instâncias.

Processador de gravação

Um processador de registros é a lógica em seu aplicativo que processa dados de um fragmento no stream CDC do Amazon Keyspaces. Um processador de registros é instanciado por um trabalhador para cada fragmento que ele gerencia.

Locação

Uma locação representa a responsabilidade pelo processamento de um fragmento. Os trabalhadores usam arrendamentos para coordenar qual trabalhador está processando qual fragmento. A KCL armazena dados de leasing em uma tabela no Amazon DynamoDB.

Checkpoint

Um ponto de verificação é um registro da posição no fragmento até a qual o processador de registros processou os registros com sucesso. O ponto de verificação permite que seu aplicativo retome o processamento de onde parou se um funcionário falhar.

Com o adaptador Amazon Keyspaces Kinesis instalado, você pode começar a desenvolver na interface KCL, com as chamadas de API direcionadas perfeitamente ao endpoint de stream do

Amazon Keyspaces. Para obter uma lista de endpoints disponíveis, consulte [the section called “Acesse endpoints de fluxo CDC”](#).

Quando a aplicação é iniciada, ela chama a KCL para instanciar um operador. Você deve fornecer ao trabalhador as informações de configuração do aplicativo, como o descritor e as AWS credenciais do fluxo, e o nome de uma classe de processador de registros fornecida por você. Como ele executa o código no processador de registro, o operador executa as seguintes tarefas:

- Conecta-se ao stream
- Enumera os fragmentos no fluxo
- Coordena as associações do estilhaço com outros operadores (se houver)
- Cria uma instância de um processador de registro para cada fragmento que gerencia
- Extrai registros do fluxo
- Envia os registros ao processador de registros correspondente
- Registros processados pelos pontos de verificação
- Equilibra as associações de estilhaço-operador quando a contagem de instância de operadores muda
- Equilibra as associações de fragmento-operador quando os fragmentos se dividem

Implemente um aplicativo de consumidor KCL para streams CDC do Amazon Keyspaces

Este tópico fornece um guia passo a passo para implementar um aplicativo de consumidor KCL para processar streams CDC do Amazon Keyspaces.

1. Pré-requisitos: Antes de começar, verifique se você tem:

- Uma tabela do Amazon Keyspaces com um stream CDC
- Permissões do IAM necessárias para que o diretor do IAM acesse o stream CDC do Amazon Keyspaces, crie e acesse tabelas do DynamoDB para processamento do stream KCL e permissões para publicar métricas. CloudWatch Para obter mais informações e um exemplo de política, consulte [the section called “Permissões para KCL”](#).
- Certifique-se de que AWS as credenciais válidas estejam configuradas em sua configuração local. Para obter mais informações, consulte [the section called “Gerenciar chaves de acesso”](#).
- Java Development Kit (JDK) 8 ou posterior
- Requisitos listados no [Readme](#) no Github.

2. Nesta etapa, você adiciona a dependência KCL ao seu projeto. Para o Maven, adicione o seguinte ao seu pom.xml:

```
<dependencies>
  <dependency>
    <groupId>software.amazon.kinesis</groupId>
    <artifactId>amazon-kinesis-client</artifactId>
    <version>3.1.0</version>
  </dependency>
  <dependency>
    <groupId>software.amazon.keyspaces</groupId>
    <artifactId>keyspaces-streams-kinesis-adapter</artifactId>
    <version>1.0.0</version>
  </dependency>
</dependencies>
```

 Note

Sempre verifique a versão mais recente do KCL no repositório [KCL GitHub](#).

3. Crie uma classe de fábrica que produza instâncias do processador de registros:

```
import software.amazon.awssdk.services.keyspacesstreams.model.Record;
import
  software.amazon.keyspaces.streamsadapter.adapter.KeyspacesStreamsClientRecord;
import
  software.amazon.keyspaces.streamsadapter.model.KeyspacesStreamsProcessRecordsInput;
import
  software.amazon.keyspaces.streamsadapter.processor.KeyspacesStreamsShardRecordProcessor;
import software.amazon.kinesis.lifecycle.events.InitializationInput;
import software.amazon.kinesis.lifecycle.events.LeaseLostInput;
import software.amazon.kinesis.lifecycle.events.ShardEndedInput;
import software.amazon.kinesis.lifecycle.events.ShutdownRequestedInput;
import software.amazon.kinesis.processor.RecordProcessorCheckpoint;

public class RecordProcessor implements KeyspacesStreamsShardRecordProcessor {
  private String shardId;

  @Override
  public void initialize(InitializationInput initializationInput) {
    this.shardId = initializationInput.shardId();
    System.out.println("Initializing record processor for shard: " + shardId);
  }
}
```

```

    }

    @Override
    public void processRecords(KeyspacesStreamsProcessRecordsInput
processRecordsInput) {
        try {
            for (KeyspacesStreamsClientRecord record :
processRecordsInput.records()) {
                Record keyspacesRecord = record.getRecord();
                System.out.println("Received record: " + keyspacesRecord);
            }

            if (!processRecordsInput.records().isEmpty()) {
                RecordProcessorCheckpointing checkpointing =
processRecordsInput.checkpointer();
                try {
                    checkpointing.checkpoint();
                    System.out.println("Checkpoint successful for shard: " +
shardId);
                } catch (Exception e) {
                    System.out.println("Error while checkpointing for shard: " +
shardId + " " + e);
                }
            }
        } catch (Exception e) {
            System.out.println("Error processing records for shard: " + shardId + "
" + e);
        }
    }

    @Override
    public void leaseLost(LeaseLostInput leaseLostInput) {
        System.out.println("Lease lost for shard: " + shardId);
    }

    @Override
    public void shardEnded(ShardEndedInput shardEndedInput) {
        System.out.println("Shard ended: " + shardId);
        try {
            // This is required. Checkpoint at the end of the shard
            shardEndedInput.checkpointer().checkpoint();
            System.out.println("Final checkpoint successful for shard: " +
shardId);
        } catch (Exception e) {

```

```

        System.out.println("Error while final checkpointing for shard: " +
shardId + " " + e);
        throw new RuntimeException("Error while final checkpointing", e);
    }
}

@Override
public void shutdownRequested(ShutdownRequestedInput shutdownRequestedInput) {
    System.out.println("Shutdown requested for shard " + shardId);
    try {
        shutdownRequestedInput.checkpointer().checkpoint();
    } catch (Exception e) {
        System.out.println("Error while checkpointing on shutdown for shard: "
+ shardId + " " + e);
    }
}
}

```

4. Crie uma fábrica de registros conforme mostrado no exemplo a seguir.

```

import software.amazon.kinesis.processor.ShardRecordProcessor;
import software.amazon.kinesis.processor.ShardRecordProcessorFactory;

import java.util.Queue;
import java.util.concurrent.ConcurrentLinkedQueue;

public class RecordProcessorFactory implements ShardRecordProcessorFactory {
    private final Queue<RecordProcessor> processors = new
ConcurrentLinkedQueue<>();

    @Override
    public ShardRecordProcessor shardRecordProcessor() {
        System.out.println("Creating new RecordProcessor");
        RecordProcessor processor = new RecordProcessor();
        processors.add(processor);
        return processor;
    }
}

```

5. Nesta etapa, você cria a classe base para configurar o KCLv3 e o adaptador Amazon Keyspaces.

```

import com.example.KCLExample.utils.RecordProcessorFactory;

```

```
import
    software.amazon.keyspaces.streamsadapter.AmazonKeyspacesStreamsAdapterClient;
import software.amazon.keyspaces.streamsadapter.StreamsSchedulerFactory;
import java.util.Arrays;
import java.util.List;
import java.util.concurrent.ExecutionException;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatch.CloudWatchAsyncClient;
import software.amazon.awssdk.services.dynamodb.DynamoDbAsyncClient;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableRequest;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableResponse;
import software.amazon.awssdk.services.keyspacesstreams.KeyspacesStreamsClient;
import software.amazon.awssdk.services.kinesis.KinesisAsyncClient;
import software.amazon.kinesis.common.ConfigsBuilder;
import software.amazon.kinesis.common.InitialPositionInStream;
import software.amazon.kinesis.common.InitialPositionInStreamExtended;
import software.amazon.kinesis.coordinator.CoordinatorConfig;
import software.amazon.kinesis.coordinator.Scheduler;
import software.amazon.kinesis.leases.LeaseManagementConfig;
import software.amazon.kinesis.processor.ProcessorConfig;
import software.amazon.kinesis.processor.StreamTracker;
import software.amazon.kinesis.retrieval.polling.PollingConfig;

public class KCLTestBase {

    protected KeyspacesStreamsClient streamsClient;
    protected KinesisAsyncClient adapterClient;
    protected DynamoDbAsyncClient dynamoDbAsyncClient;
    protected CloudWatchAsyncClient cloudWatchClient;
    protected Region region;
    protected RecordProcessorFactory recordProcessorFactory;
    protected Scheduler scheduler;
    protected Thread schedulerThread;

    public void baseSetUp() {
        recordProcessorFactory = new RecordProcessorFactory();
        setupKCLBase();
    }

    protected void setupKCLBase() {
        region = Region.US_EAST_1;

        streamsClient = KeyspacesStreamsClient.builder()
```



```

        .region(region)
        .build();
    adapterClient = new AmazonKeyspacesStreamsAdapterClient(
        streamsClient,
        region);
    dynamoDbAsyncClient = DynamoDbAsyncClient.builder()
        .region(region)
        .build();
    cloudWatchClient = CloudWatchAsyncClient.builder()
        .region(region)
        .build();
}

protected void startScheduler(Scheduler scheduler) {
    this.scheduler = scheduler;
    schedulerThread = new Thread(() -> scheduler.run());
    schedulerThread.start();
}

protected void shutdownScheduler() {
    if (scheduler != null) {
        scheduler.shutdown();
        try {
            schedulerThread.join(30000);
        } catch (InterruptedException e) {
            System.out.println("Error while shutting down scheduler " + e);
        }
    }
}

protected Scheduler createScheduler(String streamArn, String leaseTableName) {
    String workerId = "worker-" + System.currentTimeMillis();

    // Create ConfigsBuilder
    ConfigsBuilder configsBuilder = createConfigsBuilder(streamArn, workerId,
        leaseTableName);

    // Configure retrieval config for polling
    PollingConfig pollingConfig = new PollingConfig(streamArn, adapterClient);

    // Create the Scheduler
    return StreamsSchedulerFactory.createScheduler(
        configsBuilder.checkpointConfig(),
        configsBuilder.coordinatorConfig(),

```

```

        configsBuilder.leaseManagementConfig(),
        configsBuilder.lifecycleConfig(),
        configsBuilder.metricsConfig(),
        configsBuilder.processorConfig(),

configsBuilder.retrievalConfig().retrievalSpecificConfig(pollingConfig),
        streamsClient,
        region

    );
}

private ConfigsBuilder createConfigsBuilder(String streamArn, String workerId,
String leaseTableName) {
    ConfigsBuilder configsBuilder = new ConfigsBuilder(
        streamArn,
        leaseTableName,
        adapterClient,
        dynamoDbAsyncClient,
        cloudWatchClient,
        workerId,
        recordProcessorFactory);

    configureCoordinator(configsBuilder.coordinatorConfig());
    configureLeaseManagement(configsBuilder.leaseManagementConfig());
    configureProcessor(configsBuilder.processorConfig());
    configureStreamTracker(configsBuilder, streamArn);

    return configsBuilder;
}

private void configureCoordinator(CoordinatorConfig config) {
    config.skipShardSyncAtWorkerInitializationIfLeasesExist(true)
        .parentShardPollIntervalMillis(1000)
        .shardConsumerDispatchPollIntervalMillis(500);
}

private void configureLeaseManagement(LeaseManagementConfig config) {
    config.shardSyncIntervalMillis(0)
        .leasesRecoveryAuditorInconsistencyConfidenceThreshold(0)
        .leasesRecoveryAuditorExecutionFrequencyMillis(5000)
        .leaseAssignmentIntervalMillis(1000L);
}

private void configureProcessor(ProcessorConfig config) {

```

```

        config.callProcessRecordsEvenForEmptyRecordList(true);
    }

    private void configureStreamTracker(ConfigsBuilder configsBuilder, String
streamArn) {
        StreamTracker streamTracker =
StreamsSchedulerFactory.createSingleStreamTracker(
            streamArn,

InitialPositionInStreamExtended.newInitialPosition(InitialPositionInStream.TRIM_HORIZON)
        );
        configsBuilder.streamTracker(streamTracker);
    }

    public void deleteAllDdbTables(String baseTableName) {
        List<String> tablesToDelete = Arrays.asList(
            baseTableName,
            baseTableName + "-CoordinatorState",
            baseTableName + "-WorkerMetricStats"
        );

        for (String tableName : tablesToDelete) {
            deleteTable(tableName);
        }
    }

    private void deleteTable(String tableName) {
        DeleteTableRequest deleteTableRequest = DeleteTableRequest.builder()
            .tableName(tableName)
            .build();

        try {
            DeleteTableResponse response =
dynamoDbAsyncClient.deleteTable(deleteTableRequest).get();
            System.out.println("Table deletion response " + response);
        } catch (InterruptedException | ExecutionException e) {
            System.out.println("Error deleting table: " + tableName + " " + e);
        }
    }
}

```

6. Nesta etapa, você implementa a classe do processador de registros para que seu aplicativo comece a processar eventos de alteração.

```
import software.amazon.kinesis.coordinator.Scheduler;

public class KCLTest {

    private static final int APP_RUNTIME_SECONDS = 1800;
    private static final int SLEEP_INTERNAL_MS = 60*1000;

    public static void main(String[] args) {
        KCLTestBase kclTestBase;

        kclTestBase = new KCLTestBase();
        kclTestBase.baseSetUp();

        // Create and start scheduler
        String leaseTableName = generateUniqueApplicationName();

        // Update below to your Stream ARN
        String streamArn = "arn:aws:cassandra:us-east-1:759151643516:/keyspace/
cdc_sample_test/table/test_kcl_bool/stream/2025-07-01T15:52:57.529";
        Scheduler scheduler = kclTestBase.createScheduler(streamArn,
        leaseTableName);
        kclTestBase.startScheduler(scheduler);

        // Wait for specified time before shutting down - KCL applications are
        // designed to run forever, however in this
        // example we will shut it down after APP_RUNTIME_SECONDS
        long startTime = System.currentTimeMillis();
        long endTime = startTime + (APP_RUNTIME_SECONDS * 1000);
        while (System.currentTimeMillis() < endTime) {
            try {
                // Print and sleep every minute
                Thread.sleep(SLEEP_INTERNAL_MS);
                System.out.println("Application is running");
            } catch (InterruptedException e) {
                System.out.println("Interrupted while waiting for records");
                Thread.currentThread().interrupt();
                break;
            }
        }

        // Stop the scheduler
        kclTestBase.shutdownScheduler();
        kclTestBase.deleteAllDdbTables(leaseTableName);
    }
}
```

```
    }

    public static String generateUniqueApplicationName() {
        String timestamp = String.valueOf(System.currentTimeMillis());
        String randomString = java.util.UUID.randomUUID().toString().substring(0,
8);
        return String.format("KCL-App-%s-%s", timestamp, randomString);
    }
}
```

Práticas recomendadas

Siga estas melhores práticas ao usar a KCL com streams CDC do Amazon Keyspaces:

Como tratar erros

Implemente um tratamento robusto de erros em seu processador de registros para lidar com exceções sem problemas. Considere implementar a lógica de repetição para falhas transitórias.

Frequência de checkpoint

Equilibre a frequência dos pontos de verificação para minimizar o processamento duplicado e, ao mesmo tempo, garantir um rastreamento razoável do progresso. Pontos de verificação muito frequentes podem afetar o desempenho, enquanto pontos de verificação muito pouco frequentes podem levar a mais reprocessamento se um trabalhador falhar.

Escalabilidade de trabalhadores

Dimensione o número de trabalhadores com base no número de fragmentos em seu fluxo do CDC. Um bom ponto de partida é ter um trabalhador por fragmento, mas talvez seja necessário fazer ajustes com base nos requisitos de processamento.

Monitoramento

Use CloudWatch métricas fornecidas pela KCL para monitorar a integridade e o desempenho de seu aplicativo de consumo. As principais métricas incluem latência de processamento, idade do ponto de verificação e contagens de leasing.

Teste

Teste minuciosamente seu aplicativo de consumo, incluindo cenários como falhas de trabalho, refragmentação de streams e condições de carga variáveis.

Usando KCL com linguagens não Java

Embora a KCL seja principalmente uma biblioteca Java, você pode usá-la com outras linguagens de programação por meio do MultiLangDaemon. MultiLangDaemon É um Java-based daemon que gerencia a interação entre seu processador de registros não Java e o KCL.

O KCL fornece suporte para os seguintes idiomas:

- Python
- Ruby
- Node.js
- .NET

Para obter mais informações sobre o uso do KCL com linguagens não Java, consulte a documentação do [MultiLangDaemon KCL](#).

Solução de problemas

Esta seção fornece soluções para problemas comuns que você pode encontrar ao usar a KCL com streams CDC do Amazon Keyspaces.

Processamento lento

Se seu aplicativo de consumidor estiver processando registros lentamente, considere:

- Aumentar o número de instâncias de trabalho
- Otimizando sua lógica de processamento de registros
- Verificação de gargalos em sistemas a jusante

Processamento duplicado

Se você estiver vendo um processamento duplicado de registros, verifique sua lógica de ponto de verificação. Verifique se você está no ponto de verificação após processar os registros com sucesso.

Falhas de trabalhadores

Se os trabalhadores falharem com frequência, verifique:

- Restrições de recursos (CPU, memória)
- Problemas de conectividade de rede

- Problemas com permissões

Problemas com a tabela de locação

Se você estiver enfrentando problemas com a tabela de leasing da KCL:

- Verifique se seu aplicativo tem as permissões apropriadas para acessar a tabela Amazon Keyspaces
- Verifique se a tabela tem taxa de transferência provisionada suficiente

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)

Apache Cassandra-compatible `Murmur3Partitioner`. O `Murmur3Partitioner` é o particionador Cassandra padrão no Amazon Keyspaces e no Cassandra 1.2 e versões posteriores.

RandomPartitioner

Apache Cassandra-compatible `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.

Como alterar o particionador no Amazon Keyspaces

Você pode alterar o particionador usando a Cassandra Query Language (CQL). Console de gerenciamento da AWS

Console de gerenciamento da AWS

Para alterar o particionador usando o console Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS 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.

Client-side timestamps no Amazon Keyspaces

No Amazon Keyspaces, os timestamps do lado do cliente são Cassandra-compatible timestamps que persistem para cada célula em 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.

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.

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). Para obter mais informações, consulte [the section called “Use carimbos de data/hora no lado do cliente em consultas”](#). 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, pois os carimbos não podem ser NULL. Para consultar carimbos de data/hora, você pode usar a função `WRITETIME` em sua instrução DML.

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 obter mais informações sobre os preços do serviço Amazon Keyspaces, consulte os preços do serviço [Amazon Keyspaces \(para Apache Cassandra\)](#).

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 saber 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 “Estimar 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 “Configurar modos read/write de capacidade”](#).

Depois que os carimbos de data/hora do lado do cliente forem ativados para uma tabela, você não poderá desativá-la.

Para saber mais sobre como usar carimbos de data/hora no lado do cliente em consultas, consulte [the section called “Use carimbos de data/hora no lado do cliente em consultas”](#).

Tópicos

- [Como os timestamps do lado do cliente do Amazon Keyspaces se integram aos serviços AWS](#)
- [Crie uma nova tabela com carimbos de data/hora do lado do cliente no Amazon Keyspaces](#)
- [Configure carimbos de data/hora no lado do cliente para uma tabela no Amazon Keyspaces](#)
- [Use carimbos de data/hora do lado do cliente em consultas no Amazon Keyspaces](#)

Como os timestamps do lado do cliente do Amazon Keyspaces se integram aos serviços AWS

A seguinte métrica de timestamps do lado do cliente está disponível na 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 monitorar CloudWatch métricas, consulte [the section called “Monitoramento com CloudWatch”](#).

Ao usar CloudFormation, você pode habilitar timestamps do lado do cliente ao criar uma tabela Amazon Keyspaces. Para obter mais informações, consulte o [Guia do usuário do AWS CloudFormation](#).

Crie uma nova tabela com carimbos de data/hora do lado do cliente no Amazon Keyspaces

Siga esses exemplos para criar uma nova tabela Amazon Keyspaces com timestamps do lado do cliente habilitados usando o Amazon Console de gerenciamento da AWS Keyspaces, o Cassandra Query Language (CQL) ou o AWS Command Line Interface

Console

Crie uma nova tabela com carimbos de data/hora do lado do cliente (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 Client-side carimbos de data/hora.

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 Criar tabela. Sua tabela é criada com os carimbos de data/hora do lado do cliente ativados.

Cassandra Query Language (CQL)

Crie uma nova tabela usando CQL

1. Para criar uma nova tabela com carimbos de data/hora no lado do cliente ativados usando CQL, você pode usar o seguinte exemplo:

```
CREATE TABLE my_keyspace.my_table (  
    userid uuid,  
    time timeuuid,  
    subject text,  
    body text,  
    user inet,  
    PRIMARY KEY (userid, time)  
) WITH CUSTOM_PROPERTIES = {'client_side_timestamps': {'status': 'enabled'}};
```

2. 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'}
```

AWS CLI

Crie uma nova tabela usando o AWS CLI

1. Para criar uma nova tabela com carimbos de data/hora no lado do cliente ativados, você pode usar o seguinte exemplo:

```
./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}]'
```

2. 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-1:111122223333:/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": ""
}
}
```

Configure carimbos de data/hora no lado do cliente para uma tabela no Amazon Keyspaces

Siga esses exemplos para ativar os timestamps do lado do cliente para tabelas existentes usando o Amazon Keyspaces Console de gerenciamento da AWS, o Cassandra Query Language (CQL) ou o AWS Command Line Interface

Console

Para ativar os carimbos de data/hora do lado do cliente para uma tabela existente (console)

1. Faça login no Console de gerenciamento da AWS 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.

Cassandra Query Language (CQL)

Usando uma instrução CQL

1. Ative os carimbos de data/hora do lado do cliente para uma tabela existente com a instrução CQL ALTER TABLE.

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps':  
{'status': 'enabled'}};
```

2. 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'}
```

AWS CLI

Usando o AWS CLI

1. Você pode ativar os timestamps do lado do cliente para uma tabela existente usando o exemplo a AWS CLI seguir.

```
./aws keyspaces update-table \  
--keyspace-name my_keyspace \  
--table-name my_table \  
--client-side-timestamps 'status=ENABLED'
```

2. 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 e o status dos carimbos de data/hora do lado do cliente deve ser ENABLED.

```
{  
  "keyspaceName": "my_keyspace",  
  "tableName": "my_table",  
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/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": ""
}
}

```

Use carimbos de data/hora do lado do cliente em consultas no Amazon Keyspaces

Depois de ativar os carimbos de data/hora do lado do cliente, você pode passar o carimbo de data/hora em suas declarações `INSERT`, `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;
```

Multi-Region replicação para 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 todas as áreas 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 trabalho difícil de resolver conflitos e corrigir problemas de divergência de dados, para que você possa se concentrar em seu aplicativo.

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 de chave multirregionais que replicam suas tabelas em diferentes regiões geográficas 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](#)
- [Configurar a replicação multirregional para Amazon Keyspaces \(para Apache Cassandra\)](#)

Benefícios do uso da replicação multirregional

Multi-Region a replicação 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 em várias regiões, 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.

- High-speed replicação entre regiões — a Multi-Region replicação 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 abordar casos de uso de alto throughput de gravação ou aqueles que apresentam picos ou picos repentinos no throughput sem nenhum impacto no aplicativo.

- Consistência e resolução de conflitos: todas as alterações feitas nos dados em qualquer Região são replicadas para as 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 compatíveis, consulte [the section called “Observações de uso”](#).

Modos de capacidade e preços

Para um espaço de chaves multirregional, você pode usar o modo de capacidade sob demanda ou o modo de capacidade provisionada. Para obter mais informações, consulte [the section called “Configurar modos read/write de capacidade”](#).

No modo sob demanda, você paga 1 unidade de solicitação de gravação (WRU) para gravar até 1 KB de dados por linha, da mesma forma que nas tabelas de uma única região. Mas você é cobrado pelas gravações em cada região do seu keyspace multirregional. Por exemplo, escrever uma linha de 3 KB de dados em um keyspace multirregional com duas regiões requer 6 WRUs: $3 * 2 = 6$ WRUs. Além disso, gravações que incluem dados estáticos e não estáticos exigem operações adicionais de gravação.

No modo provisionado, você paga 1 unidade de capacidade de gravação (WCU) para gravar até 1 KB de dados por linha, da mesma forma que nas tabelas de uma única região. Mas 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 6 WCUs: $3 *$

2 = 6 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 fornece 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](#)
- [Multi-Region resolução de conflitos de replicação](#)
- [Multi-Region recuperação de desastres de replicação](#)
- [Multi-Region a replicação está Regiões da AWS desativada por padrão](#)
- [Multi-Region replicação e integração com recuperação pontual \(PITR\)](#)
- [Multi-Region replicação e integração com serviços AWS](#)

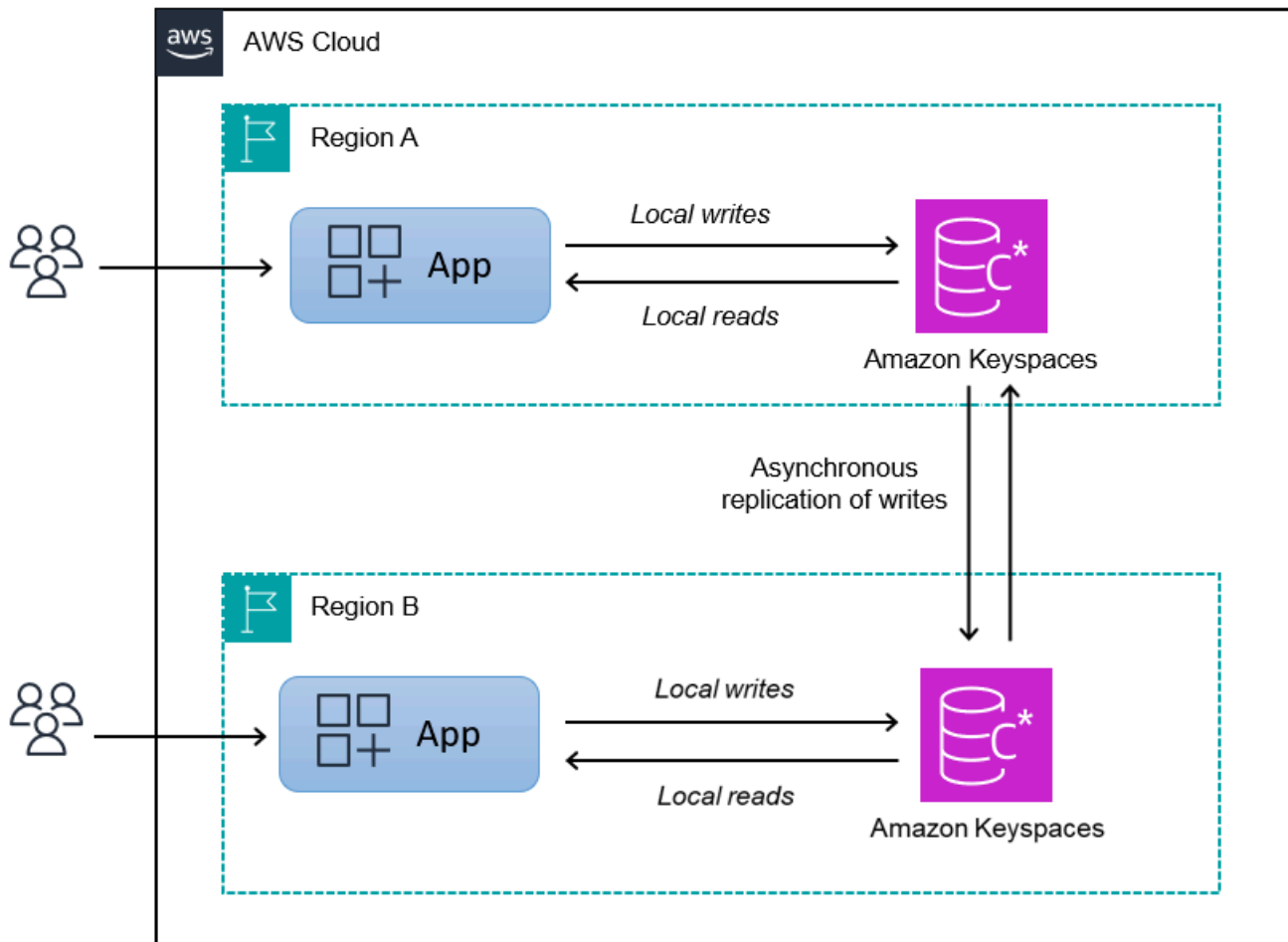
Como a replicação multirregional funciona no Amazon Keyspaces

A replicação multirregional do Amazon Keyspaces implementa uma arquitetura de resiliência de dados que distribui seus dados de forma independente e distribuída geograficamente. 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 keyspace multirregional do Amazon Keyspaces, você pode selecionar 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”](#).



Multi-Region resolução de conflitos de replicação

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 obter mais informações, consulte [the section called “Client-side carimbos de data/hora”](#).

Multi-Region recuperação de desastres de replicação

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 em várias regiões 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 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 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.

Multi-Region a replicação está Regiões da AWS desativada por padrão

A replicação multirregional do Amazon Keyspaces é suportada nas seguintes opções, Regiões da AWS que são desativadas por padrão:

- Região África (Cidade do Cabo)
- Região do Oriente Médio (Emirados Árabes Unidos)
- Região Ásia-Pacífico (Hong Kong)
- Região Oriente Médio (Bahrein)

Antes de usar uma região que está desativada por padrão com a replicação multirregional do Amazon Keyspaces, primeiro você precisa habilitar a região. Para obter mais informações, consulte [Ativar ou desativar Regiões da AWS em sua conta](#) no [Guia AWS Organizations do usuário](#).

Depois de habilitar uma região, você pode criar novos recursos do Amazon Keyspaces na região e adicionar a região a um keyspace multirregional.

Quando você desativa uma região que é usada pela replicação multirregional do Amazon Keyspaces, o Amazon Keyspaces inicia um período de carência de 24 horas. Durante essa janela de tempo, você pode esperar o seguinte comportamento:

- O Amazon Keyspaces continua realizando operações de linguagem de manipulação de dados (DML) em regiões habilitadas.
- O Amazon Keyspaces pausa a replicação de atualizações de dados das regiões habilitadas para a região desativada.
- O Amazon Keyspaces bloqueia todas as solicitações de linguagem de definição de dados (DDL) na região desativada.

Se você desativou a região por engano, poderá reativá-la em 24 horas. Se você reativar a região durante o período de carência de 24 horas, o Amazon Keyspaces tomará as seguintes medidas:

- Retomar automaticamente todas as replicações para a região reativada.
- Replique todas as atualizações de dados que ocorreram nas regiões habilitadas enquanto a região estava desativada para garantir a consistência dos dados.
- Continue com todas as operações adicionais de replicação em várias regiões automaticamente.

Caso a região permaneça desativada após o fechamento da janela de 24 horas, o Amazon Keyspaces toma as seguintes ações para remover permanentemente a região da replicação multirregional:

- Remova a região desativada de todos os espaços de chave de replicação multirregional.
- Converta réplicas de tabelas de replicação multirregional na região desativada em tabelas e espaços chave de região única.
- O Amazon Keyspaces não exclui nenhum recurso da região desativada.

Depois que o Amazon Keyspaces remover permanentemente a região desativada do keyspace multirregional, você não poderá adicionar a região desativada novamente.

Multi-Region replicação e integração com recuperação pontual (PITR)

Point-in-time a recuperação é suportada para 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.
- A PITR deve estar habilitada em todas as réplicas da tabela de origem.

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”](#).

Quando você cria uma tabela multirregional, as configurações de PITR que você define durante o processo de criação são aplicadas automaticamente a todas as tabelas em todas as regiões. Quando você altera as configurações de PITR usando `ALTER TABLE`, o Amazon Keyspaces aplica a atualização somente à tabela local e não às réplicas em outras regiões. Para habilitar a PITR para uma tabela multirregional existente, você precisa repetir a `ALTER TABLE` instrução para todas as réplicas.

Multi-Region replicação e integração com serviços 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, consulte [the 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 qualquer um dos [públicos disponíveis](#) Regiões da AWS. Para obter mais informações sobre [os Regiões da AWS que estão desativados por padrão](#), consulte [the section called “Regiões desativadas por padrão”](#).
- AWS GovCloud (US) Regions e as regiões da China não são suportadas.
- Considere as seguintes soluções alternativas até que os recursos estejam disponíveis:

Configure a vida útil (TTL) ao criar a tabela multirregional. Você não poderá ativar e desativar a TTL nem ajustar o valor da TTL posteriormente. Para obter mais informações, consulte [the section called “Dados expirados com vida útil”](#).

- Para criptografia em repouso, use uma AWS chave própria. No momento, as chaves gerenciadas pelo cliente não são compatíveis com tabelas multirregionais. Para obter mais informações, consulte .

[the section called “Como funciona”](#).

- Você pode usar ALTER KEYSPACE para adicionar uma região a um espaço de chave de região única ou multirregião. Para obter mais informações, consulte [the section called “Adicionar uma região a um keyspace”](#).
 - Antes de adicionar uma região a um espaço de chave de região única, certifique-se de que nenhuma tabela abaixo do espaço de chaves esteja configurada com chaves gerenciadas pelo cliente.
 - Todas as tags existentes configuradas para espaços de chave ou tabelas não são replicadas para a nova região.
- Ao usar o gerenciamento de capacidade provisionada com o ajuste de escala automático do Amazon Keyspaces, certifique-se de usar as operações da 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 obter mais informações, consulte [the section called “Atualize as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#). Para obter mais informações sobre como estimar o throughput da capacidade de gravação de tabelas multirregionais provisionadas, consulte [the section called “Estimar a capacidade de uma tabela multirregional”](#).

- 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` 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(s) Região(ões) replicada(s) 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 chaves 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 obter mais informações, consulte [the section called “Monitoramento com CloudWatch”](#).

Pode ser útil exibir os tempos médios e máximos para determinar o atraso médio e o maior atraso de replicação. Não há SLA sobre essa latência.

- Ao usar uma tabela multirregional no modo sob demanda, você poderá 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 o 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 obter 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 a capacidade de throughput 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 ajuste de escala automático do Amazon Keyspaces ajuda você a provisionar a capacidade de throughput de forma eficiente para workloads variáveis, ajustando automaticamente a capacidade de throughput em resposta ao tráfego real do aplicativo. Para obter mais informações, consulte [the section called “Como o ajuste de escala automático funciona para tabelas multirregionais”](#).

Configurar a replicação multirregional para Amazon Keyspaces (para Apache Cassandra)

Você pode usar o console, o Cassandra Query Language (CQL) ou o AWS Command Line Interface para criar e gerenciar tabelas e espaços chave multirregionais no Amazon Keyspaces.

Esta seção fornece exemplos de como criar e gerenciar espaços de chaves e tabelas multirregionais. Todas as tabelas criadas em um espaço de chaves multirregional herdam automaticamente as configurações multirregionais do espaço de chaves.

Para obter mais informações sobre configurações e atributos compatíveis, consulte [the section called “Observações de uso”](#).

Tópicos

- [Configurar permissões de IAM necessárias para criar e espaços de chaves e tabelas multirregionais](#)
- [Configure as permissões do IAM necessárias para adicionar um Região da AWS a um keyspace](#)
- [Criar um espaço de chaves multirregional no Amazon Keyspaces](#)
- [Adicionar um Região da AWS a um keyspace no Amazon Keyspaces](#)

- [Verifique o progresso da replicação ao adicionar uma nova região a um keyspace](#)
- [Criar uma tabela multirregional com as configurações padrão no Amazon Keyspaces](#)
- [Criar uma tabela multirregional no modo provisionado com ajuste de escala automático no Amazon Keyspaces](#)
- [Atualize a capacidade provisionada e as configurações de ajuste de escala automático para uma tabela multirregional no Amazon Keyspaces](#)
- [Exibir a capacidade provisionada e as configurações de ajuste de escala automático para uma tabela multirregional no Amazon Keyspaces](#)
- [Desativar o ajuste de escala automático para uma tabela no Amazon Keyspaces](#)
- [Definir manualmente a capacidade provisionada de uma tabela multirregional no Amazon Keyspaces](#)

Configurar 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 saber mais sobre a função vinculada ao serviço do , consulte [the section called “Multi-Region Replicação”](#).

Para criar a função vinculada ao serviço exigida pela replicação multirregional, a política do IAM principal 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.

Configure as permissões do IAM necessárias para adicionar um Região da AWS a um keyspace

Para adicionar uma região a um keyspace, o diretor do IAM precisa das seguintes permissões:

- `cassandra:Alter`
- `cassandra:AlterMultiRegionResource`
- `cassandra:Create`
- `cassandra:CreateMultiRegionResource`
- `cassandra:Select`
- `cassandra:SelectMultiRegionResource`
- `cassandra:Modify`
- `cassandra:ModifyMultiRegionResource`

Se a tabela estiver configurada no modo provisionado com o escalonamento automático ativado, as seguintes permissões adicionais serão necessárias.

- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`
- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`

Para adicionar com sucesso uma região a um keyspace de região única, o diretor do IAM também 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 saber mais sobre a função vinculada ao serviço do , consulte [the section called “Multi-Region Replicação”](#).

Para criar a função vinculada ao serviço exigida pela replicação multirregional, a política do IAM principal 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 um principal para adicionar uma região a um keyspace.

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

Criar um espaço de chaves multirregional no Amazon Keyspaces

Esta seção fornece exemplos de como criar um espaço de chaves multirregional. Você pode fazer isso no console do Amazon Keyspaces, usando CQL ou a AWS CLI. Todas as tabelas criadas em um espaço de chaves multirregional herdam automaticamente as configurações multirregionais do 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 “Multi-Region Replicação”](#).

Console

Criar um espaço de chaves multirregional (console)

1. Faça login no Console de gerenciamento da AWS 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 de Multi-Region replicação, você pode adicionar as regiões adicionais que estão disponíveis na lista.
5. Para finalizar, selecione Criar espaço de chaves.

Cassandra Query Language (CQL)

Criar um espaço de chaves multirregional usando o CQL

1. 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.

Todas as tabelas no espaço de chaves herdam a 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.

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' };
```


2. Você pode usar uma instrução CQL para consultar a tabela `tables` no espaço de chaves `system_multiregion_info` para listar programaticamente as Regiões e o status da tabela multirregional especificada. 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 instrução é semelhante ao seguinte:

keyspace_name	table_name	region	status
mykeyspace	mytable	us-east-1	ACTIVE
mykeyspace	mytable	ap-southeast-1	ACTIVE
mykeyspace	mytable	eu-west-1	ACTIVE

CLI

Crie um novo espaço de chave multirregional usando o AWS 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
```

Para criar uma tabela multirregional, consulte [the section called “Criar uma tabela multirregional com as configurações padrão”](#) e [the section called “Criar uma tabela multirregional no modo provisionado”](#).

Adicionar um Região da AWS a um keyspace no Amazon Keyspaces

Você pode adicionar um novo Região da AWS a um espaço de chave que seja um espaço de chave único ou multirregional. A nova região de réplica é aplicada a todas as tabelas no keyspace.

Para alterar um espaço de chave de região única para um espaço de chave de várias regiões, você precisa habilitar os carimbos de data/hora do lado do cliente para todas as tabelas no espaço de chaves. Para obter mais informações, consulte [the section called “Client-side carimbos de data/hora”](#).

Se você estiver adicionando uma região adicional a um keyspace multirregional, o Amazon Keyspaces precisa replicar as tabelas existentes na nova região usando uma restauração única entre regiões para cada tabela existente. As taxas de restauração de cada tabela são cobradas por GB. Para obter mais informações, consulte [Backup e restauração na página](#) de preços do Amazon Keyspaces (para Apache Cassandra). Não há cobrança pela transferência de dados entre regiões para essa operação de restauração. Além dos dados, todas as propriedades da tabela, com exceção das tags, serão replicadas para a nova região.

Você pode usar a `ALTER KEYSPACE` instrução no CQL, o `update-keyspace` comando com o AWS CLI ou o console para adicionar uma nova região a um único keyspace ou a um multirregional no Amazon Keyspaces. Para executar o extrato com sucesso, a conta que você está usando precisa estar localizada em uma das regiões onde o espaço chave já está disponível. Enquanto a réplica está sendo adicionada, você não pode realizar nenhuma outra operação de linguagem de definição de dados (DDL) nos recursos que estão sendo atualizados e replicados.

Para obter mais informações sobre as permissões necessárias para adicionar uma região, consulte [the section called “Configurar permissões do IAM para adicionar região”](#).

Note

Ao adicionar uma região adicional a um keyspace de uma única região, o Amazon Keyspaces cria uma função vinculada ao serviço com o nome em sua conta. `AWSServiceRoleForAmazonKeyspacesReplication` Essa função permite que o Amazon Keyspaces replique tabelas para novas regiões e replique gravações de uma tabela para todas as réplicas de uma tabela multirregional em seu nome. Para saber mais, consulte [the section called “Multi-Region Replicação”](#).

Console

Siga estas etapas para adicionar uma região a um keyspace usando o console do Amazon Keyspaces.

Adicionar uma região a um keyspace (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces e, em seguida, escolha um keyspace na lista.

3. Escolha a guia Regiões da AWS.
4. Na Regiões da AWS guia, escolha Adicionar região.
5. Na caixa de diálogo Adicionar região, escolha a região adicional que você deseja adicionar ao espaço de teclas.
6. Para finalizar, escolha Adicionar.

Cassandra Query Language (CQL)

Adicionar uma região a um keyspace usando CQL

- Para adicionar uma nova região a um keyspace, você pode usar a seguinte declaração. Neste exemplo, o keyspace já está disponível nas regiões Leste dos EUA (Norte da Virgínia) e Oeste dos EUA (Oregon), e a instrução CQL está adicionando a Região Oeste dos EUA (Norte da Califórnia).

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
    'class': 'NetworkTopologyStrategy',
    'us-east-1': '3',
    'us-west-2': '3',
    'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

CLI

Adicione uma região a um espaço de teclas usando a AWS CLI

- Para adicionar uma nova região a um keyspace usando a CLI, você pode usar o exemplo a seguir. Observe que o valor padrão para `client-side-timestamps` é `DISABLED`. Com o `update-keyspace` comando, você deve alterar o valor para `ENABLED`.

```
aws keyspaces update-keyspace \
--keyspace-name my_keyspace \
--replication-specification '{"replicationStrategy": "MULTI_REGION",
"regionList": ["us-east-1", "eu-west-1", "eu-west-3"] }' \
--client-side-timestamps '{"status": "ENABLED"}'
```

Verifique o progresso da replicação ao adicionar uma nova região a um keyspace

Adicionar uma nova região a um keyspace do Amazon Keyspaces é uma operação de longa duração. Para monitorar o progresso, você pode usar as consultas mostradas nesta seção.

Cassandra Query Language (CQL)

Usando o CQL para verificar o progresso da adição da região

- Para verificar o progresso da criação das novas réplicas de tabela em um determinado espaço de chave, você pode consultar a `system_multiregion_info.keyspaces` tabela. A seguinte instrução CQL é um exemplo disso.

```
SELECT keyspace_name, region, status, tables_replication_progress
FROM system_multiregion_info.keyspaces
WHERE keyspace_name = 'my_keyspace';
```

Enquanto uma operação de replicação está em andamento, o status mostra o progresso da criação da tabela na nova região. Este é um exemplo em que 5 em cada 10 tabelas foram replicadas para a nova região.

keyspace_name	region	status	tables_replication_progress
my_keyspace	us-east-1	Updating	
my_keyspace	us-west-2	Updating	
my_keyspace	eu-west-1	Creating	50%

Depois que o processo de replicação for concluído com êxito, a saída deverá ser semelhante a este exemplo.

keyspace_name	region	status
my_keyspace	us-east-1	Active
my_keyspace	us-west-2	Active
my_keyspace	eu-west-1	Active

CLI

Usando o AWS CLI para verificar o progresso da adição da região

- Para confirmar o status da criação da réplica da tabela para um determinado espaço de chave, você pode usar o exemplo a seguir.

```
aws keyspaces get-keyspace \
--keyspace-name my_keyspace
```

A saída deve ser semelhante a este exemplo:

```
{
  "keyspaceName": "my_keyspace",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "replicationStrategy": "MULTI_REGION",
  "replicationRegions": [
    "us-east-1",
    "eu-west-1"
  ]
  "replicationGroupStatus": [
    {
      "RegionName": "us-east-1",
      "KeyspaceStatus": "Active"
    },
    {
      "RegionName": "eu-west-1",
      "KeyspaceStatus": "Creating",
      "TablesReplicationProgress": "50.0%"
    }
  ]
}
```

Criar uma tabela multirregional com as configurações padrão no Amazon Keyspaces

Esta seção fornece exemplos de como criar uma tabela multirregional no modo sob demanda com todas as configurações padrão. Você pode fazer isso no console do Amazon Keyspaces, usando CQL ou a AWS CLI. Todas as tabelas criadas em um espaço de chaves multirregional herdam automaticamente as configurações multirregionais do espaço de chaves.

Para criar um espaço de chaves multirregional, consulte [the section called “Criar um espaço de chaves multirregional”](#).

Console

Criar uma tabela multirregional com as configurações padrão (console)

1. Faça login no Console de gerenciamento da AWS 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 On-demand. Para obter mais informações sobre como configurar o modo provisionado, consulte [the section called “Criar uma tabela multirregional no modo provisionado”](#).
 - Gerenciamento de chaves de criptografia: apenas a opção Chave pertencente à AWS é compatível.
 - Client-side timestamps — Esse recurso 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.

Cassandra Query Language (CQL)

Criar uma tabela multirregional no modo sob demanda com configurações padrão

- Para criar uma tabela multirregional com as configurações padrão, você pode usar a instrução CQL 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'
    }
  };
```

CLI

Usando o AWS CLI

1. Para criar uma tabela multirregional com configurações padrão, você só precisa especificar o esquema. Você pode usar o seguinte exemplo.

```
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"
}
```

2. Para confirmar as configurações da tabela, você pode usar a instrução a seguir.

```
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
    }
  }
]
}
```

Criar uma tabela multirregional no modo provisionado com ajuste de escala automático no Amazon Keyspaces

Esta seção fornece exemplos de como criar uma tabela multirregional no modo provisionado com ajuste de escala automático. Você pode fazer isso no console do Amazon Keyspaces, usando CQL ou a AWS CLI.

Para obter mais informações sobre as configurações suportadas e os recursos de replicação multirregional, consulte [the section called “Observações de uso”](#)

Para criar um espaço de chaves multirregional, consulte [the section called “Criar um espaço de chaves multirregional”](#).

Ao criar uma nova tabela multirregional no modo provisionado com configurações de ajuste de escala automático, você pode especificar as configurações gerais da tabela que são válidas para todas as Regiões da AWS em que a tabela é replicada. Em seguida, você pode sobrescrever as configurações de capacidade de leitura e ler as configurações de ajuste de escala automático para cada réplica. No entanto, a capacidade de gravação permanece sincronizada entre todas as réplicas para garantir que haja capacidade suficiente para replicar gravações em todas as regiões.

 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 obter mais informações, consulte [the section called “Uso de perfis vinculados ao serviço”](#).

Console

Criar uma tabela multirregional com ajuste de escala automático habilitado

1. Faça login no Console de gerenciamento da AWS 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 clustering opcionais.
7. Na seção Configurações da tabela, selecione Personalizar configurações.
8. Continue com as configurações Read/write de capacidade.
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 todas as Regiões da AWS em 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 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.
- Desmarque a caixa de seleção Escalar 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 o recurso de ajuste de escala automático 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 para a tabela. As unidades de capacidade de gravação permanecem sincronizadas em todas as Regiões da AWS para garantir que haja capacidade suficiente para replicar eventos de gravação em todas as regiões.
 - Desmarque Escalar 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 o recurso de ajuste de escala automático do Amazon Keyspaces para tabelas multirregionais provisionadas.

12. Escolha Criar tabela. Sua tabela é criada com os parâmetros padrão de ajuste de escala automático.

Cassandra Query Language (CQL)

Criar uma tabela multirregional com modo de capacidade provisionada e ajuste de escala automático usando CQL

- Para criar uma tabela multirregional no modo provisionado com ajuste de escala automático, primeiro você deve especificar o modo de capacidade definindo `CUSTOM_PROPERTIES` para a tabela. Depois de especificar o modo de capacidade provisionada, você pode definir as configurações do ajuste de escala automático para a tabela usando `AUTOSCALING_SETTINGS`.

Para obter informações detalhadas sobre as configurações do ajuste de escala automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consulte [the section called “Crie uma nova tabela com dimensionamento automático”](#).

Para definir a capacidade de leitura de uma réplica de tabela em uma região específica, você pode configurar os seguintes parâmetros como parte do `replica_updates` da tabela:

- A região
- As unidades de capacidade de leitura provisionadas (opcional)
- Configurações do ajuste de escala automático para a capacidade de leitura (opcional)

O exemplo a seguir mostra uma instrução `CREATE TABLE` para uma tabela multirregional no modo provisionado. As configurações gerais do ajuste de escala automático da capacidade de gravação e leitura são as mesmas. No entanto, as configurações do ajuste de escala automático de leitura especificam períodos de espera adicionais de 60 segundos antes de aumentar ou diminuir a capacidade de leitura da tabela. Além disso, as configurações do ajuste de escala automático da capacidade de leitura para a Região Leste dos EUA (N. 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
                    }
                }
            }
        }
    }
}
```

```
}  
};
```

CLI

Crie uma nova tabela multirregional no modo provisionado com escalonamento automático usando o AWS CLI

- Para criar uma tabela multirregional no modo provisionado com configuração de ajuste de escala automático, você pode usar a AWS CLI. Observe que você deve usar o comando `create-table` da CLI do Amazon Keyspaces para definir configurações de ajuste de escala automático multirregionais. Isso ocorre porque o Application Auto Scaling, o serviço que o Amazon Keyspaces usa para realizar o ajuste de escala automático em seu nome, não oferece suporte a várias regiões.

Para obter mais informações sobre as configurações de ajuste de escala automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consulte [the section called “Crie uma nova tabela com dimensionamento automático”](#).

Para definir a capacidade de leitura de uma réplica de tabela em uma região específica, você pode configurar os seguintes parâmetros como parte do `replicaSpecifications` da tabela:

- A região
- As unidades de capacidade de leitura provisionadas (opcional)
- Configurações do ajuste de escala automático para a capacidade de leitura (opcional)

Ao criar tabelas multirregionais provisionadas com configurações complexas de ajuste de escala automático e configurações diferentes para réplicas de tabela, é útil carregar as configurações de ajuste de escala 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](#) e extrair `auto-scaling.json` 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
```

Atualize a capacidade provisionada e as configurações de ajuste de escala automático para uma tabela multirregional no Amazon Keyspaces

Esta seção 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 de ajuste de escala automático e como elas funcionam, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).

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 ajuste de escala automático. Isso ocorre porque as operações subjacentes da API Application Auto Scaling não são. Region-aware

Para obter mais informações sobre como estimar o throughput da capacidade de gravação de tabelas multirregionais provisionadas, consulte [the section called “Estimar a capacidade de uma tabela multirregional”](#).

Para obter mais informações sobre a API do Amazon Keyspaces, consulte [Referência da API do Amazon Keyspaces](#).

Ao atualizar o modo provisionado ou as configurações de ajuste de escala automático de uma tabela multirregional, você pode atualizar as configurações de capacidade de leitura e a configuração de ajuste de escala automático de leitura para cada réplica da tabela.

No entanto, a capacidade de gravação permanece sincronizada entre todas as réplicas para garantir que haja capacidade suficiente para replicar gravações em todas as regiões.

Cassandra Query Language (CQL)

Atualize a capacidade provisionada e as configurações de ajuste de escala automático de uma tabela multirregional usando CQL

- Você pode usar `ALTER TABLE` para atualizar o modo de capacidade e as configurações de ajuste de escala automático de uma tabela existente. Se estiver atualizando uma tabela que está atualmente no modo de capacidade sob demanda, é necessário usar `capacity_mode`. Se sua tabela já estiver no modo de capacidade provisionada, esse campo poderá ser omitido.

Para obter informações detalhadas sobre as configurações do ajuste de escala automático, a política de rastreamento de metas, o valor alvo e as configurações opcionais, consulte [the section called “Crie uma nova tabela com dimensionamento automático”](#).

Na mesma declaração, você também pode atualizar as configurações de capacidade de leitura e ajuste de escala automático de réplicas de tabela em regiões específicas atualizando a propriedade `replica_updates` da 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
                    }
                }
            }
        }
    }
}
};

```

CLI

Atualize a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional usando o AWS 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 da CLI do Amazon Keyspaces para criar ou modificar configurações de ajuste de escala automático multirregional. Isso ocorre porque o Application Auto Scaling, o serviço que o Amazon Keyspaces usa para realizar o ajuste de escala automático da capacidade da tabela em seu nome, não suporta várias Regiões da AWS.

Para atualizar a capacidade de leitura de uma réplica de tabela em uma região específica, você pode alterar um dos seguintes parâmetros opcionais do `replicaSpecifications` da tabela:

- As unidades de capacidade de leitura provisionadas (opcional)

- Configurações do ajuste de escala automático para a capacidade de leitura (opcional)

Ao atualizar tabelas multirregionais com configurações complexas de ajuste de escala automático e configurações diferentes para réplicas de tabela, é útil carregar as configurações de ajuste de escala 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](#) e extrair `auto-scaling.json` 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
```

Exibir a capacidade provisionada e as configurações de ajuste de escala automático para uma tabela multirregional no Amazon Keyspaces

Você pode exibir a capacidade provisionada e as configurações de ajuste de escala automático de uma tabela multirregional no console do Amazon Keyspaces, usando CQL ou a AWS CLI. Esta seção fornece exemplos de como fazer isso usando CQL e o AWS CLI

Cassandra Query Language (CQL)

Exibir a capacidade provisionada e as configurações de ajuste de escala automático de uma tabela multirregional usando CQL

- Para ver a configuração do ajuste de escala 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}}}
```

CLI

Visualize a capacidade provisionada e as configurações de escalabilidade automática de uma tabela multirregional usando o AWS CLI

- Para exibir a configuração de ajuste de escala automático de uma tabela multirregional, você pode usar a operação `get-table-auto-scaling-settings`. 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
                }
            }
        }
    }
}
```

Desativar o ajuste de escala automático para uma tabela no Amazon Keyspaces

Esta seção fornece exemplos de como desativar o ajuste de escala automático para uma tabela multirregional no modo de capacidade provisionada. Você pode fazer isso no console do Amazon Keyspaces, usando CQL ou a AWS CLI.

Important

Recomendamos usar o ajuste de escala automático para tabelas multirregionais que usam o modo de capacidade provisionada. Para obter mais informações, consulte [the section called “Estimar a capacidade de uma tabela multirregional”](#).

Note

Para excluir a função vinculada ao serviço usada pelo Application Auto Scaling, você deve desativar o ajuste de escala automático em todas as tabelas da conta em todas as Regiões da AWS.

Console

Desativar o ajuste de escala automático do Amazon Keyspaces para uma tabela multirregional existente no console

1. Faça login no Console de gerenciamento da AWS 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 depois escolha a guia Capacidade.
3. Na seção Configurações de capacidade, escolha Editar.
4. Para desativar o ajuste de escala automático do Amazon Keyspaces, desmarque a opção Escalar automaticamente. A desativação do ajuste de escala 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”](#).
5. Quando as configurações de escala automática estiverem definidas, escolha Salvar.

Cassandra Query Language (CQL)

Desative o ajuste de escala automático para uma tabela multirregional usando CQL

- Você pode usar ALTER TABLE para desativar o ajuste de escala automático de uma tabela existente. Observe que você não pode desativar o ajuste de escala automático para uma réplica de tabela individual.

No exemplo a seguir, o ajuste de escala 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
    }
};
```

CLI

Desative o escalonamento automático para uma tabela multirregional usando o AWS 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 ajuste de escala automático para uma réplica de tabela individual.

No exemplo a seguir, o ajuste de escala 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}
```

Definir manualmente a capacidade provisionada de uma tabela multirregional no Amazon Keyspaces

Se precisar desativar o ajuste de escala automático de uma tabela multirregional, você pode provisionar manualmente a capacidade de leitura para uma tabela de réplica usando CQL ou a AWS CLI.

Note

Recomendamos usar o ajuste de escala automático para tabelas multirregionais que usam o modo de capacidade provisionada. Para obter mais informações, consulte [the section called “Estimar a capacidade de uma tabela multirregional”](#).

Cassandra Query Language (CQL)

Configurar a capacidade provisionada de uma tabela multirregional manualmente usando o CQL

- Você pode usar `ALTER TABLE` para 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  
        }  
    }  
};
```

CLI

Defina manualmente a capacidade provisionada de uma tabela multirregional usando o AWS CLI

- Se você precisar desativar o ajuste de escala automático de uma tabela multirregional, pode usar o `update-table` para 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
```

Backup e restauração de dados com a recuperação para um ponto no tempo para o Amazon Keyspaces

Point-in-time a recuperação (PITR) ajuda a proteger suas tabelas do Amazon Keyspaces contra operações acidentais de gravação ou exclusão, fornecendo backups contínuos dos dados da tabela.

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 o PITR foi habilitado 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 determinado momento usando o console, o AWS SDK e o () ou a Cassandra Query Language AWS Command Line Interface (CQL AWS CLI). Para obter mais informações, consulte [Use a recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces](#).

Point-in-time as operações não têm impacto no desempenho ou na disponibilidade da tabela base, e a restauração de uma tabela não consome taxa de transferência adicional.

Para obter informações sobre cotas PITR, consulte [Cotas](#).

Para obter informações sobre preços, consulte o [Amazon Keyspaces \(para Apache Cassandra\)](#).

Tópicos

- [Como a recuperação para um ponto no tempo funciona no Amazon Keyspaces](#)
- [Use a recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces](#)

Como a recuperação para um ponto no tempo funciona no Amazon Keyspaces

Esta seção fornece uma visão geral de como funciona a recuperação para um ponto no tempo (PITR) do Amazon Keyspaces. Para obter mais informações sobre preços, consulte o [Amazon Keyspaces \(para Apache Cassandra\)](#).

Tópicos

- [Janela de tempo para backups contínuos do PITR](#)
- [Configurações de restauração de PITR](#)
- [Restauração PITR de tabelas criptografadas](#)
- [Restauração PITR de tabelas multirregionais](#)
- [Restauração PITR de tabelas com tipos definidos pelo usuário \(UDTs\)](#)
- [Tempo de restauração da tabela com PITR](#)
- [Amazon Keyspaces PITR e integração com serviços AWS](#)

Janela de tempo para backups contínuos do 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 o PITR foi ativado, 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 o PITR está habilitado, 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 o PITR estava habilitado.

Se você desabilitar o PITR e depois reabilitá-lo novamente, redefinirá a hora de início do primeiro backup disponível para quando o PITR foi reabilitado. Isso significa que a desabilitação do 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 o PITR estiver habilitado 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 o 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. O PITR não sobrescreve as tabelas existentes.

Além do esquema e dos dados da tabela, o 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 deseja sobrescrever essas configurações durante a restauração, poderá fazer isso usando `WITH custom_properties`. As propriedades personalizadas incluem as configurações a seguir.

- Read/write modo de capacidade
- Configurações de capacidade de throughput provisionada
- Configurações de PITR

Se a tabela estiver no modo de capacidade provisionada com o ajuste de escala automático ativado, a operação de restauração também restaurará as configurações de ajuste de escala automático da tabela. Você pode sobrescrevê-los usando o parâmetro `autoscaling_settings` em CQL ou `autoScalingSpecification` com a CLI. Para obter mais informações sobre as configurações de ajuste de escala automático, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).

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.

- Amazon Keyspaces altera fluxos de captura de dados (CDC)
- AWS Identity and Access Management Políticas (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 o PITR, o Amazon Keyspaces restaura as configurações de criptografia da tabela de origem. Se a tabela foi criptografada com um 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 gerenciada pelo cliente 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 [Solução de problemas de acesso por chave](#) no Guia do AWS Key Management Service desenvolvedor.

Restauração PITR de tabelas multirregionais

Você pode restaurar uma tabela multirregional usando o PITR. Para que a operação de restauração seja bem-sucedida, a PITR deve estar habilitada em todas as réplicas da tabela de origem e as tabelas de origem e de destino devem 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 espaço de chaves. 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 obter mais informações sobre a replicação multirregional, consulte [the section called “Como funciona”](#)

Restauração PITR de tabelas com tipos definidos pelo usuário (UDTs)

Você pode restaurar uma tabela que usa UDTs. Para que a operação de restauração seja bem-sucedida, os UDTs referenciados precisam existir e ser válidos no keyspace.

Se algum UDT necessário estiver ausente quando você tentar restaurar uma tabela, o Amazon Keyspaces tentará restaurar o esquema do UDT automaticamente e, em seguida, continuará restaurando a tabela.

Se você removeu e recriou o UDT, o Amazon Keyspaces restaura o UDT com o novo esquema do UDT e rejeita a solicitação de restauração da tabela usando o esquema original do UDT. Nesse caso, se você quiser restaurar a tabela com o esquema UDT antigo, poderá restaurar a tabela em um novo espaço de teclas. Quando você exclui e recria uma UDT, mesmo que o esquema da UDT recriada seja igual ao esquema da UDT excluída, a UDT recriada é considerada uma nova UDT. Nesse caso, o Amazon Keyspaces rejeita a solicitação para restaurar a tabela com o esquema antigo do UDT.

Se o UDT estiver ausente e o Amazon Keyspaces tentar restaurar o UDT, a tentativa falhará se você atingir o número máximo de UDTs para a conta na região.

Para obter mais informações sobre cotas de UDT e valores padrão, consulte [the section called “Cotas e valores padrão para tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces”](#) Para obter mais informações sobre como trabalhar com UDTs, consulte [the section called “User-defined tipos \(UDTs\)”](#).

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.

Amazon Keyspaces PITR e integração com serviços AWS

As seguintes operações de PITR são registradas usando AWS CloudTrail para permitir monitoramento e auditoria contínuos.

- Crie uma nova tabela com o PITR habilitado ou desabilitado.
- Habilite ou desabilite o PITR em uma tabela existente.
- Restaure uma tabela ativa ou excluída.

Para obter mais informações, consulte [Registro de chamadas de API do Amazon Keyspaces com AWS CloudTrail](#).

Você pode realizar as seguintes ações usando CloudFormation.

- Crie uma nova tabela com o PITR habilitado ou desabilitado.
- Habilite ou desabilite o PITR em uma tabela existente.

Para obter mais informações, consulte [Referência de tipos de recursos do Cassandra](#) no [CloudFormation Guia do usuário](#).

Use a recuperação para um ponto no tempo (PITR) do Amazon Keyspaces

Com o Amazon Keyspaces (para Apache Cassandra), você pode restaurar tabelas em um momento específico usando Point-in-Time Restore (PITR). A PITR permite que você restaure uma tabela para um estado anterior nos últimos 35 dias, fornecendo recursos de proteção e recuperação de dados. Esse recurso é valioso em casos como exclusão acidental de dados, erros de aplicativos ou para teste. Você pode recuperar dados de forma rápida e eficiente, minimizando o tempo de inatividade e a perda de dados. As seções a seguir orientam você pelo processo de restauração de tabelas usando PITR no Amazon Keyspaces, garantindo a integridade dos dados e a continuidade dos negócios.

Tópicos

- [Configurar permissões do IAM da tabela de restauração para o PITR do Amazon Keyspaces](#)
- [Configurar PITR para uma tabela no Amazon Keyspaces](#)
- [Desativar a PITR para uma tabela do Amazon Keyspaces](#)
- [Restaurar uma tabela do backup para um determinado ponto no tempo no Amazon Keyspaces](#)
- [Restaurar uma tabela excluída usando a PITR do Amazon Keyspaces](#)

Configurar permissões do IAM da tabela de restauração para o PITR do Amazon Keyspaces

Esta seção resume como configurar permissões para que um diretor AWS Identity and Access Management (IAM) restaure tabelas do Amazon Keyspaces. No IAM, a política gerenciada pela AWS AmazonKeyspacesFullAccess inclui as permissões para restaurar tabelas do Amazon Keyspaces. Para implementar uma política personalizada com o mínimo de permissões necessárias, considere os requisitos descritos na próxima seção.

Para restaurar uma tabela com sucesso, a entidade principal 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.

Este é 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 obter 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 da VPC e recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces”](#).

Configurar PITR para uma tabela no Amazon Keyspaces

Você pode configurar uma tabela no Amazon Keyspaces para operações de backup e restauração usando PITR com o console, o CQL e a AWS CLI.

Ao criar uma nova tabela usando CQL ou o AWS CLI, você deve habilitar explicitamente a PITR na instrução `create table`. Quando você cria uma tabela usando o console, a PITR será ativada por padrão.

Para saber como restaurar uma tabela, consulte [the section called “Restaurar uma tabela para um ponto no tempo”](#).

Console

Configurar a PITR para uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS 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 recuperação pontual, selecione Ativar Point-in-time recuperação.
5. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Configurar a PITR para uma tabela usando CQL

1. Você pode gerenciar as configurações de PITR para tabelas usando a propriedade customizada `point_in_time_recovery`.

Para ativar a PITR ao criar uma tabela, você deve definir o status `point_in_time_recovery` para `enabled`. Você pode usar o seguinte comando CQL 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 customizada de recuperação para um ponto no tempo for especificado, a recuperação para um ponto no tempo estará desabilitada por padrão.

2. Para habilitar o 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'}}
```

CLI

Configure o PITR para uma tabela usando o AWS CLI

1. Você pode gerenciar as configurações de PITR para tabelas usando a API UpdateTable.

Para ativar a PITR ao criar uma tabela, você deve incluir `point-in-time-recovery 'status=ENABLED'` no comando de criação de tabela. Você pode usar o seguinte comando AWS CLI 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 de recuperação para um ponto no tempo for especificado, a recuperação para um ponto no tempo estará desabilitada por padrão.

2. Para confirmar a configuração de recuperação pontual de uma tabela, você pode usar o comando a seguir AWS CLI .

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

3. Para habilitar o PITR 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'
```

Desativar a PITR para uma tabela do Amazon Keyspaces

Você pode desativar a PITR para uma tabela do Amazon Keyspaces a qualquer momento usando o console, o CQL ou a AWS CLI.

 Important

Desabilitar o PITR exclui seu histórico de backup imediatamente, mesmo se você reabilitar a PITR na tabela em 35 dias.

Para saber como restaurar uma tabela, consulte [the section called “Restaurar uma tabela para um ponto no tempo”](#).

Console

Desativar a PITR para uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS 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 recuperação pontual, desmarque a caixa de seleção Ativar Point-in-time recuperação.
5. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Desativar a PITR para uma tabela usando CQL

- Para desabilitar a PITR para uma tabela existente, execute o comando CQL a seguir.

```
ALTER TABLE mykeyspace.mytable  
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Desative a PITR para uma tabela usando a AWS CLI

- Para desativar a PITR para uma tabela existente, execute o AWS CLI comando a seguir.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'  
--point-in-time-recovery 'status=DISABLED'
```

Restaurar uma tabela do backup para um determinado ponto no tempo no Amazon Keyspaces

A seção a seguir demonstra como restaurar uma tabela existente do Amazon Keyspaces para um ponto específico no tempo.

Note

Esse procedimento supõe que a tabela que você está usando foi configurada com a recuperação para um ponto no tempo. Para habilitar a PITR para uma tabela, consulte [the section called “Configurar PITR”](#).

 Important

Enquanto uma restauração estiver em andamento, não modifique nem exclua as políticas AWS Identity and Access Management (IAM) que concedem permissão ao principal do IAM (por exemplo, usuário, grupo ou função) para realizar a restauração. Do contrário, pode haver um comportamento inesperado. Por exemplo, se você remover permissões de gravação de uma tabela enquanto ela estiver sendo restaurada, 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.

Console

Restaurar uma tabela para um ponto específico no tempo usando o console

1. Faça login no Console de gerenciamento da AWS 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 aquela que você quer restaurar.
4. Na guia Backups da tabela, na seção Point-in-time Recuperação, escolha Restaurar.
5. Para o nome da nova tabela, insira um nome para a tabela restaurada, por exemplo **mytable_restored**.
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 restaurada mudará para Ativo.

Cassandra Query Language (CQL)

Restaurar uma tabela para um ponto no tempo usando 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 para um ponto no tempo está habilitada para a tabela, consulte o `system_schema_mcs.tables` como mostrado neste exemplo.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Point-in-time a recuperação é ativada 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 a hora atual. Quando você omite a cláusula `WITH restore_timestamp = ...`, é usado o carimbo de data/hora atual.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

- Você também pode restaurar para um ponto específico no tempo, definido por um `restore_timestamp` no formato ISO 8601. 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.

3. 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 mudará para Active (Ativo).

CLI

Restaure uma tabela para um determinado momento usando o AWS CLI

1. Criar 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 o 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-1:111122223333:/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": ""
  }
}

```

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-1:111122223333:/keyspace/myKeyspace/table/myTable_restored"
}

```

- Para restaurar a tabela para a hora atual, você pode omitir o parâmetro `restore-timestamp`.

```

aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name 'myTable_restored1'

```

Restaurar uma tabela excluída usando a PITR do Amazon Keyspaces

O procedimento a seguir mostra como restaurar uma tabela excluída do backup para o momento da exclusão. Você pode fazer isso usando CQL ou o AWS CLI

 Note

Esse procedimento pressupõe que o PITR foi ativado na tabela excluída.

Cassandra Query Language (CQL)

Restaurar uma tabela excluída usando o CQL

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;
```

CLI

Restaurar uma tabela excluída usando o AWS CLI

1. Excluir uma tabela criada anteriormente com a PITR habilitada. O comando a seguir é um exemplo.

```
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-1:111122223333:/keyspace/myKeyspace/table/myTable_restored2"
}
```

Dados expirados com vida útil (TTL) para Amazon Keyspaces (para Apache Cassandra)

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 o TTL em um AdTech aplicativo 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.

Você pode definir, modificar ou desativar as configurações de TTL padrão para tabelas novas e existentes usando o console, o Cassandra Query Language (CQL) ou a AWS CLI.

Em tabelas com TTL padrão configurada, você pode usar instruções do CQL para substituir as configurações de TTL padrão da tabela e aplicar valores de TTL personalizados a linhas e colunas. Para obter mais informações, consulte [the section called “Usar INSERT para definir TTL personalizada para novas linhas”](#) e [the section called “Usar UPDATE para definir TTL personalizada para linhas e colunas”](#).

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 de `delete`s. 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

- [Amazon Keyspaces Time to Live e integração com serviços AWS](#)
- [Criar uma tabela com as configurações de vida útil \(TTL\) padrão](#)

- [Atualizar o valor padrão de vida útil \(TTL\) de uma tabela](#)
- [Criar tabela com as configurações de tempo de vida \(TTL\) personalizadas habilitadas](#)
- [Atualizar tabela com vida útil \(TTL\) personalizada](#)
- [Usar a instrução INSERT para definir valores de tempo de vida \(TTL\) personalizados para novas linhas](#)
- [Usar a instrução UPDATE para editar as configurações de vida útil \(TTL\) personalizadas para linhas e colunas](#)

Amazon Keyspaces Time to Live e integração com serviços AWS

A métrica TTL a seguir está disponível na 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 CloudWatch métricas, consulte [the section called “Monitoramento com CloudWatch”](#).

Ao usar CloudFormation, você pode ativar o TTL ao criar uma tabela do Amazon Keyspaces. Para obter mais informações, consulte o [Guia do usuário do AWS CloudFormation](#).

Criar uma tabela com as configurações de vida útil (TTL) padrão

No Amazon Keyspaces, você pode definir um valor de TTL padrão para todas as linhas em uma tabela quando a tabela é criada.

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.

Os valores de TTL são definidos em segundos e o valor máximo configurável é 630.720.000 segundos, o que equivale a 20 anos.

Após a criação da tabela, você pode substituir a configuração de TTL padrão da tabela para linhas ou colunas específicas com instruções de DML CQL. Para obter mais informações, consulte [the section called “Usar INSERT para definir TTL personalizada para novas linhas”](#) e [the section called “Usar UPDATE para definir TTL personalizada para linhas e colunas”](#).

Quando você ativa o TTL em uma tabela, o Amazon Keyspaces começa a armazenar metadados TTL-related adicionais 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.

Os exemplos a seguir mostram como criar uma tabela com um valor de TTL padrão.

Console

Crie uma tabela com um valor de vida útil padrão usando o console.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 Criar tabela. Sua tabela é criada com o valor de TTL padrão especificado.

Cassandra Query Language (CQL)

Criar uma tabela com um valor de TTL padrão usando CQL

1. A instrução a seguir cria uma tabela com o valor de TTL padrão definido para 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;
```

2. 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`.

```
DESC TABLE my_table;
```

```
CREATE TABLE my_keyspace.my_table (  
    userid uuid,  
    time timeuuid,  
    body text,  
    subject text,  
    user inet,  
    PRIMARY KEY (userid, time)  
    ) WITH CLUSTERING ORDER BY (time ASC)  
    AND bloom_filter_fp_chance = 0.01  
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}  
    AND comment = ''  
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}  
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}  
    AND crc_check_chance = 1.0  
    AND dclocal_read_repair_chance = 0.0  
    AND default_time_to_live = 3024000  
    AND gc_grace_seconds = 7776000  
    AND max_index_interval = 2048  
    AND memtable_flush_period_in_ms = 3600000  
    AND min_index_interval = 128  
    AND read_repair_chance = 0.0  
    AND speculative_retry = '99PERCENTILE';
```

CLI

Crie uma nova tabela com um valor TTL padrão usando a AWS CLI

1. Você pode usar o seguinte comando para criar uma tabela com o valor de TTL padrão definido para um ano.

```
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}]' \
    --default-time-to-live '31536000'
```

2. Para confirmar o status da TTL da tabela, você pode usar o seguinte comando.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

A saída do comando é semelhante ao exemplo a seguir

```
{
  "keyspaceName": "myKeyspace",
  "tableName": "myTable",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
  "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": "2024-09-02T10:52:22.190000+00:00"
  },
  "encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
    "status": "DISABLED"
  },
  "ttl": {
    "status": "ENABLED"
  },
  "defaultTimeToLive": 31536000,
  "comment": {
    "message": ""
  },
  "replicaSpecifications": []
}

```

Atualizar o valor padrão de vida útil (TTL) de uma tabela

É possível atualizar uma tabela existente com um novo valor de TTL padrão. Os valores de TTL são definidos em segundos e o valor máximo configurável é 630.720.000 segundos, o que equivale a 20 anos.

Quando você ativa o TTL em uma tabela, o Amazon Keyspaces começa a armazenar metadados TTL-related adicionais 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 a TTL for habilitada para uma tabela, você poderá substituir a configuração de TTL padrão da tabela para linhas ou colunas específicas com instruções DML CQL. Para obter mais informações, consulte [the section called “Usar INSERT para definir TTL personalizada para novas linhas”](#) e [the section called “Usar UPDATE para definir TTL personalizada para linhas e colunas”](#).

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.

Siga estas etapas para atualizar as configurações de vida útil padrão para tabelas existentes usando o console, CQL ou a AWS CLI.

Console

Atualizar o valor de TTL padrão de uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS 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, 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.

Cassandra Query Language (CQL)

Atualizar o valor de TTL padrão de uma tabela usando CQL

1. Você pode usar `ALTER TABLE` para editar as configurações de tempo de vida (TTL) padrão de uma tabela. Para atualizar as configurações de TTL padrão da tabela para 2.592.000 segundos, o que representa 30 dias, você pode usar a seguinte instrução.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

2. 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`.

```
DESC TABLE my_table;
```

A saída da instrução deve ser semelhante a este exemplo.

```
CREATE TABLE my_keyspace.my_table (  
    id int PRIMARY KEY,  
    date timestamp,  
    name text  
) WITH bloom_filter_fp_chance = 0.01  
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}  
    AND comment = ''  
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}  
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}  
    AND crc_check_chance = 1.0  
    AND dclocal_read_repair_chance = 0.0  
    AND default_time_to_live = 2592000  
    AND gc_grace_seconds = 7776000  
    AND max_index_interval = 2048  
    AND memtable_flush_period_in_ms = 3600000  
    AND min_index_interval = 128  
    AND read_repair_chance = 0.0  
    AND speculative_retry = '99PERCENTILE';
```

CLI

Atualize o valor TTL padrão de uma tabela usando a AWS CLI

1. Você pode usar `update-table` para editar o valor de TTL padrão de uma tabela. Para atualizar as configurações de TTL padrão da tabela para 2.592.000 segundos, o que representa 30 dias, você pode usar a seguinte instrução.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'  
--default-time-to-live '2592000'
```

2. Para confirmar o valor de TTL padrão atualizado, você pode usar a instrução a seguir.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

A saída da instrução será semelhante à do exemplo a seguir.

```
{  
    "keyspaceName": "myKeyspace",  
    "tableName": "myTable",
```

```

    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
    "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": "2024-09-02T10:52:22.190000+00:00"
    },
    "encryptionSpecification": {
      "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
      "status": "DISABLED"
    },
    "ttl": {
      "status": "ENABLED"
    },
    "defaultTimeToLive": 2592000,
    "comment": {
      "message": ""
    }
  },

```

```
"replicaSpecifications": []  
}
```

Criar tabela com as configurações de tempo de vida (TTL) personalizadas habilitadas

Para criar uma tabela com 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 os seguintes comandos.

Note

Se uma tabela for criada com as configurações de `ttr` personalizadas habilitadas, você não poderá desativar a configuração posteriormente.

Cassandra Query Language (CQL)

Criar uma tabela com configuração de TTL personalizada usando CQL

- ```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH
CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

## CLI

Crie uma nova tabela com configuração de TTL personalizada usando o AWS CLI

- Você pode usar o seguinte comando para criar uma tabela com a TTL habilitada.

```
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}]' \
 --ttl 'status=ENABLED'
```

- Para confirmar se a TTL está habilitada para a tabela, você pode usar a instrução a seguir.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

A saída da instrução será semelhante à do exemplo a seguir.

```
{
 "keyspaceName": "myKeyspace",
 "tableName": "myTable",
 "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
 "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
 "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": "2024-09-02T11:18:55.796000+00:00"
 },
 "encryptionSpecification": {
 "type": "AWS_OWNED_KMS_KEY"
 },
 "pointInTimeRecovery": {
 "status": "DISABLED"
 },
 "ttl": {
```



```
 "status": "ENABLED"
 },
 "defaultTimeToLive": 0,
 "comment": {
 "message": ""
 },
 "replicaSpecifications": []
}
```

## Atualizar tabela com vida útil (TTL) personalizada

Para habilitar as configurações de vida útil personalizadas para uma tabela de modo que os valores da TTL possam ser aplicadas a linhas e colunas individuais sem definir um valor de TTL padrão para toda a tabela, você pode usar os seguintes comandos.

### Note

Depois de habilitado o `tTL`, você não poderá desabilitá-lo para a tabela.

## Cassandra Query Language (CQL)

Habilitar as configurações de TTL personalizadas para uma tabela usando CQL

- ```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'tTL':{'status': 'enabled'}};
```

CLI

Ative as configurações personalizadas de TTL para uma tabela usando o AWS CLI

1. É possível usar o seguinte comando para atualizar a configuração de TTL personalizada de uma tabela.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
--ttl 'status=ENABLED'
```

2. Para confirmar que a TTL agora está habilitada para a tabela, você pode usar a instrução a seguir.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

A saída da instrução será semelhante à do exemplo a seguir.

```
{
  "keyspaceName": "myKeyspace",
  "tableName": "myTable",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2024-09-02T11:32:27.349000+00:00",
  "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": "2024-09-02T11:32:27.349000+00:00"
  },
  "encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
```

```
    "status": "DISABLED"
  },
  "ttl": {
    "status": "ENABLED"
  },
  "defaultTimeToLive": 0,
  "comment": {
    "message": ""
  },
  "replicaSpecifications": []
}
```

Usar a instrução **INSERT** para definir valores de tempo de vida (TTL) personalizados para novas linhas

Note

Antes de definir valores de TTL personalizados para linhas usando a instrução INSERT, você deve primeiro habilitar a TTL personalizada na tabela. Para obter mais informações, consulte [the section called “Atualizar TTL personalizada da tabela”](#).

Para substituir o valor de TTL padrão de uma tabela definindo datas de expiração para linhas individuais, você pode usar a instrução INSERT:

- INSERT: insira uma nova linha de dados com um conjunto de valores de TTL.

A configuração de valores de TTL para linhas usando a instrução INSERT tem precedência sobre a configuração de TTL padrão da tabela.

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;
```

Usar a instrução **UPDATE** para editar as configurações de vida útil (TTL) personalizadas 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 “Atualizar TTL personalizada da tabela”](#).

Você pode usar a instrução UPDATE para substituir o valor de TTL padrão de uma tabela definindo a data de expiração para linhas e colunas individuais:

- Linhas: você pode atualizar uma linha de dados existente com um valor de TTL personalizado.
- Colunas: você pode atualizar um subconjunto de colunas nas linhas existentes com um valor de TTL personalizado.

A configuração de valores de TTL para linhas e colunas tem precedência sobre a configuração de TTL padrão da tabela.

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

```

Usando esse serviço com um AWS SDK

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 permitem que os desenvolvedores criem facilmente aplicações em seu idioma de preferência.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++ exemplos de código
AWS CLI	AWS CLI exemplos de código
AWS SDK para Go	AWS SDK para Go exemplos de código
AWS SDK para Java	AWS SDK para Java exemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript exemplos de código

Documentação do SDK	Exemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin exemplos de código
AWS SDK para .NET	AWS SDK para .NET exemplos de código
AWS SDK para PHP	AWS SDK para PHP exemplos de código
Ferramentas da AWS para PowerShell	Ferramentas da AWS para PowerShell exemplos de código
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) exemplos de código
AWS SDK para Ruby	AWS SDK para Ruby exemplos de código
AWS SDK para Rust	AWS SDK para Rust exemplos de código
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP exemplos de código
AWS SDK para Swift	AWS SDK para Swift exemplos de código

Exemplo de disponibilidade

Não consegue encontrar o que precisa? Solicite um exemplo de código usando o link Fornecer feedback na parte inferior desta página.

Como trabalhar 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 para usar tags para rotular recursos no Amazon Keyspaces](#).
2. Criar tags usando [Marque keyspaces, tabelas e streams no Amazon Keyspaces](#).
3. Use [Crie relatórios de alocação de custos usando tags 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 para usar tags para rotular recursos no Amazon Keyspaces

Cada tag consiste em uma chave e um valor, ambos definidos por você. As seguintes restrições são aplicáveis:

- Cada keyspace, tabela ou stream do Amazon Keyspaces pode ter somente 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 chave não se aplicam automaticamente às tabelas dentro desse espaço de tecla. Para aplicar a mesma tag a um espaço de chaves e a todas as suas tabelas, cada recurso deve ser marcado individualmente.
- As tags aplicadas a uma tabela não se aplicam automaticamente ao stream dessa tabela. Para aplicar as mesmas tags a uma tabela e ao stream durante a criação da tabela, você pode usar o `PropagateTagsOnEnable` sinalizador ao criar a tabela. Usando essa sinalização, o Amazon Keyspaces aplica as tags da tabela ao stream durante a criação do stream. Quando o stream está ativo, as alterações nas tags da tabela não se aplicam ao stream.
- 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 valores das tags diferenciam maiúsculas de minúsculas.
- O comprimento máximo da chave é 128 caracteres Unicode.
- O número máximo de tags que você pode atribuir a um recurso é 256.
- Os caracteres permitidos são letras, espaço em branco e números, além dos seguintes caracteres especiais: `+ - = . _ : /`
- O número máximo de tags por recurso é 50.
- AWS- os nomes e valores das tags atribuídos recebem automaticamente o `aws :` prefixo, que você não pode atribuir. AWS-os nomes de tag atribuídos não contam para o limite de 50. User-assigned os nomes das tags têm o prefixo `user :` no relatório de alocação de custos.
- Não é possível colocar uma data retroativa na aplicação de uma tag.

Marque keyspaces, tabelas e streams no Amazon Keyspaces

Você pode adicionar, listar, editar ou excluir tags para keyspaces, tabelas e streams usando o console Amazon Keyspaces, o ou a Cassandra Query Language AWS CLI(CQL). Em seguida, você pode ativar essas tags definidas pelo usuário para que elas apareçam no Gerenciamento de Faturamento e Custos da AWS console para rastrear a alocação de custos. Para obter mais informações, consulte [Crie relatórios de alocação de custos usando tags 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 .

Para obter informações sobre a estrutura da tag, consulte [Restrições para usar tags para rotular recursos no Amazon Keyspaces](#).

Tópicos

- [Adicionar tags ao criar um novo espaço de chaves](#)
- [Adicionar tags a um espaço de chaves](#)
- [Excluir tags de um espaço de chaves](#)

- [Exibir as tags de um espaço de chaves](#)
- [Adicionar tags ao criar uma tabela](#)
- [Adicionar tags a uma tabela](#)
- [Excluir tags de uma tabela](#)
- [Exibir as tags de uma tabela](#)
- [Adicione tags a um novo stream ao criar uma tabela](#)
- [Adicionar tags a um novo fluxo para uma tabela existente](#)
- [Adicionar novas tags a um stream](#)
- [Excluir tags de um stream](#)
- [Exibir as tags de um stream](#)

Adicionar tags ao criar um novo espaço de chaves

Você pode usar o console Amazon Keyspaces, o CQL ou o AWS CLI para adicionar tags ao criar um novo keyspace.

Console

Definir uma tag ao criar um novo espaço de chaves usando o console

1. Faça login no Console de gerenciamento da AWS 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.
4. Em Tags selecione Adicionar nova tag e insira uma chave e um valor.
5. Selecione Criar espaço de chaves.

Cassandra Query Language (CQL)

Definir uma tag ao criar um novo espaço de chaves usando CQL

- O exemplo a seguir cria um novo espaço de chaves com tags.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Defina uma tag ao criar um novo espaço de teclas usando a AWS CLI

- A instrução a seguir cria um novo espaço de chaves com tags.

```
aws keyspaces create-keyspace --keyspace-name 'myKeyspace' --tags  
'key=key1,value=val1' 'key=key2,value=val2'
```

Adicionar tags a um espaço de chaves

Os exemplos a seguir mostram como adicionar tags a um espaço de chaves no Amazon Keyspaces.

Console

Adicionar uma tag a um espaço de chaves existente usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces.
3. Selecione um espaço de chave na lista. Em seguida, escolha a guia Tags, onde você pode ver as tags do espaço de chaves.
4. Escolha Gerenciar tags para adicionar, editar ou excluir tags.
5. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Adicionar uma tag a um espaço de chave existente usando CQL

- ```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

## CLI

Adicione uma tag a um espaço de teclas existente usando a AWS CLI

- O seguinte exemplo mostra como adicionar novas tags a um espaço de chaves existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

## Excluir tags de um espaço de chaves

### Console

Excluir uma etiqueta de um espaço de chave existente usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces.
3. Selecione um espaço de chave na lista. Em seguida, escolha a guia Tags, onde você pode ver as tags do espaço de chaves.
4. Escolha Gerenciar tags e exclua as tags que você não precisa mais.
5. Escolha Salvar alterações.

### Cassandra Query Language (CQL)

Excluir uma tag de um espaço de chaves existente usando CQL

- ```
ALTER KEYSPACE mykeyspace DROP TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Exclua uma tag de um espaço de teclas existente usando a AWS CLI

- A instrução a seguir remove as tags especificadas de um espaço de chaves.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Exibir as tags de um espaço de chaves

Os exemplos a seguir mostram como ler tags usando o console, o CQL ou a AWS CLI.

Console

Exibir tags de um espaço de chaves usando o console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Keyspaces.
3. Selecione um espaço de chave na lista. Em seguida, escolha a guia Tags, onde você pode ver as tags do espaço de chaves.

Cassandra Query Language (CQL)

Exibir as tags de um espaço de chaves usando o CQL

Para ler as tags anexadas a um espaço de chaves, use a instrução CQL a seguir.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

A cláusula WHERE é obrigatória e deve usar um dos seguintes formatos:

- `keyspace_name = 'mykeyspace' AND resource_type = 'keyspace'`
- `resource_id = arn`
- A instrução 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-1:111122223333:/keyspace/mykeyspace/ | mykeyspace
| mykeyspace | keyspace | {'key1': 'val1', 'key2': 'val2'}
```

CLI

Visualize as tags de um espaço de teclas usando o AWS CLI

- Este exemplo mostra como listar as tags do recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/'
```

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"
    }
  ]
}
```

Adicionar tags ao criar uma tabela

Você pode usar o console Amazon Keyspaces, o CQL ou o AWS CLI para adicionar tags às novas tabelas ao criá-las.

Console

Adicionar uma tag ao criar uma tabela usando o (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 Criar tabela.

Cassandra Query Language (CQL)

Adicionar tags ao criar uma tabela usando CQL

- O exemplo a seguir cria uma nova tabela com tags.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Adicione tags ao criar uma nova tabela usando a AWS CLI

- 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'
```

Adicionar tags a uma tabela

Você pode adicionar tags a uma tabela existente no Amazon Keyspaces usando o console, o CQL ou a AWS CLI.

Console

Adicionar tags a uma tabela usando o console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas.
3. Escolha uma tabela na lista e selecione a guia Tags.
4. Escolha Gerenciar tags para adicionar tags à tabela.
5. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Adicionar tags a uma tabela usando CQL

- A instrução a seguir mostra como adicionar tags a uma tabela existente.

```
ALTER TABLE mykeyspace.mytable ADD TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Adicione tags a uma tabela usando o AWS CLI

- O seguinte exemplo mostra como adicionar novas tags à tabela existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Excluir tags de uma tabela

Console

Excluir tags de uma tabela usando o console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas.
3. Escolha uma tabela na lista e selecione a guia Tags.
4. Escolha Gerenciar tags para excluir tags de uma tabela.
5. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Excluir tags de uma tabela usando CQL

- A instrução a seguir mostra como excluir tags de uma tabela existente.

```
ALTER TABLE mytable DROP TAGS {'key3':'val3', 'key4':'val4'};
```

CLI

Adicione tags a uma tabela usando o AWS CLI

- A instrução a seguir remove as tags especificadas de um espaço de chaves.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags  
'key=key3,value=val3' 'key=key4,value=val4'
```

Exibir as tags de uma tabela

Os exemplos a seguir mostram como visualizar as tags de uma tabela no Amazon Keyspaces usando o console, o CQL ou o AWS CLI

Console

Exibir as tags de uma tabela usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas.
3. Escolha uma tabela na lista e selecione a guia Tags.

Cassandra Query Language (CQL)

Exibir as tags de uma tabela usando CQL

Para ler as tags anexadas a uma tabela, use a instrução CQL a seguir.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

A cláusula WHERE é obrigatória e deve usar um dos seguintes formatos:

- `keyspace_name = 'mykeyspace' AND resource_name = 'mytable'`
- `resource_id = arn`
- A consulta a seguir retorna as tags da tabela especificada.

```
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-1:111122223333:/keyspace/mykeyspace/table/mytable |  
mykeyspace   | mytable      | table        | {'key1': 'val1', 'key2':  
'val2'}
```

CLI

Visualize as tags de uma tabela usando o AWS CLI

- Este exemplo mostra como listar as tags do recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn  
'arn:aws:arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/  
table/my_table/stream/2025-05-11T21:21:33.291cassandra:us-east-1:111122223333:/  
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"  
    }  
  ]  
}
```

Adicione tags a um novo stream ao criar uma tabela

Você pode adicionar tags aos streams ao criar uma nova tabela com um stream usando CQL ou AWS CLI para marcar um stream.

 Note

O Amazon Keyspaces CDC exige a presença de uma função vinculada ao serviço (AWSServiceRoleForAmazonKeyspacesCDC) que publique dados métricos dos streams do Amazon Keyspaces CDC na sua conta em seu nome. "cloudwatch:namespace": "AWS/Cassandra" CloudWatch Esta função é criada automaticamente para você. Para obter mais informações, consulte [the section called “Transmissões CDC”](#).

Cassandra Query Language (CQL)

Adicione tags a um stream ao criar uma nova tabela usando CQL

1. Para criar uma nova tabela com um fluxo e aplicar as tags da tabela automaticamente ao fluxo, você pode usar o 'propagate_tags': 'TABLE' sinalizador. A instrução a seguir é um exemplo disso.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'propagate_tags': 'TABLE'
    }
};
```

2. Para aplicar novas tags ao stream, você pode usar o exemplo a seguir.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'tags': { 'key': 'string', 'value': 'string' },
    }
};
```

CLI

Adicione tags a um stream ao criar uma nova tabela usando o AWS CLI

1. Para criar uma tabela com um fluxo e aplicar as tags da tabela automaticamente ao fluxo, você pode usar o `propagateTags=Table` sinalizador. O código a seguir é um exemplo disso.

```
aws keyspaces create-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--schema-definition 'allColumns=[{name=pk,type=int},  
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]' \  
\  
--tags key=tag_key,value=tag_value \  
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Para aplicar tags diferentes ao stream, você pode usar o exemplo a seguir.

```
aws keyspaces create-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--schema-definition 'allColumns=[{name=pk,type=int},  
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]' \  
\  
--tags key=tag_key,value=tag_value \  
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,  
value=tag_value}]'
```

Adicionar tags a um novo fluxo para uma tabela existente

Você pode adicionar tags ao criar um novo fluxo para uma tabela existente. Você pode usar o `PropagateTags` sinalizador para aplicar as tags da tabela ao stream ou especificar novas tags para o stream. Você pode usar o CQL ou o AWS CLI para marcar um novo stream.

Note

O Amazon Keyspaces CDC exige a presença de uma função vinculada ao serviço (`AWSServiceRoleForAmazonKeyspacesCDC`) que publique dados métricos dos streams do Amazon Keyspaces CDC na sua conta em seu nome. `"cloudwatch:namespace":`

"AWS/Cassandra" CloudWatch Esta função é criada automaticamente para você. Para obter mais informações, consulte [the section called "Transmissões CDC"](#).

Console

Adicione tags ao criar um novo stream usando o (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, escolha Tabelas e, em seguida, escolha a tabela para a qual você deseja adicionar um fluxo.
3. Escolha a guia Streams.
4. Na seção Detalhes do stream, escolha Editar.
5. Selecione Ativar transmissões.
6. Selecione o tipo de visualização e continue até Tags para criar tags para o stream.
7. Você pode selecionar uma das seguintes opções:
 - Sem tags — Use essa opção se você não quiser criar nenhuma tag para o stream.
 - Copiar tags da tabela — Use essa opção se quiser copiar as tags da tabela para o stream. Depois de copiar as tags, você pode editá-las para o stream. Observe que essa opção só está disponível se a tabela tiver tags.
 - Adicionar novas tags — Você pode adicionar até 50 tags ao stream escolhendo Adicionar nova tag.
8. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Adicione tags ao criar um novo stream

1. Para criar um novo fluxo para uma tabela existente e aplicar as tags da tabela ao fluxo, você pode usar o 'propagate_tags': 'TABLE' sinalizador. As declarações a seguir são um exemplo disso.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':  
  { 'view_type': 'NEW_IMAGE', 'propagate_tags': 'TABLE' } };
```

2. Para criar um novo fluxo para uma tabela existente e especificar novas tags, você pode usar o exemplo a seguir.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':  
  { 'view_type': 'NEW_IMAGE', 'tags': { 'key': 'string', 'value': 'string' }} };
```

CLI

Adicione tags ao criar um novo stream usando o AWS CLI

1. Para criar um novo stream com tags, você pode usar o `propagateTags=TABLE` sinalizador para aplicar as tags da tabela automaticamente ao stream. O código a seguir é um exemplo disso.

```
aws keyspaces update-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Para criar um novo fluxo para uma tabela existente e especificar novas tags, você pode usar o exemplo a seguir.

```
aws keyspaces update-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,  
value=tag_value}]'
```

Adicionar novas tags a um stream

Você pode adicionar novas tags a um stream existente no Amazon Keyspaces usando o CQL ou o AWS CLI. Você só pode adicionar tags ao stream mais recente.

Console

Adicionar tags a um stream existente (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em: <https://console.aws.amazon.com/keyspaces/home>

2. No painel de navegação, escolha Tabelas e, em seguida, escolha a tabela com o fluxo que você deseja marcar.
3. Escolha a guia Streams.
4. Na seção Tags, escolha Gerenciar tags.
5. Escolha Adicionar nova tag para adicionar uma nova tag. Você pode criar até 50 tags repetindo essa etapa.
6. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Adicionar tags a um stream usando CQL

- A declaração a seguir mostra como adicionar tags a um stream existente.

```
ALTER TABLE mykeyspace.mytable ADD TAGS_FOR_CDC {'key1':'val1', 'key2':'val2'};
```

CLI

Adicione tags a um stream usando o AWS CLI

- O exemplo a seguir mostra como adicionar novas tags a um stream existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Excluir tags de um stream

Para excluir tags de um stream, você pode usar CQL ou o AWS CLI. Você só pode excluir as tags do stream mais recente.

Console

Excluir tags de uma tabela usando o console do Amazon Keyspaces

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas.
3. Escolha uma tabela na lista e escolha a guia Streams.
4. Na seção Tags, escolha Gerenciar tags para excluir tags da tabela.
5. Depois da tag que você deseja excluir, escolha Remover.
6. Escolha Salvar alterações.

Cassandra Query Language (CQL)

Excluir tags de um stream usando CQL

- A declaração a seguir mostra como excluir tags de um stream existente.

```
ALTER TABLE mytable DROP TAGS_FOR_CDC {'key3':'val3', 'key4':'val4'};
```

CLI

Excluir tags de um stream usando o AWS CLI

- A declaração a seguir remove as tags especificadas de um stream.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Exibir as tags de um stream

Os exemplos a seguir mostram como visualizar as tags de um stream no Amazon Keyspaces usando CQL ou o AWS CLI

Console

Visualize as tags de um stream usando o console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em. <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas.
3. Escolha uma tabela na lista e escolha a guia Streams.
4. Você pode ver as tags do stream na seção Tags.

Cassandra Query Language (CQL)

Visualize as tags de um stream usando CQL

Para ler as tags anexadas a um stream, você deve especificar o ARN do recurso do stream na WHERE cláusula. A sintaxe CQL a seguir é um exemplo disso.

```
SELECT * FROM system_schema_mcs.tags WHERE resource_id = stream_arn;
```

- A consulta a seguir retorna as tags do fluxo especificado.

```
SELECT tags FROM system_schema_mcs.tags WHERE resource_id =  
'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/  
stream/2025-05-06T17:17:39.800';
```

A saída dessa consulta é semelhante à seguinte.

```
resource_id
                                     | keyspace_name | resource_name
resource_type | tags
-----+-----+-----
+-----+-----+-----
arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-04-02T23:00:07.052      |      singleks | 2025-04-02T23:00:07.052 |
stream | {'tagkey': 'tagval'}
```

CLI

Visualize as tags de um stream usando o AWS CLI

- Este exemplo mostra como listar as tags de todos os fluxos no espaço de teclas especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291'
```

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"
    }
  ]
}
```

Crie relatórios de alocação de custos usando tags para Amazon Keyspaces

AWS usa tags para organizar os custos dos recursos em seu relatório de alocação de custos. AWS fornece dois tipos de etiquetas de alocação de custos:

- Uma tag AWS gerada. AWS define, cria e aplica essa tag para você.

- User-defined etiquetas. 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 as tags AWS geradas:

1. Faça login no Console de gerenciamento da AWS e abra o console Billing and Cost Management <https://console.aws.amazon.com/billing/home#>.
2. No painel de navegação, escolha Cost Allocation Tags.
3. Em AWS Generated Cost Allocation Tags (Etiquetas de alocação de custos geradas), escolha Activate (Ativar).

Para ativar tags definidas pelo usuário:

1. Faça login no Console de gerenciamento da AWS e abra o console Billing and Cost Management <https://console.aws.amazon.com/billing/home#>.
2. No painel de navegação, escolha Cost Allocation Tags.
3. Em Tags User-Defined de alocação de custos, escolha Ativar.

Depois de criar e ativar as tags, AWS gera um relatório de alocação de custos com seu uso e custos agrupados pelas tags ativas. O relatório de alocação de custos inclui todos os AWS custos de cada período de cobrança. 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](#).

Crie recursos do Amazon Keyspaces com AWS CloudFormation

O Amazon Keyspaces é integrado com AWS CloudFormation, um serviço que ajuda você a modelar e configurar seus AWS espaços de chaves e tabelas para que você possa gastar menos tempo

criando e gerenciando seus recursos e infraestrutura. Você cria um modelo que descreve os espaços de chave e as tabelas desejados e se CloudFormation encarrega de provisionar e configurar esses recursos para você.

Ao usar CloudFormation, você pode reutilizar seu modelo para configurar seus recursos do Amazon Keyspaces de forma consistente e repetida. Basta descrever seus recursos uma vez e, em seguida, provisionar os mesmos recursos repetidamente em várias Contas da AWS regiões.

Amazon Keyspaces e modelos CloudFormation

Para provisionar e configurar recursos para o Amazon Keyspaces, é preciso entender os [modelos do CloudFormation](#). Os modelos são arquivos de texto formatados em JSON ou YAML. Esses modelos descrevem os recursos que você deseja provisionar em suas CloudFormation pilhas. Se você não estiver familiarizado com JSON ou YAML, você pode usar o CloudFormation Designer para ajudá-lo a começar a usar modelos. CloudFormation Para obter mais informações, consulte [O que é o Designer de CloudFormation ?](#) no Manual do usuário do AWS CloudFormation .

O Amazon Keyspaces oferece suporte à criação de keyspaces e tabelas em CloudFormation. Para as tabelas que você cria usando CloudFormation modelos, você pode especificar o esquema, o read/write modo, as configurações de taxa de transferência provisionada e outros recursos compatíveis. Para obter mais informações, incluindo exemplos de modelos JSON e YAML para espaços de chave e tabelas, consulte a referência do tipo de [recurso Amazon Keyspaces \(para Apache Cassandra\)](#) na Referência de modelo.AWS CloudFormation

Saiba mais sobre CloudFormation

Para saber mais sobre isso CloudFormation, consulte os seguintes recursos:

- [AWS CloudFormation](#)
- [AWS CloudFormation Guia do usuário](#)
- [AWS CloudFormation Guia do usuário da interface de linha de comando](#)

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 “Criar um modelo de dados”](#) e [the section called “Editar um modelo 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 “Visualizar um modelo de dados”](#).

Tópicos

- [Fazer download do NoSQL Workbench](#)
- [Noções básicas do NoSQL Workbench](#)
- [Visualizar modelos de dados com o NoSQL Workbench](#)
- [Criar um novo modelo de dados com o NoSQL Workbench](#)
- [Editar modelos de dados existentes com o NoSQL Workbench](#)
- [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.

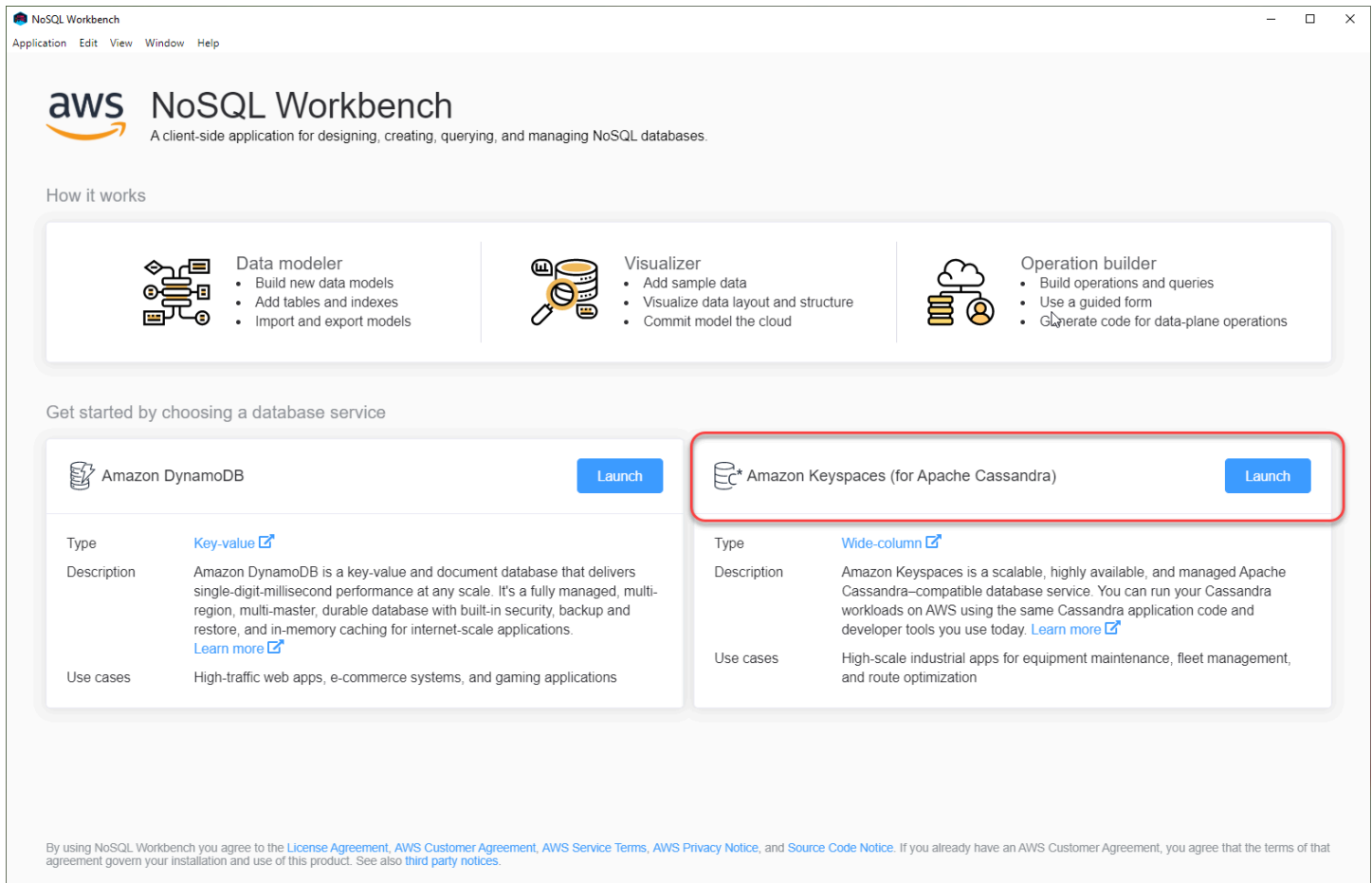
Sistema operacional	Link para download
macOS	Download para macOS
Linux*	Download para Linux
Windows	Download para Windows

* O NoSQL Workbench oferece suporte a Ubuntu 12.04, Fedora 21 e Debian 8 ou quaisquer versões mais recentes dessas distribuições do Linux.

- Depois que o download for concluído, inicie o aplicativo e siga as instruções na tela para concluir a instalação.

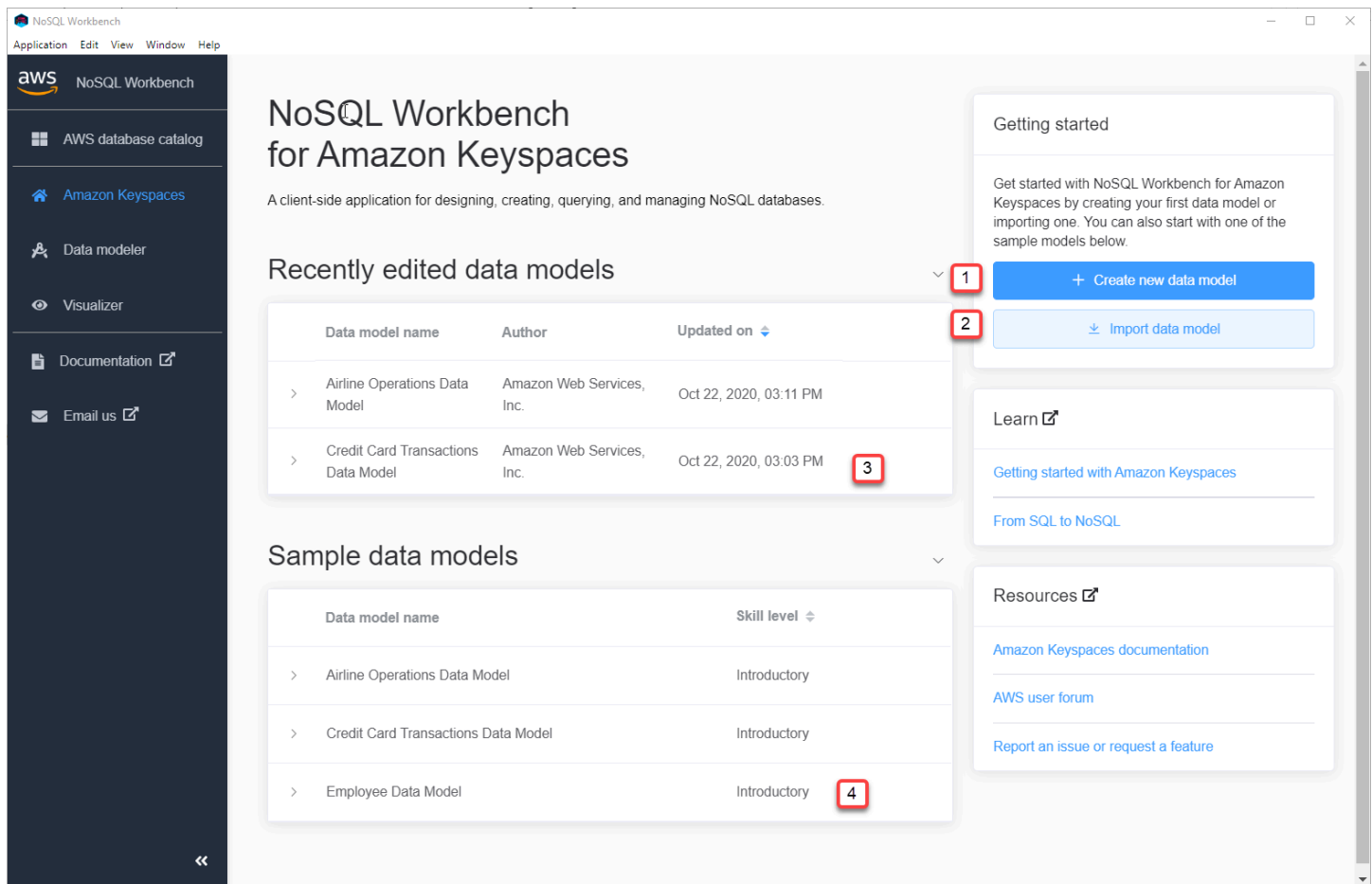
Noções básicas do NoSQL Workbench

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 “Criar um modelo de dados”](#). Para editar um modelo de dados existente, consulte [the section called “Editar um modelo de dados”](#).

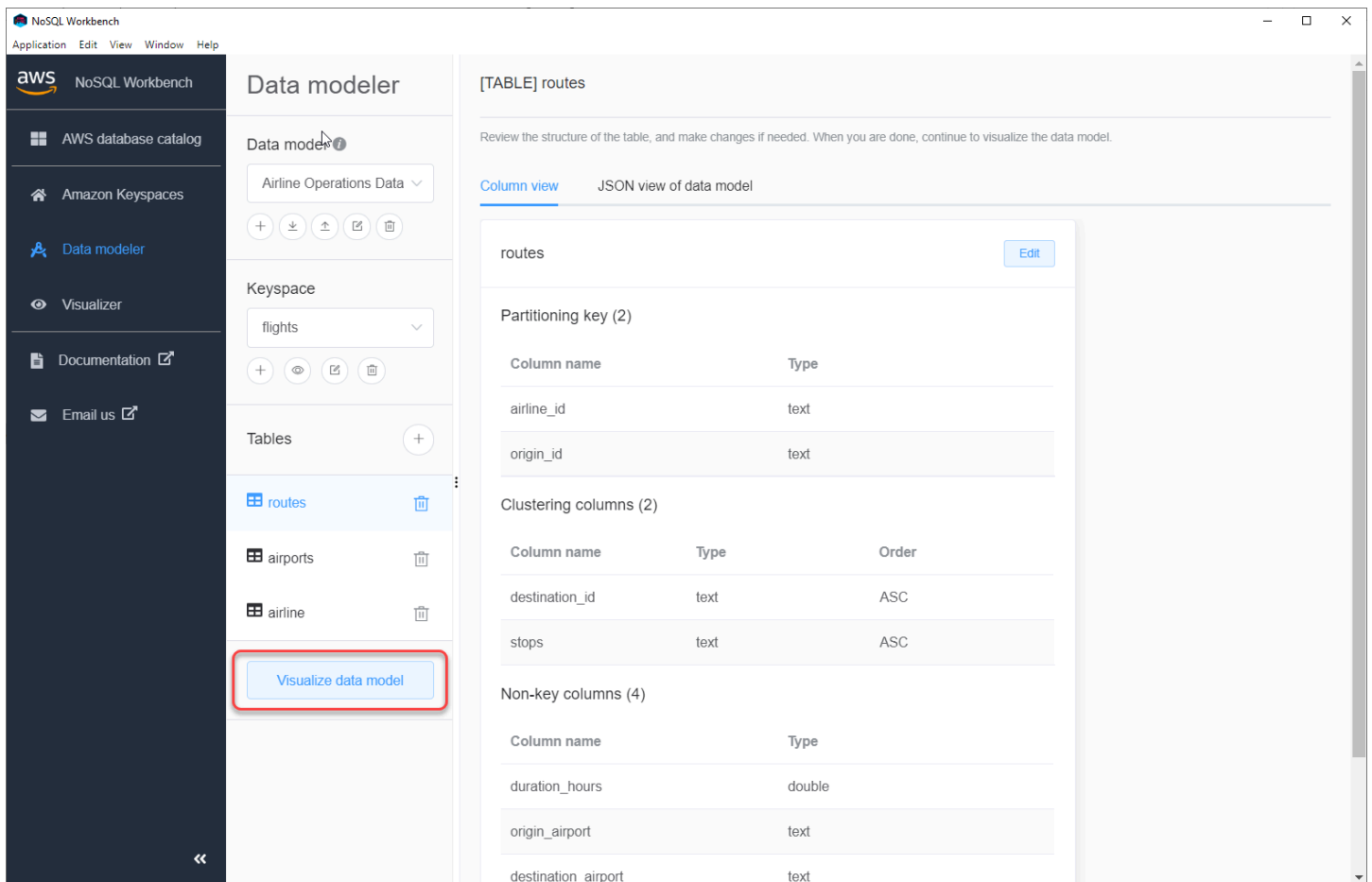
Visualizar modelos de dados com o NoSQL Workbench

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.



The screenshot shows the AWS NoSQL Workbench Data Modeler interface. The left sidebar contains the following menu items: AWS database catalog, Amazon Keyspaces, Data modeler (selected), Visualizer, Documentation, and Email us. The main area is titled 'Data modeler' and shows the 'Airline Operations Data' dataset. The 'Keyspaces' section lists 'flights'. The 'Tables' section lists 'routes', 'airports', and 'airline'. The 'Visualize data model' button is highlighted with a red box. The main area displays the 'routes' table structure in 'Column view'.

[TABLE] routes

Review the structure of the table, and make changes if needed. When you are done, continue to visualize the data model.

Column view JSON view of data model

routes

Partitioning key (2)

Column name	Type
airline_id	text
origin_id	text

Clustering columns (2)

Column name	Type	Order
destination_id	text	ASC
stops	text	ASC

Non-key columns (4)

Column name	Type
duration_hours	double
origin_airport	text
destination_airport	text

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.

The screenshot shows the AWS NoSQL Workbench Visualizer interface. The left sidebar contains navigation options: AWS database catalog, Amazon Keyspaces, Data modeler, Visualizer (selected), Documentation, and Email us. The main panel is titled 'Visualizer' and shows the 'Data model' for 'Airline Operations'. The 'Keyspace' is set to 'flights'. The 'Tables' list includes 'routes', 'airports', and 'airline'. The 'routes' table is selected, and its schema is displayed in the main panel. The table has the following columns: 'airline_id (text)' (Primary key), 'destination_id (text)' (Partition key), 'stops (text)' (Clustering column), 'duration_hours (double)' (Non-key column), and 'origin_airport (text)' (Non-key column). The table contains three rows of data. A red box highlights the '+ Add new row' button at the bottom of the table.

Primary key			Non-key columns	
Partition key	Clustering columns		duration_hours (double)	origin_airport (text)
airline_id (text)	destination_id (text)	stops (text)		
acme_airlines	MCI	1	0.33333333	Newark Liberty
trusted_airlines	MIC	2	0.33333333	Phoenix Sky Ha
freedom_airline	EWR	1	0.33333333	San Francisco

Exibição agregada

Depois de confirmar o esquema da tabela, você pode agregar exibições do modelo de dados.

Visualizer

Data model ⓘ

Airline Operations Data ▾

⬇ ⬆

Keyspace

flights ▾

Tables

- routes
- airports
- airline

Aggregate view

Commit to Amazon Keyspaces

Commit to Apache Cassandra

[TABLE] routes Edit

Use this view to get a visual representation of your tables schema, and add sample data. When you are done, commit your model to Amazon Keyspaces.

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

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.

Visualizer

Data model ⓘ

Airline Operations Data ▾

↓ ↑

Keyspace

flights ▾

Tables

- routes
- airports
- airline

Aggregate view

Commit to Amazon Keyspaces

Commit to Apache Cassandra

Export to PNG

Aggregate view

[TABLE] routes

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

[TABLE] airports

Primary key	Non-key columns			
Partition key				
airport_id (text)	name (text)	city (text)	iata_code (text)	details (map<text,text>)
EWR				

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”](#).

Criar um novo modelo de dados com o NoSQL Workbench

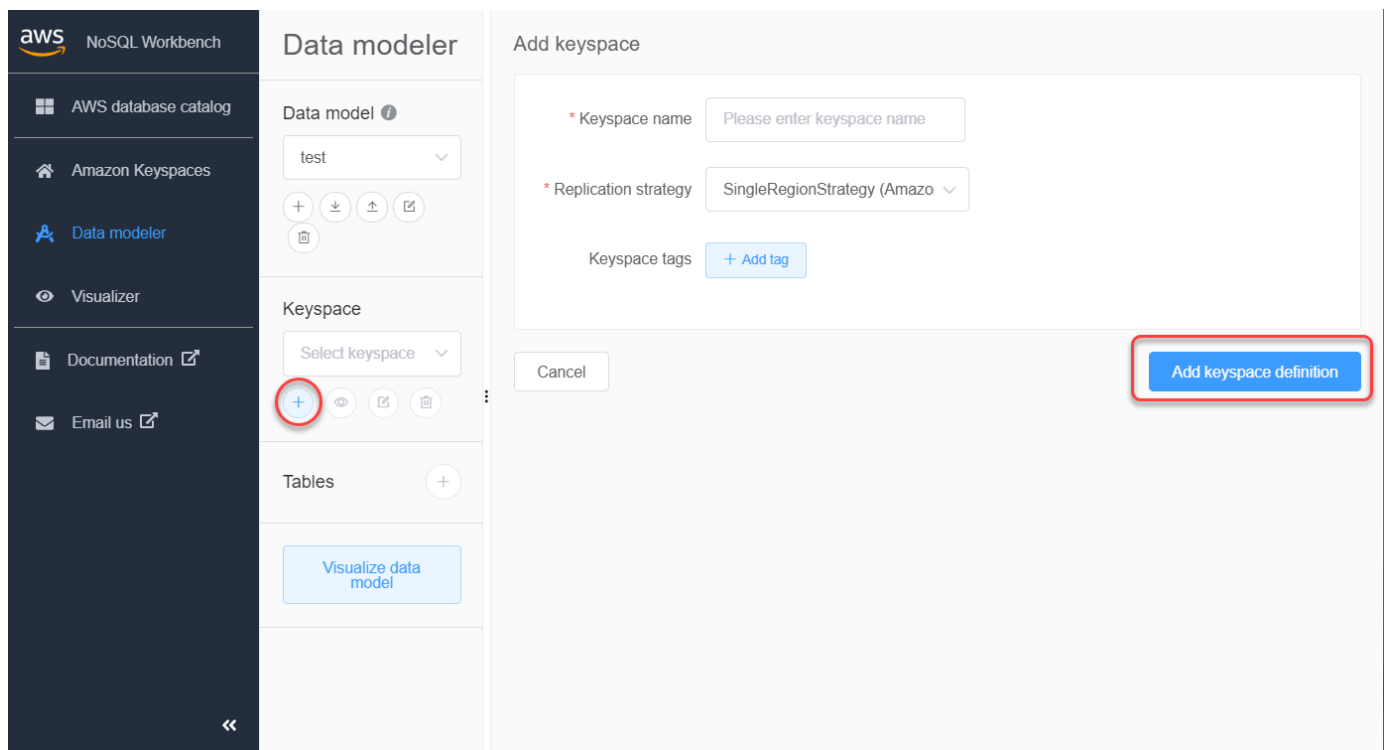
Você pode usar o modelador de dados NoSQL Workbench para projetar novos modelos de dados com base nos padrões de acesso a dados do seu aplicativo. Para criar um novo modelo de dados para o Amazon Keyspaces, você pode usar o modelador de dados NoSQL Workbench para criar espaços de chaves, tabelas e colunas. 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 o SingleRegionStrategy para replicar dados três vezes automaticamente em várias AWS zonas de disponibilidade. Se você planeja comprometer o modelo de dados em um cluster Apache Cassandra, você pode escolher ou. SimpleStrategyNetworkTopologyStrategy
- 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 “Como trabalhar 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 — Escolha o modo de read/write capacidade para a tabela. Você pode escolher capacidade provisionada ou sob demanda. Para saber mais sobre modos de capacidade, consulte [the section called “Configurar modos read/write de capacidade”](#).
 - 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 “Como trabalhar 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.

The screenshot shows the AWS NoSQL Workbench interface. On the left is a dark sidebar with navigation links: AWS database catalog, Amazon Keyspaces, Data modeler (highlighted), Visualizer, Documentation, and Email us. The main area is titled 'Data modeler' and contains a 'Data model' section with a dropdown set to 'test', a 'Keyspace' section with a dropdown set to 'key1', and a 'Tables' section with a red box around a '+' button. Below the 'Tables' section is a 'Visualize data model' button. The right pane is titled 'Add table definition' and contains a form with the following fields: 'Keyspace' (dropdown set to 'key1'), '* Table name' (text input with placeholder 'Please enter table name'), '* Columns' (table with 'Enter column name' and 'Column type' dropdown, and a '+ Add another column' button), '* Partition key' (dropdown with placeholder 'Select columns to add to partition'), 'Clustering columns' (+ Add clustering column button), 'Capacity mode' (radio buttons for 'Provisioned' and 'On-demand', with 'On-demand' selected), and 'Table tags' (+ Add tag button). At the bottom right of the form is a red box around the 'Add table definition' button. A 'Cancel' button is at the bottom left of the form.

Editar modelos de dados existentes com o NoSQL Workbench

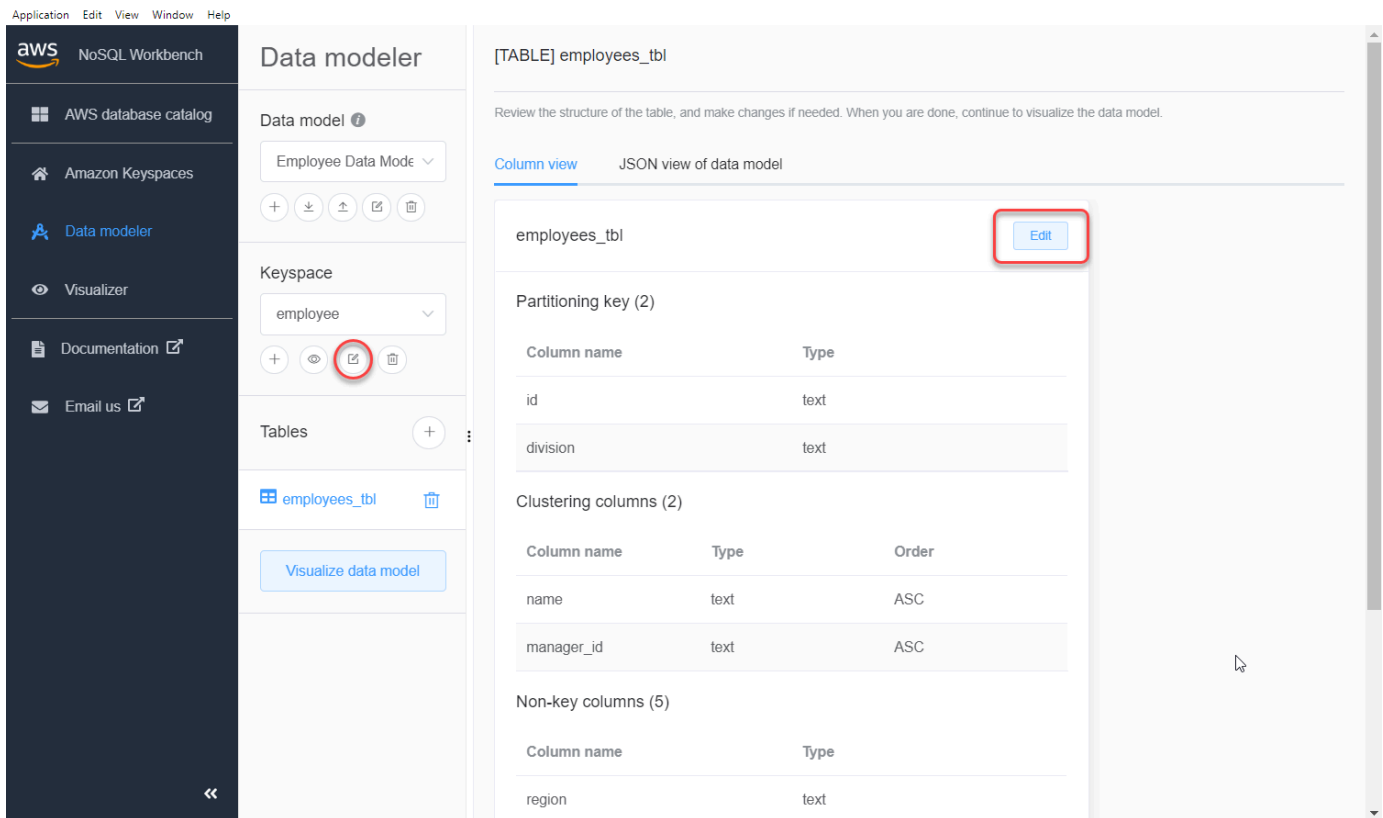
Você pode usar o modelador de dados para importar e modificar modelos de dados existentes criados usando o NoSQL Workbench. O modelador de dados também inclui alguns modelos de dados de amostra para ajudá-lo a começar a modelar dados. Os modelos de dados que você pode editar no NoSQL Workbench podem 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 o `SingleRegionStrategy` para replicar dados três vezes automaticamente em várias AWS zonas de disponibilidade. Se você planeja comprometer o modelo de dados em um cluster Apache Cassandra, você pode escolher ou `SimpleStrategy` `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 “Como trabalhar 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 — Escolha o modo de read/write capacidade para a tabela. Você pode escolher capacidade provisionada ou sob demanda. Para saber mais sobre modos de capacidade, consulte [the section called “Configurar modos read/write de capacidade”](#).
 - 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 “Como trabalhar 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 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.

The screenshot shows the AWS NoSQL Workbench Visualizer interface. On the left, the 'Visualizer' sidebar is open, showing the 'Data model' section with 'Airline Operations Data' selected. Below this, the 'Keyspace' is set to 'flights' and the 'Tables' list includes 'routes', 'airports', and 'airline'. The 'Commit to Amazon Keyspaces' button is highlighted with a red rectangle. The main area displays the 'routes' table schema and data.

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

Tópicos

- [Antes de começar](#)
- [Conectar-se ao Amazon Keyspaces com credenciais específicas do serviço](#)
- [Conecte-se ao Amazon Keyspaces com credenciais AWS Identity and Access Management \(IAM\)](#)
- [Usar uma conexão salva](#)
- [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 dos seguintes certificados digitais e salve os arquivos localmente ou em seu diretório pessoal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opcional — para compatibilidade com versões anteriores)

Para baixar os certificados, você pode usar os seguintes comandos.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

Note

O Amazon Keyspaces usava anteriormente certificados TLS ancorados na CA Starfield Class 2. AWS está migrando tudo Regiões da AWS para certificados emitidos pelo Amazon Trust Services (Amazon Root Cas 1—4). Durante essa transição, configure os clientes para que confiem tanto no Amazon Root CAs 1—4 quanto na raiz Starfield para garantir a compatibilidade em todas as regiões.

Combine todos os certificados baixados em um único pem arquivo com o nome **keyspaces-bundle.pem** em nossos exemplos. Você pode fazer isso executando o seguinte comando : Anote o caminho para o arquivo, você precisará disso mais tarde.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
```

```
AmazonRootCA4.pem \  
sf-class2-root.crt \  
> keyspaces-bundle.pem
```

Depois de salvar o arquivo do certificado, você pode se conectar ao Amazon Keyspaces. Uma opção é se conectar usando credenciais específicas do serviço. Service-specific credenciais são um nome de usuário e uma senha associados a um usuário específico do IAM 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 “Crie credenciais programáticas de acesso”](#).

Para conectar-se com credenciais específicas do serviço, consulte [the section called “Conecte-se com credenciais específicas do serviço”](#).

Para conectar-se com credenciais do IAM, consulte [the section called “Conectar-se com credenciais do IAM”](#).

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 “Crie 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”](#).
- 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

Conecte-se ao Amazon Keyspaces com credenciais AWS Identity and Access Management (IAM)

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 [Console de gerenciamento da AWS](#) da página de login do IAM. Para obter informações sobre credenciais AWS de segurança, incluindo acesso programático e alternativas às credenciais de longo prazo, consulte as credenciais de [AWS segurança no Guia do usuário](#) do IAM. Para obter detalhes sobre como fazer login no seu Conta da AWS, consulte [Como fazer login AWS no](#) Guia do Início de Sessão da AWS usuário.
 - Para acesso à CLI, é necessário ter 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 [Uso de credenciais temporárias com AWS recursos](#) 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”](#).

- 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 AWS certificado que foi baixado na primeira etapa.
 - Conexão persistente — Marque essa caixa de seleção se quiser salvar os segredos da AWS conexão localmente.
2. Escolha Confirmar para atualizar o Amazon Keyspaces com o modelo de dados.

i 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. **i**

Persist connection



If you select this check box, AWS connection secrets will be persisted in

C:\Users\alexis.fid.aws\credentials

Cancel

Reset

Commit

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.

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 : >

* Saved connections

default



* 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.

Cancel

Reset

Commit

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

Password

* Contact points

[+ Add another contact point](#)

* Local data center

* Port

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 dados não relacionais para o Amazon Keyspaces com mais facilidade. Os clientes	5 de outubro de 2020

Alteração	Descrição	Data
	do NoSQL Workbench estão disponíveis para Windows, macOS e Linux. Para obter mais informações, consulte NoSQL Workbench para Amazon Keyspaces .	

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).

As noções básicas são exemplos de código que mostram como realizar as operações essenciais em um serviço.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- [Exemplos básicos para Amazon Keyspaces usando SDKs AWS](#)
 - [Olá, Amazon Keyspaces](#)
 - [Aprenda as noções básicas do Amazon Keyspaces com um SDK AWS](#)
 - [Ações para Amazon Keyspaces usando SDKs AWS](#)
 - [Use CreateKeyspace com um AWS SDK](#)
 - [Use CreateTable com um AWS SDK](#)
 - [Use DeleteKeyspace com um AWS SDK](#)
 - [Use DeleteTable com um AWS SDK](#)
 - [Use GetKeyspace com um AWS SDK](#)
 - [Use GetTable com um AWS SDK](#)
 - [Use ListKeyspaces com um AWS SDK](#)
 - [Use ListTables com um AWS SDK](#)
 - [Use RestoreTable com um AWS SDK](#)
 - [Use UpdateTable com um AWS SDK](#)

Exemplos básicos para Amazon Keyspaces usando SDKs AWS

Os exemplos de código a seguir mostram como usar os elementos básicos do Amazon Keyspaces (para Apache Cassandra) com SDKs da AWS .

Exemplos

- [Olá, Amazon Keyspaces](#)
- [Aprenda as noções básicas do Amazon Keyspaces com um SDK AWS](#)
- [Ações para Amazon Keyspaces usando SDKs AWS](#)
 - [Use CreateKeyspace com um AWS SDK](#)
 - [Use CreateTable com um AWS SDK](#)
 - [Use DeleteKeyspace com um AWS SDK](#)
 - [Use DeleteTable com um AWS SDK](#)
 - [Use GetKeyspace com um AWS SDK](#)
 - [Use GetTable com um AWS SDK](#)
 - [Use ListKeyspaces com um AWS SDK](#)
 - [Use ListTables com um AWS SDK](#)
 - [Use RestoreTable com um AWS SDK](#)
 - [Use UpdateTable com um AWS SDK](#)

Olá, Amazon Keyspaces

Os exemplos de código a seguir mostram como começar a usar o Amazon Keyspaces.

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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.fromEnvironment { 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 [ListKeyspaces](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) Referência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Aprenda as noções básicas do Amazon Keyspaces com 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 recuperação para um ponto no tempo.
- 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.
- Restaure a tabela ao estado anterior e limpe os recursos.

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
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.WriteLine($"Created {keyspaceName}. Waiting for it to become
available. ");
    do
    {

```

```
        getKeyspaceArn = await
keyspacesWrapper.GetKeyspace(keyspaceName);
        Console.WriteLine(". ");
    } 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.WriteLine("Waiting for the new table to be active. ");
}
```

```
do
{
    try
    {
        resp = await keyspacesWrapper.GetTable(keyspaceName,
tableName);

        Console.WriteLine(".");
    }
    catch (ResourceNotFoundException)
    {
        Console.WriteLine(".");
    }
} 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.WriteLine(".");
        }
    }

    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 para .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 [AWS Code Examples Repository](#).

```
/**
 * 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)
                .addColumnns(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 saiba 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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\n"
        "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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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 schemaDefinition0b =
    SchemaDefinition {
        partitionKeys = keyList
        allColumns = colList
    }

val timeRecovery =
    PointInTimeRecovery {
        status = PointInTimeRecoveryStatus.Enabled
    }

val tableRequest =
    CreateTableRequest {
        keyspaceName = keySpaceVal
        tableName = tableNameVal
        schemaDefinition = schemaDefinition0b
        pointInTimeRecovery = timeRecovery
    }

KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
    val response = keyClient.createTable(tableRequest)
    println("The table ARN is ${response.resourceArn}")
}

suspend fun listKeyspacesPaginator() {
    KeyspacesClient.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Consulte detalhes da API nos tópicos a seguir na Referência de API do 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 saiba 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 keyspace 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, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Ações para Amazon Keyspaces usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Keyspaces com AWS SDKs. 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](#)
- [Use CreateTable com um AWS SDK](#)
- [Use DeleteKeyspace com um AWS SDK](#)
- [Use DeleteTable com um AWS SDK](#)
- [Use GetKeyspace com um AWS SDK](#)
- [Use GetTable com um AWS SDK](#)
- [Use ListKeyspaces com um AWS SDK](#)
- [Use ListTables com um AWS SDK](#)
- [Use RestoreTable com um AWS SDK](#)
- [Use UpdateTable com um AWS SDK](#)

Use **CreateKeyspace** com um AWS SDK

Os exemplos de código a seguir mostram como usar o `CreateKeyspace`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateKeyspace](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateKeyspace](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { 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 [CreateKeyspace](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateKeyspace](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_kys->createkeyspace(
        iv_keyspacename = iv_keyspace_name ).
    MESSAGE 'Keyspace created successfully.' TYPE 'I'.
CATCH /aws1/cx_kysconflictexception.
    MESSAGE 'Keyspace already exists.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.

```

```
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateKeyspace](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **CreateTable** com um AWS SDK

Os exemplos de código a seguir mostram como usar o CreateTable.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateTable](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateTable](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 schemaDefinition0b =
    SchemaDefinition {
        partitionKeys = keyList
        allColumns = collist
    }

val timeRecovery =
    PointInTimeRecovery {
        status = PointInTimeRecoveryStatus.Enabled
    }

val tableRequest =
```

```

        CreateTableRequest {
            keyspaceName = keySpaceVal
            tableName = tableNameVal
            schemaDefinition = schemaDefinitionObj
            pointInTimeRecovery = timeRecovery
        }

        KeyspacesClient.fromEnvironment { 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 [CreateTable](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [CreateTable](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    " Define schema with columns
    DATA(lt_columns) = VALUE /aws1/
cl_kyscolumnndefinition=>tt_columnndefinitionlist(
    ( NEW /aws1/cl_kyscolumnndefinition( iv_name = 'title' iv_type =
'text' ) )
    ( NEW /aws1/cl_kyscolumnndefinition( iv_name = 'year' iv_type =
'int' ) )
    ( NEW /aws1/cl_kyscolumnndefinition( iv_name = 'release_date' iv_type =
'timestamp' ) )
    ( NEW /aws1/cl_kyscolumnndefinition( iv_name = 'plot' iv_type =
'text' ) )
    ).

    " Define partition keys
    DATA(lt_partition_keys) = VALUE /aws1/
cl_kyspartitionkey=>tt_partitionkeylist(
    ( NEW /aws1/cl_kyspartitionkey( iv_name = 'year' ) )
    ( NEW /aws1/cl_kyspartitionkey( iv_name = 'title' ) )
    ).

    " Create schema definition
    DATA(lo_schema) = NEW /aws1/cl_kysschemadefinition(
        it_allcolumns = lt_columns
        it_partitionkeys = lt_partition_keys ).

    " Enable point-in-time recovery
    DATA(lo_pitr) = NEW /aws1/cl_kyspointintimerec(
        iv_status = 'ENABLED' ).

    oo_result = lo_kys->createtable(
        iv_keyspacename = iv_keyspace_name

```

```

        iv_tablename = iv_table_name
        io_schemadefinition = lo_schema
        io_pointintimerecovery = lo_pitr ).
    MESSAGE 'Table created successfully.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = "{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [CreateTable](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **DeleteKeyspace** com um AWS SDK

Os exemplos de código a seguir mostram como usar o DeleteKeyspace.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteKeyspace](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteKeyspace](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}
```

- Para obter detalhes da API, consulte a [DeleteKeyspace](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteKeyspace](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_kys->deletekeyspace(  
        iv_keyspacename = iv_keyspace_name ).  
    MESSAGE 'Keyspace deleted successfully.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteKeyspace](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **DeleteTable** com um AWS SDK

Os exemplos de código a seguir mostram como usar o DeleteTable.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteTable](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteTable](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}
```

```
}
```

- Para obter detalhes da API, consulte a [DeleteTable](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [DeleteTable](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    lo_kys->deletetable(
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name ).
    MESSAGE 'Table deleted successfully.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DeleteTable](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **GetKeyspace** com um AWS SDK

Os exemplos de código a seguir mostram como usar o GetKeyspace.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetKeyspace](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetKeyspace](#) Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response: GetKeyspaceResponse =
            keyClient.getKeyspace(keyRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}
```

- Para obter detalhes da API, consulte a [GetKeyspace](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetKeyspace](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_kys->getkeyspace(  
        iv_keyspacename = iv_keyspace_name ).  
    MESSAGE 'Keyspace retrieved successfully.' TYPE 'I'.  
CATCH /aws1/cx_kysresourcenotfoundex.  
    MESSAGE 'Keyspace does not exist.' TYPE 'I'.  
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetKeyspace](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **GetTable** com um AWS SDK

Os exemplos de código a seguir mostram como usar o `GetTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetTable](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetTable](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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.fromEnvironment { 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 [GetTable](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [GetTable](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_kys->gettable(
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name ).
    MESSAGE 'Table information retrieved successfully.' TYPE 'I'.
CATCH /aws1/cx_kysresourcenotfoundex.
    MESSAGE 'Table does not exist.' TYPE 'I'.

```

```
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetTable](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **ListKeyspaces** com um AWS SDK

Os exemplos de código a seguir mostram como usar o `ListKeyspaces`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) na Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun listKeyspacesPaginator() {  
    KeyspacesClient.fromEnvironment { 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 [ListKeyspaces](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListKeyspaces](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_kys->listkeyspaces(
        iv_maxresults = iv_max_results ).
    MESSAGE 'Keyspaces listed successfully.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListKeyspaces](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **ListTables** com um AWS SDK

Os exemplos de código a seguir mostram como usar o `ListTables`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListTables](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListTables](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { 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 [ListTables](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [ListTables](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_kys->listtables(  
        iv_keyspacename = iv_keyspace_name ).  
    MESSAGE 'Tables listed successfully.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListTables](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **RestoreTable** com um AWS SDK

Os exemplos de código a seguir mostram como usar o `RestoreTable`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [RestoreTable](#) na Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [RestoreTable](#) Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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.fromEnvironment { 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 [RestoreTable](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [RestoreTable](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

TRY.

```

oo_result = lo_kys->restoretale(
    iv_sourcekeyspacename = iv_source_keyspace_name
    iv_sourcetablename = iv_source_table_name
    iv_targetkeyspacename = iv_target_keyspace_name
    iv_targettablename = iv_target_table_name
    iv_restoretimestamp = iv_restore_timestamp ).
MESSAGE 'Table restore initiated successfully.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).

```

```
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [RestoreTable](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **UpdateTable** com um AWS SDK

Os exemplos de código a seguir mostram como usar o UpdateTable.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- [Conheça os conceitos básicos](#)

.NET

SDK para .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [UpdateTable](#) a Referência AWS SDK para .NET da API.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [UpdateTable](#) a Referência AWS SDK for Java 2.x da API.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}
```

```
}
}
```

- Para obter detalhes da API, consulte a [UpdateTable](#) referência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba 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 [UpdateTable](#) Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    " Add a new column to track watched movies
    DATA(lt_add_columns) = VALUE /aws1/
cl_kyscolumndefinition=>tt_columndefinitionlist(
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'watched' iv_type =
'boolean' ) )
    ).

    oo_result = lo_kys->updatetable(
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name

```

```
        it_addcolumns = lt_add_columns ).  
        MESSAGE 'Table updated successfully.' TYPE 'I'.  
        CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.  
        MESSAGE lv_error TYPE 'E'.  
    ENDTRY.
```

- Para obter detalhes da API, consulte a [UpdateTable](#) referência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulte [Usando esse serviço com um AWS SDK](#). 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 GitHub nos repositórios de amostras AWS.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 AWS autenticação 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>

CloudFormation modelo para criar um CloudWatch painel da Amazon para métricas do Amazon Keyspaces (para Apache Cassandra)

Esse repositório fornece CloudFormation modelos para configurar rapidamente CloudWatch métricas para o Amazon Keyspaces. O uso desse modelo permitirá que você comece com mais facilidade, fornecendo CloudWatch painéis pré-criados implantáveis com métricas comumente usadas.

<https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates>.

Usando 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>

Usando 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) Linguagem de consulta Cassandra (CQL) para conversor CloudFormation

Esse pacote implementa uma ferramenta de linha de comando para converter scripts do Apache Cassandra Query Language (CQL) em CloudFormation modelos (CloudFormation), o que permite que os esquemas do Amazon Keyspaces sejam facilmente gerenciados em pilhas. 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 DataStax Java 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 read/write grandes objetos para obter um desempenho mais rápido e reduzir a taxa de transferência e os custos de 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>.

Práticas recomendadas de design e arquitetura com o Amazon Keyspaces (para Apache Cassandra)

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

- [Principais diferenças e princípios de design do NoSQL](#)
 - [Diferenças entre design relacional de dados e NoSQL](#)
 - [Dois conceitos-chave para design em NoSQL](#)
 - [Abordar o design em NoSQL](#)
- [Otimize as conexões do driver do cliente para o ambiente sem servidor](#)
 - [Como as conexões funcionam no Amazon Keyspaces](#)
 - [Como configurar conexões no Amazon Keyspaces](#)
 - [Como configurar a política de nova tentativa para 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](#)
- [Melhores práticas de modelagem de dados: recomendações para projetar modelos de dados](#)
 - [Como usar chaves de partição de forma eficaz no Amazon Keyspaces](#)
 - [Use fragmentação de gravação para distribuir uniformemente as workloads entre as partições](#)
 - [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 alvo \(<= 50%\)](#)
 - [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 para otimizar os custos do Amazon Keyspaces.](#)
 - [Como identificar recursos não utilizados](#)
 - [Identificar recursos de tabela não utilizados](#)
 - [Limpar recursos de tabela não utilizados](#)
 - [Como limpar backups não utilizados ou recuperação para um ponto no tempo \(PITR\)](#)
- [Avaliar seus padrões de uso de tabelas para otimizar o desempenho e o custo](#)
 - [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](#)

Principais diferenças e princípios de design do NoSQL

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

- [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 ajuste de escala, 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 com antecedência quais serão os picos de carga de consulta pode ajudar a determinar como particionar os dados para melhor usar a I/O capacidade.

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. Também é importante que um grande volume de consultas não se concentre em uma parte do banco de dados, onde elas podem exceder a I/O capacidade. 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.

Otimize as conexões do driver do cliente para o ambiente sem servidor

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.

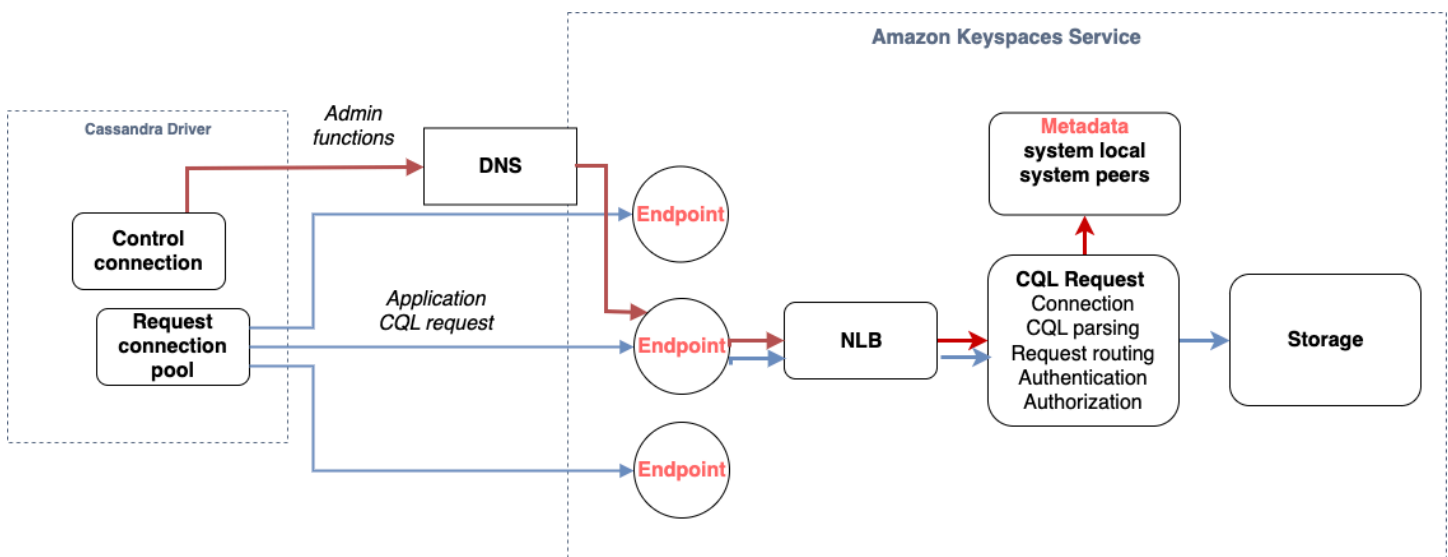
Tópicos

- [Como as conexões funcionam no Amazon Keyspaces](#)
- [Como configurar conexões no Amazon Keyspaces](#)
- [Como configurar a política de nova tentativa para 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 a política de nova tentativa para conexões no Amazon Keyspaces

Ao configurar a política de repetição para uma conexão com o Amazon Keyspaces, recomendamos que você implemente a política de repetição do Amazon Keyspaces `AmazonKeyspacesExponentialRetryPolicy`. Essa política de repetição é mais adequada para tentar novamente em conexões diferentes com o Amazon Keyspaces do que o `DefaultRetryPolicy` do driver.

Com o `AmazonKeyspacesExponentialRetryPolicy`, você pode configurar o número de tentativas de nova conexão que atendam às suas necessidades. Por padrão, o número de tentativas de repetição do `AmazonKeyspacesExponentialRetryPolicy` é definido como 3.

Uma vantagem adicional é que a política de repetição do Amazon Keyspaces devolve a exceção original retornada pelo serviço, o que indica por que a tentativa de solicitação falhou. A política de repetição padrão retorna apenas o `NoHostAvailableException` genérico, o que pode ocultar informações sobre a falha da solicitação.

Para configurar a política de repetição de solicitações usando o `AmazonKeyspacesExponentialRetryPolicy`, recomendamos que você configure um pequeno número de novas tentativas e trate todas as exceções retornadas no código do aplicativo.

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

Como configurar conexões por meio de endpoints da VPC no Amazon Keyspaces

Ao se conectar por meio de endpoints da VPC privados, você provavelmente tem 3 endpoints disponíveis. O número de endpoints da VPC 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 exemplo a seguir. 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 obter 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 "Configure o 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"));

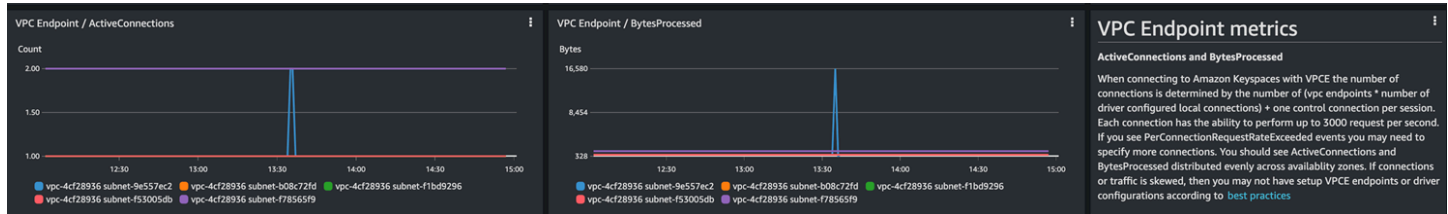
logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

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 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](#).

Melhores práticas de modelagem de dados: recomendações para projetar modelos de dados

A modelagem de dados eficaz é crucial para otimizar o desempenho e minimizar os custos ao trabalhar com o Amazon Keyspaces (para Apache Cassandra). Este tópico aborda as principais

considerações e recomendações para criar modelos de dados que se adequem aos padrões de acesso aos dados do seu aplicativo.

- **Design da chave de partição:** a chave de partição desempenha um papel fundamental na determinação de como os dados são distribuídos entre as partições no Amazon Keyspaces. A escolha de uma chave de partição apropriada pode afetar significativamente o desempenho da consulta e os custos de throughput. Esta seção discute estratégias para criar chaves de partição que promovam a distribuição uniforme da atividade de leitura e gravação entre partições.
- **Considerações importantes:**
 - **Distribuição uniforme de atividades:** busque uma atividade uniforme de leitura e gravação em todas as partições para minimizar os custos de throughput e aproveitar a capacidade de expansão de forma eficaz.
 - **Padrões de acesso:** alinhe o design da chave de partição com os padrões primários de acesso aos dados do seu aplicativo.
 - **Tamanho da partição:** evite criar partições muito grandes, pois isso pode afetar o desempenho e aumentar os custos.

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 de seu aplicativo em todas as partições ajuda a minimizar os custos de taxa de transferência, e isso se aplica tanto aos modos de

capacidade sob demanda quanto aos modos de capacidade provisionada. read/write 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.

Quando uma partição apresenta alta taxa de transferência sustentada de leitura ou gravação, dependendo dos padrões de tráfego, o Amazon Keyspaces pode dividir automaticamente a partição em duas novas partições. Cada nova partição contém um subconjunto das linhas da partição original, distribuindo a taxa de transferência uniformemente em ambas as partições.

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 “Use a capacidade de expansão”](#).

Tópicos

- [Use fragmentação de gravação para distribuir uniformemente as workloads entre as partições](#)

Use fragmentação de gravação para distribuir uniformemente as workloads entre as partições

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.

Um cálculo simples provavelmente seria suficiente, como o produto dos valores dos pontos de UTF-8 código 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 para otimizar os custos do Amazon Keyspaces.](#)
- [Avaliar seus padrões de uso de tabelas para otimizar o desempenho e o custo](#)
- [Avaliar sua capacidade provisionada para o provisionamento do tamanho certo](#)

Avaliar seus custos no nível da tabela

A ferramenta Cost Explorer encontrada no Console de gerenciamento da AWS 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 de custos de tabelas individuais 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](#)

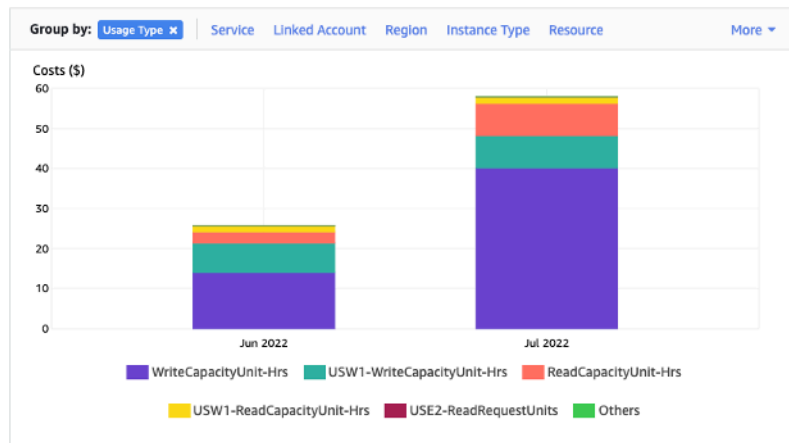
Como visualizar os custos de uma única tabela do Amazon Keyspaces

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 dimensionar a tabela para calcular o custo mensal de armazenamento para a 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 ajuste de escala 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 obter mais informações, consulte [the section called “Como trabalhar 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 obter mais informações, consulte [the section called “Crie relatórios de alocação de custos”](#).

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 obter mais informações, consulte [the section called “Configurar modos read/write de capacidade”](#).

On-demand modo de capacidade

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).

On-demand as tabelas são cobradas contando o número de solicitações reais em relação à tabela, então você paga apenas pelo que usa 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.

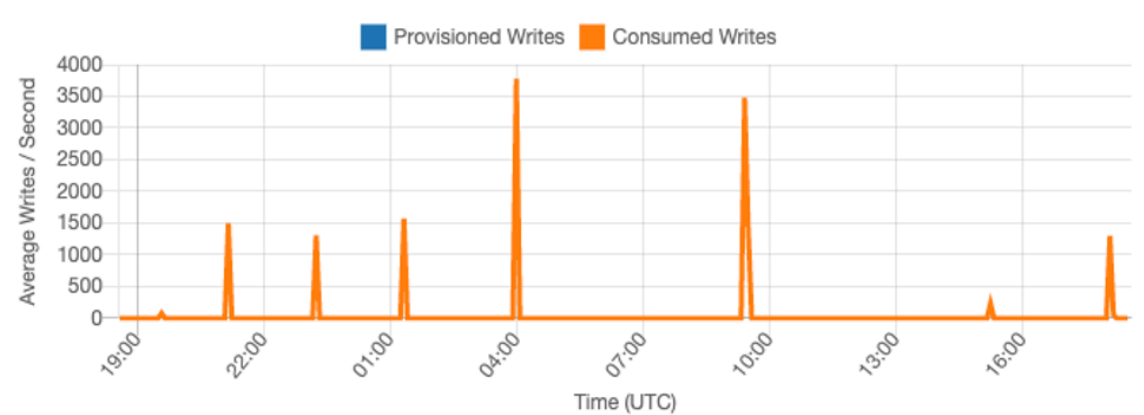
O modo de capacidade provisionada exige um equilíbrio entre não superprovisionar ou subprovisionar a tabela para obter a baixa ocorrência de erros de capacidade de throughput insuficiente e os custos otimizados.

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.

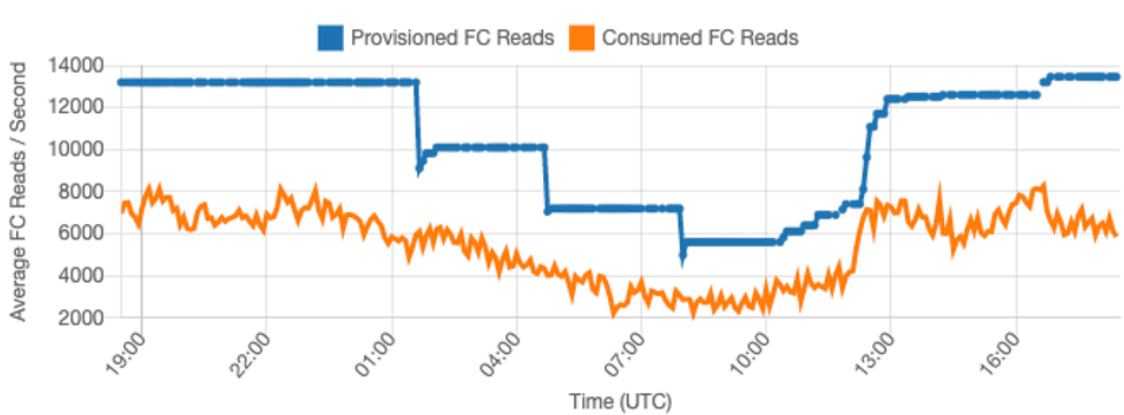
Como as tabelas sob demanda são cobradas por solicitação, não há mais nada a fazer no nível da tabela para otimizar o custo. É necessário avaliar regularmente as tabelas sob demanda para verificar se a carga de trabalho ainda tem as características acima. Caso a carga de trabalho esteja estabilizada, 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:

- Predicible/cyclical tráfego por uma determinada hora ou 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 alvo \(\$\leq 50\%\$ \)](#)
- [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 alvo ($\leq 50\%$)

Você pode usar o AWS CLI ou Console de gerenciamento da AWS para monitorar e identificar as políticas do TargetValues Application Auto Scaling em seus recursos do Amazon Keyspaces.

Note

Ao usar tabelas multirregionais no modo de capacidade provisionada com o ajuste de escala automático do Amazon Keyspaces, certifique-se de usar as operações da API do Amazon Keyspaces para configurar o ajuste de escala automático. As operações subjacentes da API Application Auto Scaling que o Amazon Keyspaces chama em seu nome não têm recursos multirregionais. Para obter mais informações, consulte [the section called “Exibir as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#).

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:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
      "PolicyName": "$<full-table-name>",
      "ServiceNamespace": "cassandra",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:WriteCapacityUnits",
      "PolicyType": "TargetTrackingScaling",
      "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 70.0,
        "PredefinedMetricSpecification": {
          "PredefinedMetricType": "KeyspacesWriteCapacityUtilization"
        }
      },
      "Alarms": [
        ...
      ],
      "CreationTime": "2022-03-04T16:23:48.641000+10:00"
    },
    {
      "PolicyARN": "arn:aws:autoscaling:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
      "PolicyName": "$<full-table-name>",
      "ServiceNamespace": "cassandra",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:ReadCapacityUnits",
      "PolicyType": "TargetTrackingScaling",
      "TargetTrackingScalingPolicyConfiguration": {
```

```
        "TargetValue": 70.0,
        "PredefinedMetricSpecification": {
            "PredefinedMetricType": "KeyspacesReadCapacityUtilization"
        },
        "Alarms": [
            ...
        ],
        "CreationTime": "2022-03-04T16:23:47.820000+10:00"
    }
]
```

Console de gerenciamento da AWS

1. Faça login no Console de gerenciamento da AWS e navegue até a página de CloudWatch serviço em [Introdução ao Console de gerenciamento da AWS](#). 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:

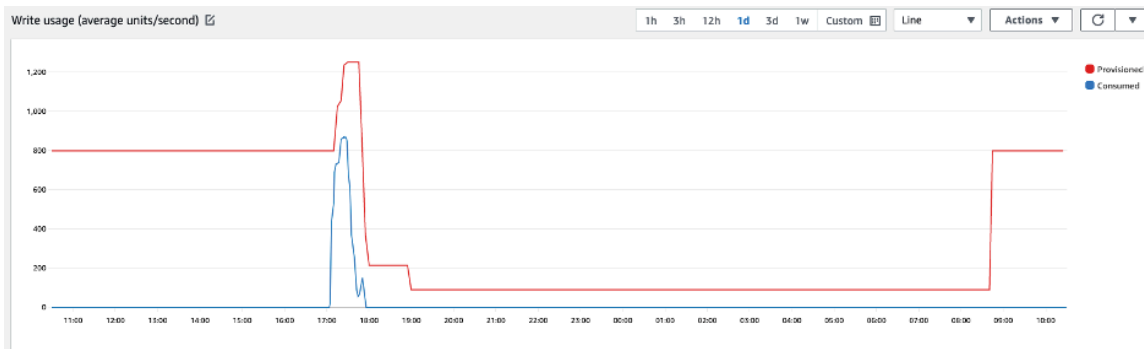
```
{
  "ScheduledActions": [
    {
      "ScheduledActionName": "my-5-8-scheduled-down-action",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:111122223333:scheduledAction:<uuid>:resource/keyspaces/table/table-
name:scheduledActionName/my-5-8-scheduled-down-action",
      "ServiceNamespace": "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:us-east-1:111122223333: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 ajuste de escala 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](#). On-demand as tabelas são perfeitas para cargas de trabalho com picos nos 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 para otimizar os custos do Amazon Keyspaces.

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](#)
- [Como limpar backups não utilizados ou recuperação para um ponto no tempo \(PITR\)](#)

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 Console de gerenciamento da AWS , 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"
},
```

Console de gerenciamento da AWS

As etapas a seguir permitem avaliar a utilização do seu recurso por meio do Console de gerenciamento da AWS.

1. Faça login no Console de gerenciamento da AWS e navegue até a página CloudWatch de serviço em <https://console.aws.amazon.com/cloudwatch/>. Selecione a 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 read/write de capacidade controla como você é cobrado pela taxa de transferência 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 Console de gerenciamento da AWS ou do AWS Command Line Interface.

Como limpar backups não utilizados ou recuperação para um ponto no tempo (PITR)

O Amazon Keyspaces oferece Point-in-time recuperação, 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 [the section called “Backup e restauração com recuperação para um ponto no tempo”](#) para saber se as tabelas têm backups habilitados que podem não ser mais necessários.

Avaliar seus padrões de uso de tabelas para otimizar o desempenho e o custo

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
ReadThrottleEvents	WriteThrottleEvents

Você pode fazer isso por meio do AWS CLI ou do Console de gerenciamento da AWS.

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	2022-02-21 T23:00:00
<end-time>	O final do intervalo de avaliação, especificado no formato ISO8601	2022-02-22 T06:00:00

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,
        40.947256944444445,
        25.618194444444444,
        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.

Console de gerenciamento da AWS

1. Faça login no Console de gerenciamento da AWS e navegue até a página de CloudWatch serviço em [Introdução ao Console de gerenciamento da AWS](#). 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 sua tabela 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 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 $100 * () m1 / \text{provisionedWCU}$.
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 intermitência](#) é um recurso do Amazon Keyspaces que permite que os clientes consumam temporariamente mais do que o provisionado originalmente (RCUs/WCUs mais do que a taxa de transferência provisionada por segundo que foi definida para a 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, consulte [the section called “Use a capacidade de expansão”](#). Assim que as RCUs e WCUs não utilizadas forem esgotadas, você pode experimentar 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](#).

Solução de problemas do Amazon Keyspaces (para Apache Cassandra)

Este guia aborda as etapas de solução de problemas para vários cenários ao trabalhar com o Amazon Keyspaces (para Apache Cassandra). Ele inclui informações sobre como resolver erros gerais, problemas de conexão, problemas de gerenciamento de capacidade e erros de linguagem de definição de dados (DDL).

- Erros gerais
 - Solução de problemas de exceções de alto nível como `NoNodeAvailableException`, `NoHostAvailableException` e `AllNodesFailedException`.
 - Como isolar erros subjacentes das exceções do driver Java.
 - Como implementar políticas de repetição e configurar conexões corretamente.
- Problemas de conexão
 - Como resolver erros ao se conectar aos endpoints do Amazon Keyspaces usando drivers de cliente `cqlsh` ou Apache Cassandra.
 - Solução de problemas de conexões de endpoints VPC, Cassandra-stress conexões e erros de configuração do IAM.
 - Como lidar com perdas de conexão durante a importação de dados.
- Erros de gerenciamento de capacidade
 - Como reconhecer e resolver erros de capacidade insuficiente relacionados a tabelas, partições e conexões.
 - Monitoramento de métricas relevantes do Amazon Keyspaces no Amazon CloudWatch Logs.
 - Como otimizar as conexões e o throughput para melhorar o desempenho.
- Erros de linguagem de definição de dados (DDL)
 - Como solucionar erros ao criar, acessar ou restaurar tabelas e espaços de chaves.
 - Como lidar com falhas relacionadas a configurações de tempo de vida (TTL) personalizadas, limites de coluna e exclusões de intervalo.
 - Considerações sobre workloads de exclusão pesada.

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

- [Como solucionar problemas gerais no Amazon Keyspaces](#)
- [Como solucionar problemas de conexões no Amazon Keyspaces](#)
- [Como solucionar erros de gerenciamento de capacidade no Amazon Keyspaces](#)
- [Como solucionar erros da linguagem de definição de dados no Amazon Keyspaces](#)

Como solucionar 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 ao estabelecer a conexão de controle ou ao realizar solicitações `read/write/prepare/execute/batch`.

Quando o erro ocorre ao estabelecer a conexão de controle, é um sinal de que todos os pontos de contato especificados em seu aplicativo estão inacessíveis. Quando o erro ocorre durante a execução de `read/write/prepare/execute` queries, 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 `prepare/execute` operações `read/write` /. 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, consulte [the 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 falhada `read/write /prepare`. Para configurar `idempotence` para `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 de forma programática para uma única solicitação em seu aplicativo Java.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");  
s.setIdempotent(true);
```

Ou é possível definir a idempotência padrão para todo o aplicativo Java de forma programática, 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 o `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 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. Para isso, use a consulta a seguir 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 endpoints da VPC, consulte [the section called “Conexões de endpoint da VPC”](#).

Para obter um tutorial detalhado que mostra como estabelecer uma conexão ao Amazon Keyspaces usando o driver Datastax 4.x Cassandra, consulte [the section called “Plugin de autenticação para Java 4.x”](#).

Como solucionar 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 endpoints da VPC”](#).

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”](#) 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 SSL/TLS certificado está ausente, o que resulta no seguinte erro.

```
# 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. Para obter mais informações, consulte [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

```
certfile = path_to_file/keysaces-bundle.pem
```

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/keyspaces-bundle.pem
#

# 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. Para obter mais informações, consulte [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **[X509] PEM lib**.

Esse pode ser o caso se o pem arquivo de SSL/TLS certificado não for válido, o que resultará no seguinte erro.

```
# 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, verifique se você baixou os certificados digitais necessários. Para obter mais informações, consulte [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

Você está tentando se conectar ao Amazon Keyspaces usando cqlsh, mas recebe um erro **SSL unknown**.

Esse pode ser o caso se o pem arquivo do SSL/TLS certificado estiver vazio, o que resultará no seguinte erro.

```
# 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, verifique se você baixou os certificados digitais necessários. Você pode confirmar isso usando as etapas do tópico a seguir [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

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 SSL/TLS certificado não puder ser verificado, o que resultará no seguinte erro.

```
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, verifique se você baixou os certificados digitais necessários. Você pode confirmar isso usando as etapas do tópico a seguir [the section called “Como configurar manualmente as conexões cqlsh para o TLS”](#).

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, verifique se sua instância do Amazon EC2 tem uma regra de saída que permite o tráfego para o Amazon Keyspaces. Se esse não for o caso, você precisa criar um grupo de segurança da 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, `CQLSH/CASSANDRA` escolha no menu suspenso Tipo.

Depois de criar o grupo de segurança com a regra de tráfego de saída, você precisa adicioná-lo à instância. Selecione a instância e, em seguida, escolha Ações, Segurança e Mudar grupos de segurança. Adicione o 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 exibir 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 “Identity-based exemplos de políticas”](#).

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 **USERNAME** e **PASSWORD** em seu código corresponde 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 Suporte.

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 erro a seguir.

```
[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 da pulsação e diminua-o para 30 segundos, se for maior que isso.

```
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 “Crie 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 “Crie 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 algum dos componentes necessários para a assinatura estiver errado ou não estiver definido corretamente para a sessão.

- `AWS_ACCESS_KEY_ID`
- `AWS_SECRET_ACCESS_KEY`
- `AWS_DEFAULT_REGION`

Veja a seguir um exemplo de chaves de acesso inválidas.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
  {'11.234.11.234:9142':
    AuthenticationFailed('Failed to authenticate to 11.234.11.234:9142: Error from server:
    code=0100
    [Bad credentials] message="Authentication failure: Invalid signature"')})
```

Para resolver esse problema, verifique se as chaves de acesso e as Região da AWS foram configuradas corretamente para que o plug-in SigV4 acesse o Amazon Keyspaces. Para saber mais sobre as chaves de acesso e o complemento SigV4, consulte [the section called “Crie 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 tutorial passo a passo para aprender como configurar uma conexão por meio de endpoints da VPC de interface para Amazon Keyspaces, consulte [the section called “Conexão com endpoints da VPC”](#).

1. Confirme se a entidade IAM usada para se conectar ao Amazon Keyspaces tem read/write acesso 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 IPv4 privados ao se conectar a partir de uma rede IPv4, dependendo da configuração da VPC e da região. AWS

```
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 a um driver padrão, o usuário deve ter pelo menos SELECT acesso às tabelas do sistema, pois a maioria dos drivers lê o sistema keyspaces/tables quando estabelece 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.

Como solucionar erros 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. Insufficient-capacity eventos 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 obter mais informações, consulte [the section called “Configurar modos read/write de capacidade”](#).
- **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 obter 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.

Saiba como configurar conexões para o Amazon Keyspaces, consulte [the section called “Como configurar conexões”](#). Para obter mais informações sobre como otimizar conexões nos endpoints da VPC, 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 3.000 RCU/RRU por segundo. Para mitigar essas exceções, recomendamos que você revise seu modelo de dados para distribuir o read/write tráfego em mais partições.
- `WriteThrottleEvents`: solicitações para Amazon Keyspaces que excedem a capacidade de gravação de uma tabela.

Para saber mais sobre CloudWatch, consulte [the 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 obter 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 obter mais informações, consulte [the section called “Gerencie a capacidade de throughput com o ajuste de escala automático”](#).
- Escolha o modo de On-demand capacidade para a mesa. Para obter mais informações, consulte [the section called “Configurar o 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 `StoragePartitionThroughputCapacityExceeded`, a capacidade da partição é temporariamente excedida. Isso pode ser controlado automaticamente pela capacidade adaptativa ou pela capacidade sob demanda. Recomendamos revisar seu modelo de dados para distribuir o read/write tráfego em mais partições para mitigar esses erros. As partições de armazenamento do Amazon Keyspaces podem suportar até 1.000 WCU/WRU por segundo e 3.000 RCU/RRU por segundo. Para saber mais sobre como melhorar seu modelo de dados para distribuir o read/write tráfego 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. Para obter mais informações, consulte .

[the section called “Como configurar políticas de repetição”](#).

- 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 a read/write produtividade. Para obter 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 obter 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 obter instruções detalhadas, consulte [the 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 tabela, consulte [the 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 em [the section called “Estimar o tamanho da linha”](#).

Como solucionar erros 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 Console de gerenciamento da AWS, 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 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 recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces, mas a restauração falha](#)
- [Estou tentando usar INSERT/UPDATE para editar configurações personalizadas de Time to Live \(TTL\), 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 obter mais informações, consulte [the section called “Verifique o status da criação do keyspace”](#).

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 obter mais informações, consulte [the section called “Verifique o status da criação da tabela”](#).

Estou tentando restaurar uma tabela usando a recuperação para um ponto no tempo (PITR) do Amazon Keyspaces, mas a restauração falha

Se você está tentando restaurar uma tabela do Amazon Keyspaces com recuperação para um ponto no tempo (PITR) e vê o processo de restauração começar, mas não ser concluído com êxito, talvez você não tenha configurado todas as permissões necessárias para o processo de restauração para esta 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 “Configurar permissões do IAM para restauração”](#).

Estou tentando usar INSERT/UPDATE para editar configurações personalizadas de Time to Live (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 “Atualizar TTL personalizada da tabela”](#).

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 obter 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 obter 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 obter 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.

Monitoramento do Amazon Keyspaces (para Apache Cassandra)

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do Amazon Keyspaces e de suas outras AWS soluções. AWS fornece as seguintes ferramentas de monitoramento para monitorar o Amazon Keyspaces, relatar quando algo está errado e realizar ações automáticas quando apropriado:

- O Amazon Keyspaces oferece um painel pré-configurado que Console de gerenciamento da AWS mostra a latência e os erros agregados em todas as tabelas da conta.
- A Amazon CloudWatch monitora seus AWS recursos e os aplicativos em que você executa 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 CloudWatch do usuário da Amazon](#).
- Os CloudWatch alarmes da Amazon monitoram uma única métrica durante um período que você especifica e executam uma ou mais ações com base no valor da métrica em relação a um determinado limite em 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.

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, consulte [Monitorando o Amazon Keyspaces com a Amazon CloudWatch](#).

- O Amazon CloudWatch Logs permite que você monitore, armazene e acesse seus arquivos de log a partir de tabelas e outras fontes do Amazon Keyspaces. CloudTrail CloudWatch Os registros podem monitorar as informações nos arquivos de log e notificá-lo quando determinados limites forem atingidos. É possível também 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 realizados por sua Conta da AWS ou em nome dela e entrega os arquivos de log a um bucket do Amazon S3 que você especificar. Você pode identificar quais usuários e contas chamaram AWS, o endereço IP de origem de onde as chamadas foram feitas e quando elas ocorreram. Para mais informações, consulte o [Guia do usuário do AWS CloudTrail](#).

EventBridge da Amazon é um serviço de ônibus de eventos sem servidor que facilita a conexão de seus aplicativos com dados de várias fontes. EventBridge fornece um fluxo de dados em tempo real de seus próprios aplicativos, aplicativos Software-as-a-Service (SaaS) e AWS serviços e encaminha esses dados para destinos como o Lambda. Isso permite monitorar eventos que ocorram em serviços e criem arquiteturas orientadas a eventos. Para obter mais informações, consulte o [Guia EventBridge do usuário da Amazon](#).

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 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](#)
- [Visualizando as métricas do Amazon Keyspaces em CloudWatch](#)
- [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 <code>SystemErrors</code> para saber 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, você deve 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 1. Defina o Período para <code>ConsumedReadCapacityUnits</code> e <code>ProvisionedReadCapacityUnits</code> para o intervalo que você deseja monitorar. 2. Altere a Estatística para <code>ConsumedReadCapacityUnits</code> de Average a Sum. 3. Crie uma nova Expressão matemática vazia. 4. Na seção Detalhes da nova expressão matemática, insira o ID de <code>ConsumedReadCapacityUnits</code> e divida a métrica pela função CloudWatch PERÍODO da métrica (<code>metric_id/ (PERIOD (metric_id))</code>). 5. Desmarque <code>ConsumedReadCapacityUnits</code>.

Como?	Métricas relevantes
	Agora você pode comparar sua capacidade média de leitura consumida com sua capacidade provisionada. Para obter mais 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 1. Defina o Período para <code>ConsumedWriteCapacityUnits</code> e <code>ProvisionedWriteCapacityUnits</code> para o intervalo que você deseja monitorar. 2. Altere a Estatística para <code>ConsumedWriteCapacityUnits</code> de <code>Average</code> a <code>Sum</code>. 3. Crie uma nova Expressão matemática vazia. 4. Na seção Detalhes da nova expressão matemática, insira o ID de <code>ConsumedWriteCapacityUnits</code> e divida a métrica pela função <code>CloudWatch PERÍODO da métrica (metric_id/ (PERIOD (metric_id)))</code>. 5. Desmarque <code>ConsumedWriteCapacityUnits</code>. 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 métricas e dimensões para a Amazon CloudWatch. Todas as métricas são agregadas e relatadas a cada minuto. As seções a seguir são uma visão geral de todas as métricas disponíveis para o Amazon Keyspaces.

Tópicos

- [Métricas do Amazon Keyspaces](#)
- [Dimensões para métricas do Amazon Keyspaces](#)

- [Métricas da captura de dados de alteração \(CDC\) do Amazon Keyspaces](#)
- [Dimensões do Amazon Keyspaces alteram métricas de captura de dados \(CDC\)](#)

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	Description
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. Unidades: Count Estatística válida: • Maximum: o número máximo de unidades de capacidade e 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. Unidades: Count Estatística válida:

Métrica	Description
	• Maximum: o número máximo de unidades de capacidade e 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. Unidades: Percent Estatística válida: • Maximum: o percentual máximo de unidades de capacidade de leitura provisionada utilizadas pela conta. • Minimum: o percentual mínimo de unidades de capacidade de leitura provisionada utilizadas pela conta. • 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	Description
AccountProvisionedWriteCapacityUtilization	O percentual de unidades de capacidade de gravação provisionada utilizadas por uma conta. Unidades: Percent Estatística válida: • Maximum: a porcentagem máxima de unidades de capacidade de gravação provisionada utilizadas pela conta. • Minimum: a porcentagem mínima de unidades de capacidade de gravação provisionada utilizadas pela conta. • 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	Description
<code>BillableTableSizeInBytes</code>	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. Unidades: Bytes Dimensões: Keyspace, TableName Estatística válida: • <code>Maximum</code>: o tamanho máximo de armazenamento da tabela. • <code>Minimum</code>: o tamanho mínimo de armazenamento da tabela. • <code>Average</code>: o tamanho médio de armazenamento da tabela. Essa métrica é calculada em intervalos de 4 a 6 horas.

Métrica	Description
<code>ConditionalCheckFailedRequests</code>	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, <code>ConditionalCheckFailedRequests</code> será incrementado em um. As verificações de condição avaliadas como falsas consomem unidades de capacidade e de gravação com base no tamanho da linha. Para obter mais informações, consulte the section called “Estimar o consumo de capacidade de LWT”. Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Minimum • Maximum • Average • SampleCount • Sum

Métrica	Description
ConsumedReadCapacityUnits	O número de unidades de capacidade de leitura consumidas ao longo do período especificado. Para obter mais informações, consulte o modo Read/Write de capacidade.  Note 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 ”
Unidades: Count	
Dimensões: Keyspace, TableName	
Estatística válida:	
• Minimum: o número mínimo de unidades de capacidade e de leitura consumidas por qualquer solicitação individual à tabela. • Maximum: o número máximo de unidades de capacidade e de leitura consumidas por qualquer solicitação individual à tabela.	

Métrica	Description
	• Average: a capacidade de leitura por solicitação média consumida.  Note O valor Average é influenciado por períodos de inatividade em que o valor da amostra será zero. • Sum: o total de unidades de capacidade de leitura consumidas. Essa é a estatística mais útil para a métrica <code>ConsumedReadCapacityUnits</code>. • SampleCount : o número de solicitações ao Amazon Keyspaces, mesmo que nenhuma capacidade de leitura tenha sido consumida.  Note O valor SampleCount é influenciado por períodos de inatividade em que o valor da amostra será zero.

Métrica	Description
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 o modo Read/Write de capacidade.  Note 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 ” Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Minimum: o número mínimo de unidades de capacidade e de gravação consumidas por qualquer solicitação individual à tabela. • Maximum: o número máximo de unidades de capacidade e de gravação consumidas por qualquer solicitação individual à tabela.

Métrica	Description
	- Average: a capacidade de gravação por solicitação média consumida.  Note O valor Average é influenciado por períodos de inatividade em que o valor da amostra será zero. - Sum: o total de unidades de capacidade de gravação consumidas. Essa é a estatística mais útil para a métrica ConsumedWriteCapacityUnits . - SampleCount : o número de solicitações ao Amazon Keyspaces, mesmo que nenhuma capacidade de gravação tenha sido consumida.  Note O valor SampleCount é influenciado por períodos de inatividade em que o valor da amostra será zero.

Métrica	Description
MaxProvisionedTableReadCapacityUtilization	O percentual de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta de uma conta. Unidades: Percent Estatística válida: • Maximum: o percentual máximo de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta da conta. • Minimum: o percentual mínimo de unidades de capacidade de leitura provisionada utilizadas pela tabela de leitura provisionada mais alta da conta. • 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	Description
MaxProvisionedTableWriteCapacityUtilization	A porcentagem da capacidade de gravação provisionada utilizada pela tabela de gravação provisionada mais alta de uma conta. Unidades: Percent Estatística válida: • Maximum: a porcentagem máxima de unidades de capacidade de gravação provisionadas utilizadas pela tabela de gravação provisionada mais alta da conta. • Minimum: a porcentagem mínima de unidades de capacidade de gravação provisionadas utilizadas pela tabela de gravação provisionada mais alta da conta. • 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	Description
<code>PerConnectionRequestRateExceeded</code>	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. 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 <code>PerConnectionRequestRateExceeded</code>. Não há limite para o número de conexões que você pode ter no Amazon Keyspaces. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • <code>SampleCount</code> • <code>Sum</code>

Métrica	Description
ProvisionedReadCapacityUnits	O número de unidades de capacidade de leitura provisionada para uma tabela. A dimensão <code>TableName</code> retorna o <code>ProvisionedReadCapacityUnits</code> para a tabela. Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Minimum: a configuração mais baixa para a capacidade e de leitura provisionada. Se você usar <code>ALTER TABLE</code> para aumentar a capacidade de leitura, esta métrica mostrará o valor mais baixo de <code>ReadCapacityUnits</code> provisionado durante esse período. • Maximum: a configuração mais alta para a capacidade e de leitura provisionada. Se você usar <code>ALTER TABLE</code> para diminuir a capacidade de leitura, esta métrica mostrará o valor mais alto de <code>ReadCapacityUnits</code> provisionado durante esse período. • Average: a capacidade média de leitura provisionada. A métrica <code>ProvisionedReadCapacityUnits</code> é publicada para intervalos de cinco minutos. Portanto, se você ajustar rapidamente as unidades de capacidade e de leitura provisionada, essa estatística poderá não refletir a média real.

Métrica	Description
ProvisionedWriteCapacityUnits	O número de unidades de capacidade de gravação provisionada para uma tabela. A dimensão <code>TableName</code> retorna o <code>ProvisionedWriteCapacityUnits</code> para a tabela. Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Minimum: a configuração mais baixa para capacidade e de gravação provisionada. Se você usar <code>ALTER TABLE</code> para aumentar a capacidade de gravação, esta métrica mostrará o valor mais baixo de <code>WriteCapacityUnits</code> provisionado durante esse período. • Maximum: a configuração mais alta para a capacidade e de gravação provisionada. Se você usar <code>ALTER TABLE</code> para diminuir a capacidade de gravação, esta métrica mostrará o valor mais alto de <code>WriteCapacityUnits</code> provisionado durante esse período. • Average: a capacidade média de gravação provisionada. A métrica <code>ProvisionedWriteCapacityUnits</code> 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	Description
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. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • SampleCount • 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. Unidades: Millisecond Dimensões: TableName, ReceivingRegion Estatística válida: • Average • Maximum • Minimum

Métrica	Description
ReturnedItemCountBySelect	O número de linhas retornadas por SELECT consultas de várias linhas durante o período especificado. Multi-row SELECT consultas são consultas que não contêm a chave primária totalmente qualificada, como varreduras completas de tabelas e consultas de intervalo. O número de itens retornados não é necessariamente o mesmo que o número de itens avaliados. Por exemplo, suponha que você tenha solicitado um <code>SELECT *</code> com <code>ALLOW FILTERING</code> em uma tabela que tinha 100 linhas, mas especificou um <code>WHERE</code> que reduziu os resultados para que apenas 15 linhas fossem retornadas. Nesse caso, a resposta de SELECT conterá um <code>ScanCount</code> de 100 e um <code>Count</code> de 15 linhas retornadas. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • Minimum • Maximum • Average • SampleCount • Sum

Métrica	Description
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 3.000 RCU/RRU por segundo. Recomendamos revisar seu modelo de dados para distribuir o read/write tráfego em mais partições para mitigar essas exceções.  Note As partições lógicas do Amazon Keyspaces podem abranger várias partições de armazenamento e são virtualmente ilimitadas em tamanho. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • SampleCount • Sum
SuccessfulRequestCount	O número de solicitações bem-sucedidas processadas durante o período especificado. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • SampleCount

Métrica	Description
SuccessfulRequestLatency	As solicitações bem-sucedidas para o Amazon Keyspaces Streams durante o período especificado. O <code>SuccessfulRequestLatency</code> pode fornecer dois tipos diferentes de informações: • O tempo decorrido para solicitações bem-sucedidas (<code>Minimum</code>, <code>Maximum</code>, <code>Sum</code> ou <code>Average</code>). • O número de solicitações bem-sucedidas (<code>SampleCount</code>). <code>SuccessfulRequestLatency</code> reflete a atividade somente dentro do Amazon Keyspaces e não leva em conta a latência da rede ou a atividade do lado do cliente. Unidades: <code>Milliseconds</code> Dimensões: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estatística válida: • <code>Minimum</code> • <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code>

Métrica	Description
SystemErrors	As solicitações para o Amazon Keyspaces que geram um <code>ServerError</code> durante o período de tempo especificado. Um <code>ServerError</code> geralmente indica um erro de serviço interno. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • Sum • SampleCount
SystemReconciliationDeletes	As unidades consumidas para excluir dados gravados com marca de exclusão quando a data e a hora do lado do cliente estão habilitadas. Cada <code>SystemReconciliationDelete</code> 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 <code>SystemReconciliationDeletes</code>. Ou, para excluir uma linha inteira que contém 3,5 KB de dados, são necessárias 4 <code>SystemReconciliationDeletes</code>. Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Sum: o número total de <code>SystemReconciliationDeletes</code> consumidos em um período.

Métrica	Description
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. Unidades: Count Dimensões: Keyspace, TableName Estatística válida: • Sum: o número total de TTLDeletes consumidos em um período.

Métrica	Description
UserErrors	Solicitações para o Amazon Keyspaces que geram um erro <code>InvalidRequest</code> durante o período de tempo especificado. <code>InvalidRequest</code> os erros geralmente representam um erro do lado do cliente, por exemplo, uma combinação inválida de parâmetros, uma tentativa de atualizar uma tabela inexistente ou uma assinatura de solicitação incorreta. <code>UserErrors</code> representam o agregado de solicitações inválidas para a atual Região da AWS e a atual. Conta da AWS Unidades: Count Dimensões: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estatística válida: • <code>Sum</code> • <code>SampleCount</code>
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. Unidades: Count Dimensões: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estatística válida: • <code>SampleCount</code> • <code>Sum</code>

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	Description
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.

Métricas da captura de dados de alteração (CDC) do Amazon Keyspaces

As métricas a seguir estão disponíveis em sua CloudWatch conta para monitorar chamadas de API para o endpoint de captura de dados de alteração (CDC) do Amazon Keyspaces.

Métrica	Description
ReturnedRecordsCount	O número de registros de fluxo retornados pelas <code>GetRecords</code> operações durante o período especificado. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • Minimum

Métrica	Description
	• <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code> • <code>Sum</code>
<code>SuccessfulRequestLatency</code>	As solicitações bem-sucedidas para o Amazon Keyspaces Streams durante o período especificado. O <code>SuccessfulRequestLatency</code> pode fornecer dois tipos diferentes de informações: • O tempo decorrido para solicitações bem-sucedidas (<code>Minimum</code>, <code>Maximum</code>, <code>Sum</code> ou <code>Average</code>). • O número de solicitações bem-sucedidas (<code>SampleCount</code>). <code>SuccessfulRequestLatency</code> reflete a atividade somente nas APIs CDC do Amazon Keyspaces e não leva em conta a latência da rede ou a atividade do lado do cliente. Unidades: <code>Milliseconds</code> Dimensões: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estatística válida: • <code>Minimum</code> • <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code>

Métrica	Description
<code>SystemErrors</code>	As solicitações ao Amazon Keyspaces CDC que geram um código de HTTP 500 status durante o período de tempo especificado. Um código de HTTP 500 status geralmente indica um erro interno de serviço. Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • Sum • SampleCount
<code>UserErrors</code>	Solicitações ao Amazon Keyspaces CDC que geram um código de HTTP 400 status durante o período de tempo especificado. <code>UserErrors</code> representa o agregado de HTTP 400 erros do Amazon Keyspaces CDC no atual e no Região da AWS atual. Conta da AWS Unidades: Count Dimensões: Keyspace, TableName, Operation Estatística válida: • Sum • SampleCount

Dimensões do Amazon Keyspaces alteram métricas de captura de dados (CDC)

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	Description
<code>StreamLabel</code>	Essa dimensão limita os dados a um fluxo específico de captura de dados de alteração (CDC) quando <code>KeyspaceName</code> e <code>TableName</code> são especificados.
<code>Keyspace</code>	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.
<code>TableName</code>	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 <code>Keyspace</code> .
<code>Operation</code>	Essa dimensão limita os dados a uma das seguintes operações de API do Amazon Keyspaces Streams: <code>GetRecords</code> , e, <code>GetStream</code> , <code>GetShardIterator</code> , <code>ListStreams</code> .

Visualizando as métricas do Amazon Keyspaces em CloudWatch

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 AWS. 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"
```

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 o [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 o seguinte:

- 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 [Elemento userIdentity do CloudTrail](#).

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	Declaração	Ação de CQL	AWS Ação do SDK
CreateKeyspace	DDL	CREATE KEYSPACE	CreateKeyspace
AlterKeyspace	DDL	ALTER KEYSPACE	UpdateKeyspace
DropKeyspace	DDL	DROP KEYSPACE	DeleteKeyspace
CreateTable	DDL	CREATE TABLE	CreateTable
DropTable	DDL	DROP TABLE	DeleteTable

CloudTrail eventName	Declaração	Ação de CQL	AWS Ação do SDK
AlterTable	DDL	ALTER TABLE	UpdateTable , TagResource , UntagResource
CreateUdt	DDL	CREATE TYPE	CreateType
DropUdt	DDL	DROP TYPE	DeleteType
GetStream	DDL	nenhuma ação CQL disponível	GetStream
ListStreams	DDL	nenhuma ação CQL disponível	ListStreams

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](#).

Para obter mais informações e um exemplo que mostra como criar alarmes para eventos de dados, consulte a postagem a seguir no blog do AWS banco de dados [Usando a auditoria DML para Amazon Keyspaces \(para Apache Cassandra\)](#).

A tabela a seguir mostra os eventos de dados registrados CloudTrail por `forCassandra` table.

CloudTrail eventName	Declaração	Ação de CQL	AWS Ações do SDK
Select	DML	SELECT	GetKeyspace , GetTable, GetType,

CloudTrail eventName	Declaração	Ação de CQL	AWS Ações do SDK
			ListKeyspaces, ListTables, ListTypes, 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
GetShardIterator	DML	nenhuma ação CQL disponível	GetShardIterator
GetRecords	DML	nenhuma ação CQL disponível	GetRecords

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"
    }
  },
  "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": "e5aebec-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-4e38-4889-8475-3eab98de0fffd",
"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 na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O [modelo de responsabilidade compartilhada](#) descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços no Nuvem AWS. AWS também fornece serviços que você pode usar 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 na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Também existe a responsabilidade por outros fatores, incluindo a confidencialidade de dados, os requisitos da 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 a usar outros AWS serviços que podem ajudá-lo 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 modelo de [responsabilidade AWS compartilhada O modelo](#) de se aplica à proteção de dados no Amazon Keyspaces (para Apache Cassandra). Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o 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 privacidade de dados, consulte [Perguntas frequentes sobre privacidade de dados](#). Para obter informações sobre proteção de dados na Europa, consulte o [Centro de Regulamento Geral sobre a Proteção de Dados \(RGPD\)](#).

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com Centro de Identidade do AWS IAM ou 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) com cada conta.
- Use SSL/TLS para se comunicar com AWS os recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como [trabalhar com CloudTrail trilhas](#) no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte [Federal Information Processing Standard \(FIPS\) 140-3](#).

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com o Amazon Keyspaces ou outros Serviços da AWS usando o console, a API ou AWS os SDKs. AWS CLI Quaisquer dados inseridos em tags ou em 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, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse 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 criptografa seus dados usando o Advanced Encryption Standard (AES) de 256 bits. Isso ajuda a proteger seus dados contra o acesso não autorizado ao armazenamento subjacente.

O Amazon Keyspaces criptografa e descriptografa os dados em tabelas e streams 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 AWS KMS conceitos como criptografia de envelope, consulte [conceitos AWS KMS de serviços de gerenciamento](#) no Guia do AWS Key Management Service desenvolvedor.

Ao criar uma nova tabela, você pode escolher uma das seguintes chaves AWS KMS (chaves KMS):

- Chave pertencente à AWS — Esse é o tipo 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 (AWS KMS taxas aplicáveis).

O Amazon Keyspaces criptografa automaticamente os fluxos de captura de dados alterados (CDC) com a mesma chave da tabela subjacente. Para obter mais informações sobre o CDC, consulte [the section called “Trabalhando com fluxos do CDC”](#).

Você pode alternar entre a chave gerenciada pelo cliente 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 usando a opção padrão de Chaves pertencentes à AWS é oferecida sem custo adicional. No entanto, as cobranças do AWS KMS se aplicam a chaves gerenciadas pelo cliente. Para saber mais sobre preços, consulte [Preços do AWS KMS](#).

A criptografia em repouso do Amazon Keyspaces está disponível em todas as Regiões da AWS, incluindo as regiões da 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 o Advanced Encryption Standard (AES) 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 se integra com AWS Key Management Service (AWS KMS) para gerenciar a 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 — Esse é o tipo 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 (AWS KMS taxas aplicáveis).

AWS KMS chave (chave KMS)

A criptografia em repouso protege todos os seus dados do Amazon Keyspaces com uma AWS KMS chave. 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.

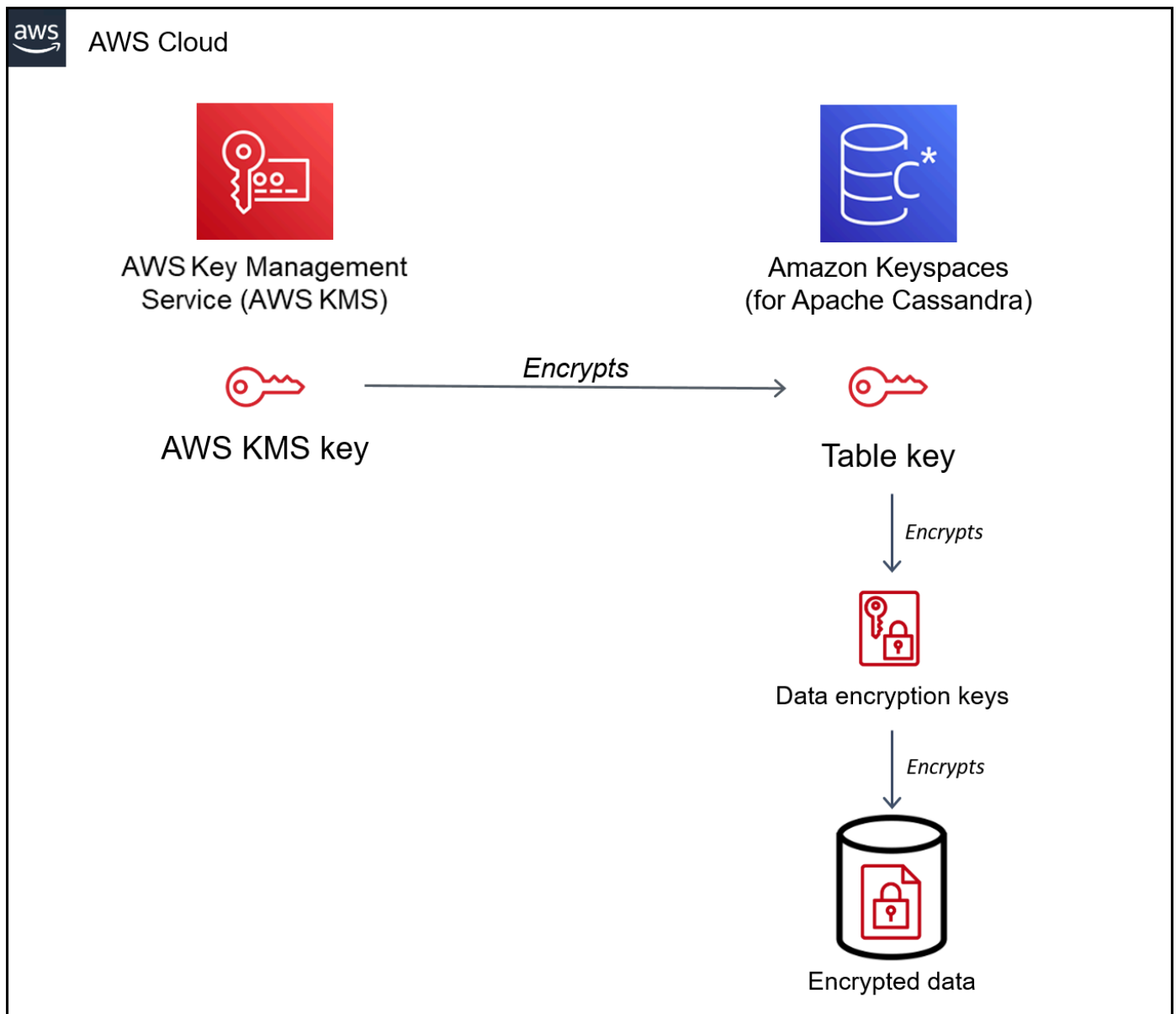
No entanto, você pode criptografar suas tabelas do Amazon Keyspaces usando [uma chave gerenciada pelo cliente](#) na sua Conta da AWS. Você pode selecionar uma chave do KMS diferente para cada tabela em uma keyspace. A chave do KMS selecionada para uma tabela também é usada para criptografar todos os 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 da tabela no Amazon Keyspaces.

Quando você acessa uma tabela criptografada, o Amazon Keyspaces envia uma solicitação para usar a chave KMS AWS KMS para descriptografar a chave da 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 usa e armazena a chave da 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 a chamada AWS KMS para cada operação do Amazon Keyspaces, o Amazon Keyspaces armazena em cache as chaves da tabela de texto simples para cada conexão na memória. Se o Amazon Keyspaces receber uma solicitação para a chave da tabela em cache após cinco minutos de inatividade, ele enviará uma nova solicitação para AWS KMS descriptografar a chave da tabela. Essa chamada captura todas as alterações feitas nas políticas de acesso da chave KMS em AWS KMS ou AWS Identity and Access Management (IAM) desde a última solicitação para descriptografar a chave da 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

- [AWS chaves de propriedade](#)
- [Chaves gerenciadas pelo cliente](#)
- [Notas de uso de criptografia em repouso](#)

AWS chaves de propriedade

Chaves pertencentes à AWS não estão armazenados no seu Conta da AWS. Elas fazem parte de uma coleção de chaves KMS que AWS possui e gerencia para uso em várias Contas da AWS. AWS serviços que você pode Chaves pertencentes à AWS usar para proteger seus dados.

Você não pode visualizar, gerenciar Chaves pertencentes à AWS, usar ou 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 uma taxa mensal ou uma taxa de uso pelo uso de Chaves pertencentes à AWS, e elas não contam nas AWS KMS cotas da sua conta.

Chaves gerenciadas pelo cliente

As chaves gerenciadas pelo cliente são chaves Conta da AWS que você cria, possui 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 Amazon CloudWatch Logs para rastrear as solicitações que o Amazon Keyspaces envia AWS KMS em seu nome. Para obter mais informações, consulte [the section called “Etapa 6: Configurar o monitoramento com AWS CloudTrail”](#).

As chaves gerenciadas pelo cliente são [cobradas](#) por cada chamada de API, e as AWS KMS cotas se aplicam 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 o 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 você 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.

- Server-side a criptografia em repouso está habilitada em todas as tabelas do Amazon Keyspaces e não pode ser desativada. 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

Você pode usar o console ou as instruções CQL para especificar as AWS KMS key para novas tabelas e atualizar as chaves de criptografia das 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.

Etapas 1: criar uma chave gerenciada pelo cliente usando 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

Etapas 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](#)
- [Exemplo de política de chave do](#)
- [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](#)

Exemplo de política de chave do

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 ViaService condição [kms:](#) permite as permissões somente quando a solicitação vem do Amazon Keyspaces em nome dos diretores listados na declaração de 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 coisa específica.
Região da AWS

- 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 um exemplo de política de chaves, substitua os diretores de exemplo pelos diretores reais do seu. 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 restringida pelo contexto de [criptografia do Amazon Keyspaces](#), que inclui o nome da tabela e Conta da AWS o ID. 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 Console de gerenciamento da AWS e abra o console do Amazon Keyspaces em <https://console.aws.amazon.com/keyspaces/home>
2. No painel de navegação, selecione Tabelas e 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 (avançada) e, no campo de pesquisa, escolha AWS KMS key ou insira um nome de 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 AWS Key Management Service desenvolvedor.

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-1:555555555555:key/11111111-1111-1111-1111-111111111111'
    }
};
```

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 AWS Key Management Service desenvolvedor.

Etapas 4: Atualizar a chave de criptografia de uma tabela existente

Você também pode usar o console do Amazon Keyspaces ou o CQL para alterar as chaves de criptografia de uma tabela existente entre uma chave KMS gerenciada pelo cliente Chave pertencente à AWS e uma chave KMS gerenciada pelo cliente a qualquer momento.

Atualizar uma tabela existente com a nova chave gerenciada pelo cliente (console)

1. Faça login no Console de gerenciamento da AWS 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 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 (avançada) e, no campo de pesquisa, escolha uma AWS KMS key ou insira um Nome de recurso da Amazon (ARN).

Note

Se a chave selecionada não for válida, consulte [Solução de problemas de acesso à chave](#) no Guia do AWS Key Management Service desenvolvedor.

Como alternativa, você pode escolher um Chave pertencente à AWS para uma tabela criptografada com uma chave gerenciada pelo cliente.

5. Selecione 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-1:555555555555: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 AWS Key Management Service desenvolvedor.

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 `ALTER TABLE` instrução 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, vincula AWS KMS criptograficamente o contexto de criptografia aos dados criptografados. Para descriptografar os dados, é necessário passar o mesmo contexto de criptografia.

O Amazon Keyspaces usa o mesmo contexto de criptografia em todas as operações AWS KMS criptográficas. 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. Ela também aparece em texto simples em registros, como em registros para o [AWS CloudTrail](#) e [Amazon CloudWatch Logs](#).

Em suas solicitações para AWS KMS, o Amazon Keyspaces usa um contexto de criptografia com três 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 pode usar AWS CloudTrail logs para rastrear as solicitações que o Amazon Keyspaces envia em seu nome. AWS KMS

As solicitações `GenerateDataKey`, `DescribeKey`, `Decrypt`, e `CreateGrant` são discutidas nesta seção. Além disso, o Amazon Keyspaces usa uma [RetireGrant](#) operação para remover uma concessão quando você exclui uma tabela.

 Note

Quando você trabalha com o Amazon Keyspaces, algumas operações podem gerar CloudTrail eventos com um `invokedBy` campo de `dynamodb.amazonaws.com`. Isso é esperado e ocorre porque o Amazon Keyspaces se integra ao Amazon DynamoDB para fornecer seu serviço.

GenerateDataKey

O Amazon Keyspaces cria uma chave de tabela exclusiva para criptografar dados em repouso. Ele envia uma [GenerateDataKey](#) solicitação para AWS KMS que especifique a chave KMS da 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-1:555555555555: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-1:555555555555:key/11111111-1111-111-1111-111111111111"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012",
    "sharedEventID": "84fbaaf0-9641-4e32-9147-57d2cb08792e"
  }
}

```

DescribeKey

O Amazon Keyspaces usa uma [DescribeKey](#) operação para determinar se a chave KMS que você selecionou 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-1:555555555555: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-1:555555555555: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 da tabela, o Amazon Keyspaces envia uma solicitação [Decrypt](#) 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 o principal usuário Conta da AWS que está acessando a tabela. Os parâmetros incluem

a chave da tabela criptografada (como um blob de texto cifrado) e o [contexto de criptografia](#) que identifica a tabela e o. Conta da AWS AWS KMS deriva a 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-1:555555555555: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": "AIDAZ3FNIIIVIZZ6H7CFQG",
    "arn": "arn:aws:iam::arn:aws:kms:eu-west-1:555555555555:key/11111111-1111-111-1111-111111111111:user/admin",
    "accountId": "arn:aws:kms:eu-west-1:555555555555: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-1:555555555555: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

Este tópico descreve como o Amazon Keyspaces (para Apache Cassandra) protege conexões de aplicativos locais com o Amazon Keyspaces e entre o Amazon Keyspaces e outros recursos dentro do mesmo. AWS 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 AWS:

- Uma AWS Site-to-Site VPN conexão. 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 Direct Connect conexão. Para obter mais informações, consulte [O que é o Direct Connect?](#) no Guia do usuário do Direct Connect .

Como um serviço gerenciado, o Amazon Keyspaces (para Apache Cassandra) é protegido pela segurança de rede global. AWS Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte [AWS Cloud Security](#). Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte [Proteção](#) de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o Amazon Keyspaces pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.

- Suítes de criptografia com sigilo direto perfeito (PFS), como DHE (Ephemeral) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

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 AWS CLI, o ou a AWS API. Para obter mais informações, consulte [Como usar o IAM com o Amazon Keyspaces](#).

O segundo método usa um plug-in de autenticação para o driver DataStax Java de código aberto para 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 do Signature Version 4 \(SigV4\) da AWS](#). Para obter mais informações, consulte [the section called “Crie credenciais do IAM para autenticação AWS”](#).

Tráfego entre AWS recursos 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 de VPC de interface são alimentados por AWS PrivateLink, que é um AWS serviço que permite a comunicação privada entre VPCs e serviços. 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 VPC de interface não exigem um gateway de internet, dispositivo NAT, conexão VPN ou conexão. 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](#)
- [Autenticação com identidades](#)
- [Gerenciar o acesso usando políticas](#)
- [Como o Amazon Keyspaces funciona com o IAM](#)
- [Exemplos de políticas baseadas em identidade do Amazon Keyspaces](#)
- [AWS políticas gerenciadas para 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 com base na sua função:

- Usuário do serviço: solicite permissões ao seu administrador se você não conseguir acessar os atributos (consulte [Solução de problemas de identidade e acesso do Amazon Keyspaces](#)).
- Administrador do serviço: determine o acesso do usuário e envie solicitações de permissão (consulte [Como o Amazon Keyspaces funciona com o IAM](#))
- Administrador do IAM: escreva políticas para gerenciar o acesso (consulte [Exemplos de políticas baseadas em identidade do Amazon Keyspaces](#))

Autenticação com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado como usuário do IAM ou assumindo uma função do IAM. Usuário raiz da conta da AWS

Você pode fazer login como uma identidade federada usando credenciais de uma fonte de identidade como Centro de Identidade do AWS IAM (IAM Identity Center), autenticação de login único ou credenciais. Google/Facebook Para ter mais informações sobre como fazer login, consulte [Como fazer login em sua Conta da AWS](#) no Guia do usuário do Início de Sessão da AWS .

Para acesso programático, AWS fornece um SDK e uma CLI para assinar solicitações criptograficamente. Para ter mais informações, consulte [AWS Signature Version 4 para solicitações de API](#) no Guia do usuário do IAM.

Conta da AWS usuário root

Ao criar um Conta da AWS, você começa com uma identidade de login chamada usuário Conta da AWS raiz que tem acesso completo a todos Serviços da AWS os recursos. É altamente recomendável não usar o usuário-raiz em tarefas diárias. Consulte as tarefas que exigem credenciais de usuário-raiz em [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 com permissões específicas para uma única pessoa ou aplicação. É recomendável usar credenciais temporárias, em vez de usuários do IAM com credenciais de longo prazo. Para obter mais informações, consulte [Exigir que usuários humanos usem a federação com um provedor de identidade para acessar AWS usando credenciais temporárias](#) no Guia do usuário do IAM.

Um [grupo do IAM](#) especifica um conjunto de usuários do IAM e facilita o gerenciamento de permissões para grandes conjuntos de usuários. Para ter mais informações, consulte [Casos de uso de usuários do IAM](#) no Guia do usuário do IAM.

Perfis do IAM

Uma [perfil do IAM](#) é uma identidade com permissões específicas que oferece credenciais temporárias. Você pode assumir uma função [mudando de um usuário para uma função do IAM \(console\)](#) ou chamando uma operação de AWS API AWS CLI ou. Para saber mais, consulte [Métodos para assumir um perfil](#) no Manual do usuário do IAM.

Os perfis do IAM são úteis para acesso de usuário federado, permissões de usuário do IAM temporárias, acesso entre contas, acesso entre serviços e aplicações em execução no Amazon EC2. Consulte mais informações em [Acesso a recursos entre contas no IAM](#) no Guia do usuário do IAM.

Gerenciar o acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política define permissões quando associada a uma identidade ou recurso. AWS avalia essas políticas quando um diretor faz uma solicitação. A maioria das políticas é armazenada AWS como documentos JSON. Para ter mais informações sobre documentos de política JSON, consulte [Visão geral das políticas JSON](#) no Guia do usuário do IAM.

Por meio de políticas, os administradores especificam quem tem acesso a que, definindo qual entidade principal pode realizar ações em quais recursos e sob quais condições.

Por padrão, usuários e perfis não têm permissões. Um administrador do IAM cria políticas do IAM e as adiciona aos perfis, os quais os usuários podem então assumir. As políticas do IAM definem permissões, independentemente do método usado para realizar a operação.

Identity-based políticas

Identity-based políticas são documentos de políticas de permissões JSON que você anexa a uma identidade (usuário, grupo ou função). Essas políticas controlam quais ações as identidades podem realizar, em quais recursos e sob quais condições. Para saber como criar uma política baseada em identidade, consulte [Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente](#) no Guia do Usuário do IAM.

Identity-based as políticas podem ser políticas em linha (incorporadas diretamente em uma única identidade) ou políticas gerenciadas (políticas autônomas anexadas a várias identidades). Para saber como escolher entre uma política gerenciada e políticas em linha, consulte [Escolher entre políticas gerenciadas e políticas em linha](#) no Guia do usuário do IAM.

Resource-based políticas

Resource-based políticas são documentos de política JSON que você anexa a um recurso. Entre os exemplos estão políticas de confiança de perfil do IAM e políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. É necessário [especificar uma entidade principal](#) em uma política baseada em recursos.

Resource-based as políticas 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 às 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 que podem definir o máximo de permissões concedidas por tipos de políticas mais comuns:

- Limites de permissões: definem o número máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM. Para saber mais sobre limites de permissões, consulte [Limites de permissões para identidades do IAM](#) no Guia do usuário do IAM.
- Políticas de Controle de Serviços (SCPs): as SCPs especificam o número máximo de permissões para uma organização ou uma unidade organizacional no AWS Organizations. Para saber mais, consulte [Políticas de controle de serviço](#) no Guia do usuário do AWS Organizations .
- Políticas de controle de recursos (RCPs): definem o número máximo de permissões disponíveis para recursos em suas contas. Consulte mais informações em [Resource control policies \(RCPs\)](#) no Guia do usuário do AWS Organizations .
- Políticas de sessão: políticas avançadas transmitidas como um parâmetro durante a criação de uma sessão temporária para um perfil ou um usuário federado. Para saber mais, 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 e 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 podem ser usadas para permitir ou negar acesso em uma política. 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. Como prática recomendada, especifique um recurso usando seu [nome do recurso da Amazon \(ARN\)](#). Para ações que não oferecem compatibilidade com permissões em nível de recurso, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

No Amazon Keyspaces, keyspaces, tabelas e streams podem ser usados no `Resource` elemento de permissões do IAM.

Note

Para acessar tabelas e espaços de chave do usuário no Amazon Keyspaces, sua política do IAM deve `cassandra:Select` incluir permissões nas tabelas do sistema:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
```

Isso se aplica aos seguintes cenários:

- AWS Acesso ao console de gerenciamento
- Operações de recursos do SDK, por exemplo `GetKeyspace`, `GetTable`, `ListKeyspaces`, e `ListTables`
- Conexões padrão do driver do cliente Apache Cassandra, porque os drivers leem automaticamente as tabelas do sistema durante a inicialização da conexão

As tabelas do sistema são somente para leitura e não podem ser modificadas.

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}
```

O recurso de stream do Amazon Keyspaces tem o seguinte ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

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-1:111122223333:/keyspace/mykeyspace/"
```

Para especificar todos os espaços de chaves que pertencem a uma conta específica, use o curinga (*):

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/*"
```

Algumas ações do Amazon Keyspaces, como as de criação de recursos, não podem ser executadas em um recurso específico. Nesses casos, é necessário utilizar o caractere curinga (*).

```
"Resource": ""
```

Por exemplo, para conceder SELECT permissões a um diretor do IAM para mytable entramykeyspace, o diretor deve ter permissões para ler tanto mytable quantokeyspace/system*. Para especificar vários recursos em uma única declaração, 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` especifica quando as instruções são executadas com base em critérios definidos. 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. 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 obter 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 “Como trabalhar 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 [AssumeRole](#) ou [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 ver 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”](#).

Service-linked funções

[Service-linked as funções](#) permitem que os AWS serviços acessem recursos em outros serviços para concluir uma ação em seu nome. Service-linked as funções aparecem na 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 “Uso de 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

- [Práticas recomendadas de política](#)
- [Uso do console do Amazon Keyspaces](#)
- [Permitir que os usuários visualizem suas próprias permissões](#)
- [Como acessar as tabelas do Amazon Keyspaces](#)
- [Acesso a recursos do Amazon Keyspaces com base em tags](#)

Práticas recomendadas de política

Identity-based as políticas determinam se alguém pode criar, acessar ou excluir recursos do Amazon Keyspaces em sua conta. Essas ações podem incorrer em custos para sua 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 saber mais, 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 recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para saber mais 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: é possível adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, é possível escrever 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 CloudFormation. Para saber mais, 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 cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para saber mais, 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 saber mais, consulte [Configuração de acesso à API protegido por MFA](#) no Guia do Usuário do IAM.

Para saber mais 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 seu console. 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 “AWS políticas gerenciadas”](#).

Permitir que os 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

Note

Para acessar tabelas e espaços de chave do usuário no Amazon Keyspaces, sua política do IAM deve `cassandra:Select` incluir permissões nas tabelas do sistema:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
```

Isso se aplica aos seguintes cenários:

- AWS Acesso ao console de gerenciamento
- Operações de recursos do SDK, por exemplo `GetKeyspace`, `GetTable`, `ListKeyspaces`, e `ListTables`
- Conexões padrão do driver do cliente Apache Cassandra, porque os drivers leem automaticamente as tabelas do sistema durante a inicialização da conexão

As tabelas do sistema são somente para leitura e não podem ser modificadas.

Veja a seguir um exemplo de política que concede acesso somente leitura (SELECT) às tabelas do sistema do Amazon Keyspaces. Para todas as amostras, substitua a ID da Região e da conta no nome do recurso da Amazon (ARN) pelo seu.

```

{
  "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 espaço de chaves 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 read/write acesso a uma tabela de usuários e acesso de leitura às tabelas do sistema.

```

{
  "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*"
      ]
    }
  ]
}

```

O exemplo de política a seguir atribui acesso de leitura às tabelas do sistema, mas restringe o acesso SELECT (leitura) e MODIFY (gravação) à tabela do usuário. mytable

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}

```

```

    ]
  },
  {
    "Effect": "Deny",
    "Action": [
      "cassandra:Select",
      "cassandra:Modify"
    ],
    "Resource": [
      "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/mytable"
    ]
  }
]
}

```

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 de API da 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 Owner=richard-roe ou owner=richard-roe. Caso contrário, ele terá o acesso negado. A chave da tag de condição Owner corresponde a Owner 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 Owner 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}"
        }
    }
}
```

AWS políticas gerenciadas para Amazon Keyspaces

Uma política AWS gerenciada é uma política autônoma criada e administrada por AWS. AWS as políticas gerenciadas são projetadas para fornecer permissões para muitos casos de uso comuns, para que você possa começar a atribuir permissões a usuários, grupos e funções.

Lembre-se de que as políticas AWS gerenciadas podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque elas estão disponíveis para uso de todos os AWS clientes. Recomendamos que você reduza ainda mais as permissões definindo as [políticas gerenciadas pelo cliente](#) que são específicas para seus casos de uso.

Você não pode alterar as permissões definidas nas políticas AWS gerenciadas. Se AWS atualizar as permissões definidas em uma política AWS gerenciada, a atualização afetará todas as identidades principais (usuários, grupos e funções) às quais a política está anexada. AWS é mais provável que atualize uma política AWS gerenciada quando uma nova AWS service (Serviço da AWS) é lançada ou novas operações de API são disponibilizadas para serviços existentes.

Para saber mais, consulte [AWS Políticas gerenciadas pela](#) no Guia do usuário do IAM.

AWS política gerenciada: AmazonKeyspacesReadOnlyAccess_v2

É possível anexar a política AmazonKeyspacesReadOnlyAccess_v2 às 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 das permissões

Esta política inclui as seguintes permissões.

- **Amazon Keyspaces:** fornece acesso somente leitura ao Amazon Keyspaces.
- **Amazon Keyspaces CDC streams**— Permite que os diretores visualizem os streams CDC do 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 os diretores visualizem dados métricos e alarmes configurados em CloudWatch. Isso é necessário para que os usuários possam visualizar o tamanho da tabela faturável e CloudWatch os alarmes que foram configurados para uma tabela.
- **AWS KMS**— Permite que os diretores visualizem as chaves configuradas em AWS KMS. Isso é necessário para que os usuários possam visualizar AWS KMS as chaves 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 o JSON formato da política, consulte [AmazonKeyspacesReadOnlyAccess_v2](#).

AWS política gerenciada: AmazonKeyspacesReadOnlyAccess

É possível anexar a política AmazonKeyspacesReadOnlyAccess às suas identidades do IAM.

Esta política concede acesso somente leitura ao Amazon Keyspaces.

Detalhes das permissões

Esta política inclui as seguintes permissões.

- **Amazon Keyspaces:** fornece acesso somente leitura ao Amazon Keyspaces.
- **Amazon Keyspaces CDC streams**— Permite que os diretores visualizem os streams CDC do 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 os diretores visualizem dados métricos e alarmes configurados em CloudWatch. Isso é necessário para que os usuários possam visualizar o tamanho da tabela faturável e CloudWatch os alarmes que foram configurados para uma tabela.
- **AWS KMS**— Permite que os diretores visualizem as chaves configuradas em AWS KMS. Isso é necessário para que os usuários possam visualizar AWS KMS as chaves 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 o JSON formato da política, consulte [AmazonKeyspacesReadOnlyAccess](#).

AWS política gerenciada: AmazonKeyspacesFullAccess

É possível anexar a política AmazonKeyspacesFullAccess às suas identidades do IAM.

Esta política concede permissões administrativas que permitem aos administradores acesso irrestrito ao Amazon Keyspaces.

Detalhes das permissões

Esta política inclui as seguintes permissões.

- **Amazon Keyspaces:** permite que as entidades 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 os diretores vejam o tamanho da tabela faturável, bem como criem, visualizem e excluam CloudWatch alarmes 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. CloudWatch
- IAM: permite que o Amazon Keyspaces crie perfis vinculados a serviços com o IAM automaticamente quando os seguintes atributos são ativados:
 - Amazon Keyspaces CDC streams— Quando um administrador habilita um stream para uma tabela, o Amazon Keyspaces cria a função vinculada ao serviço [AWSServiceRoleForAmazonKeyspacesCDC](#) para publicar CloudWatch métricas em sua conta em seu nome.
 - Application Auto Scaling— Quando um administrador habilita o Application Auto Scaling para uma tabela, o Amazon Keyspaces cria a função vinculada ao serviço [AWSServiceRoleForApplicationAutoScaling_CassandraTable](#) para realizar ações automáticas de escalabilidade em seu nome.
 - Amazon Keyspaces multi-Region replication— Quando um administrador cria um novo espaço de chave multirregional ou adiciona um novo Região da AWS a um espaço de chave de região única existente, o Amazon Keyspaces cria a [AWSServiceRoleForAmazonKeyspacesReplication](#) função vinculada ao serviço para realizar a replicação de tabelas, dados e metadados para as regiões selecionadas em seu nome.
- AWS KMS: permite que as entidades principais visualizem as chaves configuradas no AWS KMS. Isso é necessário para que os usuários possam visualizar AWS KMS as chaves 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 o JSON formato da política, consulte [AmazonKeyspacesFullAccess](#).

AWS política gerenciada: KeyspacesCDCServiceRolePolicy

Não é possível anexar a `KeyspacesCDCServiceRolePolicy` às entidades do IAM. Essa política está vinculada a uma função vinculada ao serviço que permite que o Amazon Keyspaces execute ações em seu nome. Para obter mais informações, consulte [the section called “Transmissões CDC”](#).

Essa política concede as permissões necessárias à função vinculada ao serviço `AWSServiceRoleForAmazonKeyspacesCDC` para publicar dados de métricas de stream CDC do Amazon Keyspaces em seu nome. CloudWatch

Detalhes das permissões

Esta política inclui as seguintes permissões.

- CloudWatch— Permite que a função vinculada [AWSServiceRoleForAmazonKeyspacesCDC](#) ao serviço publique dados métricos dos fluxos CDC do Amazon Keyspaces na sua conta em seu nome. `"cloudwatch:namespace": "AWS/Cassandra"` CloudWatch

Para revisar o JSON formato da política, consulte [KeyspacesCDCServiceRolePolicy](#).

Atualizações do Amazon Keyspaces para AWS políticas gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do Amazon Keyspaces desde que esse serviço começou a rastrear essas mudanças. Para obter alertas automáticos sobre alterações feitas nesta página, inscreva-se no feed RSS na página [Histórico do documento](#).

Alteração	Descrição	Data
KeyspacesCDCServiceRolePolicy – Nova política	O Amazon Keyspaces adicionou uma nova política gerenciada <code>KeyspacesCDCServiceRolePolicy</code> que concede as permissões necessárias à função vinculada ao serviço	02 de julho de 2025

Alteração	Descrição	Data
	para publicar dados de métricas de stream do CDC do <code>AWSServiceRoleForAmazonKeyspacesCDC</code> Amazon Keyspaces em seu nome. CloudWatch Para obter mais informações, consulte the section called “Transmissões CDC” .	
AmazonKeyspacesReadOnlyAccess_v2 — Atualização de uma política existente	O Amazon Keyspaces adicionou novas permissões para permitir que os diretores do IAM visualizem os streams do CDC do Amazon Keyspaces. Para obter mais informações, consulte the section called “Veja os streams do CDC” .	02 de julho de 2025
AmazonKeyspacesReadOnlyAccess – atualização para uma política existente	O Amazon Keyspaces adicionou novas permissões para permitir que os diretores do IAM visualizem os streams do CDC do Amazon Keyspaces. Para obter mais informações, consulte the section called “Veja os streams do CDC” .	02 de julho de 2025

Alteração	Descrição	Data
AmazonKeyspacesFullAccess – atualização para uma política existente	O Amazon Keyspaces criou a política Keyspaces CDCServiceRolePolicy gerenciada para a função vinculada ao serviço AWSServiceRoleForAmazonKeyspacesCDC para adicionar as permissões que são necessárias quando um administrador habilita um stream para uma tabela. O Amazon Keyspaces usa a função vinculada ao serviço AWSServiceRoleForAmazonKeyspacesCDC para publicar CloudWatch métricas em sua conta em seu nome. Para obter mais informações, consulte the section called “Transmissões CDC”.	02 de julho de 2025

Alteração	Descrição	Data
AmazonKeyspacesFullAccess – atualização para uma política existente	O Amazon Keyspaces atualizou a função vinculada <code>KeyspacesReplicationServiceRolePolicy</code> do serviço AWSServiceRoleForAmazonKeyspacesReplication para adicionar as permissões que são necessárias quando um administrador adiciona uma nova Região da AWS a um espaço de chave único ou multirregional. O Amazon Keyspaces usa a função vinculada ao serviço <code>AWSServiceRoleForAmazonKeyspacesReplication</code> para replicar tabelas, suas configurações e dados em seu nome. Para obter mais informações, consulte the section called “Multi-Region Replicação”.	19 de novembro de 2024

Alteração	Descrição	Data
AmazonKeyspacesFullAccess – atualização para uma política existente	O Amazon Keyspaces adicionou novas permissões para permitir que o Amazon Keyspaces crie uma função vinculada ao serviço quando um administrador adiciona uma nova região a um keyspace único ou multirregional. 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 “Multi-Region Replicação”.	3 de outubro de 2023

Alteração	Descrição	Data
AmazonKeyspacesReadOnlyAccess_v2 — Nova política	O Amazon Keyspaces criou uma nova política para adicionar permissões somente de leitura para clientes que se conectam ao Amazon Keyspaces por meio de endpoints VPC de interface para acessar a instância do Amazon EC2 e pesquisar informações de rede. O Amazon Keyspaces armazena os endpoints da VPC de interface disponíveis na tabela <code>system.peers</code> 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. O Amazon Keyspaces usa a função vinculada ao serviço <code>AWSServiceRoleForAmazonKeyspacesReplication</code> para realizar tarefas de replicação de dados em seu nome. Para obter mais informações, consulte the section called “Multi-Region Replicação”.	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 CloudWatch. O Amazon Keyspaces se integra à Amazon CloudWatch para permitir que você monitore o tamanho da tabela faturável. Para obter mais informações, consulte the section called “Métricas 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 CloudWatch. O Amazon Keyspaces se integra à Amazon CloudWatch para permitir que você monitore o tamanho da tabela faturável. Para obter mais informações, consulte the section called “Métricas do Amazon Keyspaces”.	7 de julho de 2022
AmazonKeyspacesReadOnlyAccess – atualização para uma política existente	O Amazon Keyspaces adicionou novas permissões para permitir que os usuários visualizem AWS KMS chaves que foram configuradas para criptografia do Amazon Keyspaces em repouso. A criptografia em repouso do Amazon Keyspaces se integra à AWS KMS proteção e ao gerenciamento das chaves de criptografia usadas para criptografar dados em repouso. Para visualizar a AWS KMS chave configurada para o Amazon Keyspaces, permissões somente de 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 AWS KMS chaves que foram configuradas para criptografia do Amazon Keyspaces em repouso. A criptografia em repouso do Amazon Keyspaces se integra à AWS KMS proteção e ao gerenciamento das chaves de criptografia usadas para criptografar dados em repouso. Para visualizar a AWS KMS chave configurada para o Amazon Keyspaces, permissões somente de leitura foram adicionadas.	1º de junho de 2021
O Amazon Keyspaces passou a monitorar alterações	O Amazon Keyspaces começou a monitorar as mudanças em suas políticas AWS gerenciadas.	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 recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces](#)
- [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 Console de gerenciamento da AWS 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 mateojackson IAM tenta usar o console para visualizar detalhes sobre um *table*, mas não tem cassandra: *Select* permissões 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 recuperação para um ponto no tempo (PITR) do Amazon Keyspaces

Se você está tentando restaurar uma tabela do Amazon Keyspaces com recuperação para um ponto no tempo (PITR) e vê o processo de restauração começar, mas não ser concluído com êxito, talvez

todas as permissões necessárias para o processo de restauração não tenham sido configuradas. 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 “Configurar permissões do IAM para restauração”](#).

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, você deve conceder permissão às pessoas ou aplicações que precisam de acesso. Se você estiver usando o Centro de Identidade do AWS IAM para gerenciar pessoas e aplicações, atribua conjuntos de permissões a usuários ou grupos para definir o nível de acesso. Os conjuntos de permissões criam e atribuem automaticamente políticas do IAM aos perfis do IAM associados à pessoa ou aplicação. Para ter mais informações, consulte [Conjuntos de permissões](#) no Guia do usuário do Centro de Identidade do AWS IAM .

Se você não estiver usando o Centro de Identidade do IAM, deverá criar entidades do IAM (usuários ou perfis) para as pessoas ou aplicações que precisam de acesso. Você deve anexar uma política à entidade que concede a elas as permissões corretas no Amazon Keyspaces. Depois que as permissões forem concedidas, forneça as credenciais ao usuário ou desenvolvedor da aplicação. Eles usarão essas credenciais para acessar AWS. Para saber mais sobre como criar grupos, políticas, permissões e usuários do IAM, consulte [Identidades do IAM](#) e [Políticas e permissões no IAM](#) no Guia do usuário do IAM.

Quero permitir que pessoas de fora da minha Conta da AWS acessem meus recursos do Amazon Keyspaces

É possível criar um perfil que os usuários de outras contas ou pessoas fora da organização podem usar para acessar seus recursos. É possível 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), é possível 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 conhecer a diferença entre perfis e políticas baseadas em recurso 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 funções vinculadas a serviços AWS Identity and Access Management \(IAM\)](#). Uma função vinculada a serviços é um tipo exclusivo de função do IAM vinculada diretamente ao Amazon Keyspaces. Service-linked as funções são predefinidas pelo Amazon Keyspaces e incluem todas as permissões que o serviço exige para chamar AWS outros serviços em seu nome.

Para obter informações sobre outros serviços que oferecem suporte a funções vinculadas a [AWS serviços, consulte serviços que funcionam com IAM](#) e procure os serviços que têm Sim na coluna de Service-linked funções. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Tópicos

- [Usando funções para o ajuste de escala automático do aplicativo Amazon Keyspaces](#)
- [Usando funções para a replicação do Amazon Keyspaces Multi-Region](#)
- [Usando funções para streams de CDC do Amazon Keyspaces](#)

Usando funções para o ajuste de escala automático do aplicativo Amazon Keyspaces

[O Amazon Keyspaces \(para Apache Cassandra\) usa funções vinculadas a serviços AWS Identity and Access Management \(IAM\)](#). Uma função vinculada a serviços é um tipo exclusivo de função do IAM vinculada diretamente ao Amazon Keyspaces. Service-linked as funções são predefinidas pelo Amazon Keyspaces e incluem todas as permissões que o serviço exige para chamar AWS outros serviços 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.

Service-linked permissões de função para Amazon Keyspaces

O Amazon Keyspaces usa a função vinculada ao serviço chamada `AWSServiceRoleForApplicationAutoScaling_CassandraTable` para permitir que o Application Auto Scaling chame o Amazon Keyspaces e a Amazon em seu nome. CloudWatch

A função `CassandraTable` vinculada ao serviço `AWSServiceRoleForApplicationAutoScaling` 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 a Console de gerenciamento da AWS, CQL, a ou a AWS API AWS 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, será possível usar esse mesmo processo para recriar o perfil em sua conta. Quando você ativa o ajuste de escala automático do Amazon Keyspaces para uma tabela, o ajuste de escala automático do aplicativo cria a função vinculada ao serviço para você novamente.

 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 [Uma nova função apareceu no meu Conta da AWS](#).

Edição de um perfil vinculado ao serviço do Amazon Keyspaces

O Amazon Keyspaces não permite que você edite a função vinculada ao CassandraTable serviço AWSServiceRoleForApplicationAutoScaling. Depois de criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para saber mais, consulte [Editar um perfil vinculado ao 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. No entanto, primeiro você deve desativar o escalonamento automático em todas as tabelas da conta Regiões da AWS antes de excluir manualmente a função vinculada ao serviço. Para desativar o ajuste de escala automático nas tabelas do Amazon Keyspaces, consulte [the section called “Desativar o ajuste de escala automático do Amazon Keyspaces para uma tabela”](#).

 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 IAM AWS CLI, o ou a AWS API para excluir a função CassandraTable vinculada ao serviço AWSServiceRoleForApplicationAutoScaling. Para obter mais informações, consulte [Excluir uma Service-Linked função](#) 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](#).

Usando funções para a replicação do Amazon Keyspaces Multi-Region

[O Amazon Keyspaces \(para Apache Cassandra\) usa funções vinculadas a serviços AWS Identity and Access Management \(IAM\)](#). Uma função vinculada a serviços é um tipo exclusivo de função do IAM vinculada diretamente ao Amazon Keyspaces. Service-linked as funções são predefinidas pelo Amazon Keyspaces e incluem todas as permissões que o serviço exige para chamar AWS outros serviços 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.

Service-linked permissões de função para Amazon Keyspaces

O Amazon Keyspaces usa a função vinculada ao serviço nomeada para permitir que o `AWSServiceRoleForAmazonKeyspacesReplication` Amazon Keyspaces adicione algo novo Regiões da AWS a um keyspace em seu nome e replique tabelas e todos os seus dados e configurações para a nova região. A função também permite que o Amazon Keyspaces replique gravações em tabelas em todas as regiões 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`
- Ação: `cassandra:AlterMultiRegionResource`
- Ação: `application-autoscaling:RegisterScalableTarget` — O Amazon Keyspaces usa as permissões de auto scaling do aplicativo quando você adiciona uma réplica a uma única tabela de regiões no modo provisionado com o auto scaling ativado.
- Ação: `application-autoscaling:DeregisterScalableTarget`
- Ação: `application-autoscaling:DescribeScalableTargets`
- Ação: `application-autoscaling:PutScalingPolicy`
- Ação: `application-autoscaling:DescribeScalingPolicies`
- Ação: `cassandra:Alter`
- Ação: `cloudwatch>DeleteAlarms`
- Ação: `cloudwatch:DescribeAlarms`
- Ação: `cloudwatch:PutMetricAlarm`

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.

As permissões para criar a função vinculada ao serviço

`AWSServiceRoleForAmazonKeyspacesReplication` estão incluídas na política

`AmazonKeyspacesFullAccess` gerenciada. Para obter mais informações, consulte [the section called “AmazonKeyspacesFullAccess”](#).

Você deve configurar permissões para permitir que seus usuários, grupos ou perfis criem, editem ou excluam um perfil vinculado ao serviço. Para obter mais informações, consulte [as permissões de Service-linked função](#) 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 keyspace multirregional na Console de gerenciamento da AWS, na ou na AWS API AWS CLI, o Amazon Keyspaces cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, será possível 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 ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para saber mais, consulte [Editar um perfil vinculado ao 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. No entanto, você deve primeiro excluir todos os espaços-chave multirregionais da conta Regiões da AWS antes de excluir manualmente a função vinculada 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 Console de gerenciamento da AWS 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](#) da CLI. AWS
- A [DeleteKeyspace](#) operação da API Amazon Keyspaces.

Excluir manualmente o perfil vinculado ao serviço

Use o console do IAM AWS CLI, o ou a AWS API para excluir a função `AWSServiceRoleForAmazonKeyspacesReplication` vinculada ao serviço. Para saber mais, consulte [Excluir um perfil vinculado 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 (N. da Califórnia)	us-west-1	Sim
Oeste dos EUA (Oregon)	us-west-2	Sim

Nome da região	Identidade da região	Suporte no Amazon Keyspaces
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
Europa (Frankfurt)	eu-central-1	Sim
Europa (Irlanda)	eu-west-1	Sim
Europa (Londres)	eu-west-2	Sim
Europa (Paris)	eu-west-3	Sim
África (Cidade do Cabo)	af-south-1	Sim
América do Sul (São Paulo)	sa-east-1	Sim
AWS GovCloud (US-East)	us-gov-east-1	Não
AWS GovCloud (US-West)	us-gov-west-1	Não

Usando funções para streams de CDC do Amazon Keyspaces

[O Amazon Keyspaces \(para Apache Cassandra\) usa funções vinculadas a serviços AWS Identity and Access Management \(IAM\)](#). Uma função vinculada a serviços é um tipo exclusivo de função do IAM vinculada diretamente ao Amazon Keyspaces. Service-linked as funções são predefinidas pelo Amazon Keyspaces e incluem todas as permissões que o serviço exige para chamar AWS outros serviços 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 a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Você não pode excluir a função vinculada ao serviço.

Service-linked permissões de função para Amazon Keyspaces

O Amazon Keyspaces usa a função vinculada ao serviço nomeada para permitir que os streams CDC do `AWSServiceRoleForAmazonKeyspacesCDC` Amazon Keyspaces publiquem CloudWatch métricas em sua conta em seu nome.

A função `AWSServiceRoleForAmazonKeyspacesCDC` vinculada ao serviço confia no seguinte serviço para assumir a função:

- `cassandra-streams.amazonaws.com`

A política de permissões de função nomeada [KeyspacesCDCServiceRolePolicy](#) permite que o Amazon Keyspaces conclua a seguinte ação em recursos no CloudWatch namespace: `AWS/Cassandra`

- Ação: `cloudwatch:PutMetricData` em *

`AWSServiceRoleForAmazonKeyspacesCDC` Fornece as permissões: Ação: `cloudwatch:PutMetricData` em todos os recursos que correspondem à seguinte condição:.

```
"cloudwatch:namespace": "AWS/Cassandra"
```

Para obter mais informações sobre `KeyspacesCDCServiceRolePolicy`, consulte [the section called “KeyspacesCDCServiceRolePolicy”](#).

Para habilitar fluxos de CDC para uma tabela, que cria automaticamente a função vinculada ao serviço `AWSServiceRoleForAmazonKeyspacesCDC`, o diretor do IAM precisa das seguintes permissões.

```
{
  "Sid": "KeyspacesCDCServiceLinkedRole",
  "Effect": "Allow",
```

```
"Action": "iam:CreateServiceLinkedRole",
"Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-streams.amazonaws.com/
AWSServiceRoleForAmazonKeyspacesCDC",
"Condition": {
  "StringLike": {
    "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
  }
}
```

As permissões para criar a função vinculada ao serviço `AWSServiceRoleForAmazonKeyspacesCDC` estão incluídas na política `AmazonKeyspacesFullAccess` gerenciada. Para obter mais informações, consulte [the section called “AmazonKeyspacesFullAccess”](#).

Criação de um perfil vinculado ao serviço para o Amazon Keyspaces

Você não precisa criar manualmente uma função vinculada ao serviço para os streams CDC do Amazon Keyspaces. Quando você habilita streams de CDC do Amazon Keyspaces em uma tabela com o Console de gerenciamento da AWS, CQL, o ou a API AWS CLI, o AWS Amazon Keyspaces cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, será possível usar esse mesmo processo para recriar o perfil em sua conta. Quando você habilita streams CDC do Amazon Keyspaces para uma tabela, o Amazon Keyspaces cria a função vinculada 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 `AWSServiceRoleForAmazonKeyspacesCDC` serviço. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para saber mais, consulte [Editar um perfil vinculado ao serviço](#) no Guia do usuário do IAM.

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 [Regiões e endpoints da AWS](#).

Validação de conformidade do Amazon Keyspaces (para Apache Cassandra)

Third-party os auditores 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 9001:2015. ISO/IEC 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. Para obter mais informações sobre sua responsabilidade de conformidade ao usar Serviços da AWS, consulte a [documentação AWS de segurança](#).

Resiliência e recuperação de desastres no Amazon Keyspaces

A infraestrutura AWS global é construída em torno Regiões da AWS de zonas de disponibilidade. Regiões da AWS fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicações 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 data center tradicionais.

O Amazon Keyspaces replica dados automaticamente três vezes em várias zonas de AWS disponibilidade dentro da mesma Região da AWS para maior durabilidade e alta disponibilidade.

Para obter mais informações sobre zonas de disponibilidade Regiões da AWS e zonas de disponibilidade, consulte [infraestrutura AWS global](#).

Além da infraestrutura AWS global, o Amazon Keyspaces oferece vários recursos para ajudar a suportar suas necessidades de resiliência e backup de dados.

Replicação em várias regiões

O Amazon Keyspaces fornece replicação em várias regiões se você precisar replicar seus dados ou aplicativos em distâncias geográficas maiores. Você pode replicar suas tabelas do Amazon Keyspaces em Regiões da AWS diferentes tabelas de sua escolha. Para obter mais informações, consulte [the section called “Multi-Region replicação”](#).

Point-in-time recuperação (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 [Point-in-timerecuperação para Amazon Keyspaces](#).

Segurança da infraestrutura no Amazon Keyspaces

Como um serviço gerenciado, o Amazon Keyspaces (para Apache Cassandra) é protegido pela segurança de rede global. AWS Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte [AWS Cloud Security](#). Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte [Proteção](#) de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o Amazon Keyspaces pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Suítes de criptografia com sigilo direto perfeito (PFS), como DHE (Ephemeral) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

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 AWS CLI, o ou a AWS API. Para obter mais informações, consulte [Como usar o IAM com o Amazon Keyspaces](#).

O segundo método usa um plug-in de autenticação para o driver DataStax Java de código aberto para 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 do Signature Version 4 \(SigV4\) da AWS](#). Para obter mais informações, consulte [the section called “Crie credenciais do IAM para autenticação AWS”](#).

Você pode chamar essas operações de API de qualquer local de rede, mas o Amazon Keyspaces oferece suporte a políticas de acesso baseadas em recursos, que podem incluir restrições com base no endereço IP de origem. Você também pode usar as políticas do Amazon Keyspaces para controlar o acesso de endpoints específicos da Amazon Virtual Private Cloud (Amazon VPC) ou de VPCs específicas. Efetivamente, isso isola o acesso à rede a um determinado recurso do Amazon Keyspaces somente da VPC específica dentro da rede. 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 de VPC da Interface são alimentados por AWS PrivateLink uma AWS tecnologia que permite a comunicação privada entre AWS serviços 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 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 da VPC e recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces](#)
- [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 tutorial passo a passo para configurar uma conexão ao Amazon Keyspaces por meio de endpoints da VPC, consulte [the section called “Conexão com endpoints da VPC”](#). 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 “Configure o 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 `system.peers` tabela com os endpoints de VPC da interface disponível melhora o balanceamento de carga e aumenta a taxa de transferência. `read/write` 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: Read-only acesso — Esse exemplo de política pode ser anexado a um endpoint de 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 Amazon 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 obter mais informações, consulte [???](#).

Políticas de endpoint da VPC e recuperação para um ponto no tempo (PITR) do Amazon Keyspaces

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 endpoints 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”](#).

Usando streams CDC do Amazon Keyspaces com endpoints de interface VPC

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 de 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 Direct Connect. Para obter mais informações, consulte [Amazon Virtual Private Cloud](#) e [Endpoints da VPC de interface \(AWS PrivateLink\)](#).

Tópicos

- [Usando endpoints VPC de interface para fluxos CDC do Amazon Keyspaces](#)
- [Amazon Keyspaces \(CDC\): interface de streams de VPC, endpoints](#)
- [Crie uma interface de streams CDC do Amazon Keyspaces \(VPC endpoint\)](#)
- [Atualize um endpoint VPC da interface de streams CDC do Amazon Keyspaces](#)
- [Listar streams usando uma interface de streams CDC do Amazon Keyspaces \(VPC\)](#)
- [Crie uma política para um endpoint VPC da interface de streams CDC do Amazon Keyspaces](#)

Usando endpoints VPC de interface para fluxos CDC do Amazon Keyspaces

Você pode usar uma interface VPC endpoint para que o tráfego entre os streams de CDC do Amazon Keyspaces e seus recursos do Amazon VPC comece a fluir pela interface VPC endpoint. Você pode usar políticas de VPC endpoint para restringir o acesso aos seus streams de CDC.

Para obter mais informações sobre os streams CDC do Amazon Keyspaces, consulte [the section called “Trabalhando com fluxos do CDC”](#)

Amazon Keyspaces (CDC): interface de streams de VPC, endpoints

Quando você cria um endpoint de interface, os streams CDC do Amazon Keyspaces geram dois tipos de nome DNS específico do endpoint para o stream: regional e zonal.

Regional

O nome do DNS regional inclui as seguintes informações:

- um ID de endpoint exclusivo da Amazon VPC
- um identificador de serviço
- o Região da AWS
- o `vpce.amazonaws.com` sufixo

Para um endpoint Amazon VPC com o ID `vpce-1a2b3c4d`, o nome DNS gerado pode ser semelhante ao exemplo a seguir: `vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com`

Zonal

O nome do DNS zonal inclui a [zona de disponibilidade](#), além das informações no nome do DNS regional. O nome DNS gerado para o endpoint Amazon VPC com o `vpce-1a2b3c4d` ID seria semelhante ao exemplo a seguir. Observe que agora inclui Região da AWS a zona de disponibilidade: `vpce-1a2b3c4d-5e6f-us-east-1a.cassandra-streams.us-east-1.vpce.amazonaws.com`

Você pode usar essa opção se sua arquitetura isolar as zonas de disponibilidade. Por exemplo, você pode usar para contenção de falhas ou para reduzir os custos regionais de transferência de dados.

Note

Para obter a confiabilidade ideal, recomendamos implantar seu serviço em um mínimo de três zonas de disponibilidade.

Crie uma interface de streams CDC do Amazon Keyspaces (VPC endpoint)

Você pode usar o SDK AWS CLI ou o AWS SDK para acessar as operações da API CDC Streams do Amazon Keyspaces por meio dos endpoints de interface do Amazon Keyspaces CDC Streams. Para obter uma lista completa de todas as operações de API disponíveis, consulte a Referência de API do [Amazon Keyspaces Streams](#).

Para obter mais informações sobre como criar endpoints de VPC, consulte [criar um endpoint de interface no](#) Guia do usuário da Amazon VPC.

Para criar um VPC endpoint, você pode usar a sintaxe no exemplo a seguir.

```
aws ec2 create-vpc-endpoint \  
  --region us-east-1 \  
  --service-name api.aws.us-east-1.cassandra-streams \  
  --vpc-id client-vpc-id \  
  --subnet-ids client-subnet-id \  
  --vpc-endpoint-type Interface \  
  --security-group-ids client-sg-id
```

Atualize um endpoint VPC da interface de streams CDC do Amazon Keyspaces

Para atualizar um VPC endpoint, você pode usar a sintaxe no exemplo a seguir.

```
aws ec2 modify-vpc-endpoint \  
  --region us-east-1 \  
  --vpc-endpoint-id client-vpc-id \  
  --policy-document policy-document \ #example optional parameter  
  --add-security-group-ids security-group-ids \ #example optional parameter
```

Listar streams usando uma interface de streams CDC do Amazon Keyspaces (VPC)

Para listar os streams que estão usando um VPC endpoint, você pode usar a sintaxe no exemplo a seguir. Certifique-se de substituir a região e o nome DNS do ID do VPC endpoint por suas próprias informações.

```
aws keyspacesstreams \
  --endpoint https://vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com \
  --region us-east-1 \
  list-streams
```

Crie uma política para um endpoint VPC da interface de streams CDC do Amazon Keyspaces

Você pode anexar uma política de endpoint ao seu endpoint da Amazon VPC que controla o acesso aos streams do Amazon Keyspaces CDC. Essa política especifica as seguintes informações:

- O principal AWS Identity and Access Management (IAM) que pode realizar ações
- As ações que podem ser executadas
- Os recursos nos quais as ações podem ser executadas

Para restringir o acesso a streams específicos do Amazon Keyspaces CDC para permitir somente serviços específicos AWS em seu acesso à Amazon VPC, você pode usar o exemplo a seguir.

A política de stream a seguir concede acesso a qualquer principal do IAM para as ações `cassandra:GetStream` e `cassandra:GetRecords` para o stream especificado `2025-02-20T11:22:33.444` anexado ao recurso `/keyspace/mykeyspace/table/mytable/` pertencente à conta `123456788901`. Para usar essa política de endpoint, certifique-se de substituir a região, o ID da conta e o recurso pelo rótulo do stream.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1216114807515",
  "Statement": [
    { "Sid": "Access-to-specific-stream-only",
      "Principal": "*",
      "Action": [
        "cassandra:GetStream",
        "cassandra:GetRecords"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/mytable/stream/2025-02-20T11:22:33.444"]
    }
  ]
}
```

```
]
}
```

 Note

O Amazon Keyspaces não oferece suporte a endpoints do Gateway para fluxos de CDC.

Análise de configuração e vulnerabilidade no Amazon Keyspaces

AWS lida com tarefas básicas de segurança, como sistema operacional (SO) convidado e aplicação de patches em bancos de dados, configuração de firewall 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 para o seu ambiente, trate-as como considerações úteis em vez de prescrições.

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 um 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, aplicativos e outros AWS serviços acessem o Amazon Keyspaces, eles devem incluir AWS credenciais válidas em suas AWS solicitações de API. Você não deve armazenar AWS credenciais diretamente no aplicativo 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 [Perfis 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:

- [AWS políticas gerenciadas \(predefinidas\)](#)
- [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 Identity-Based de políticas](#).

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 AWS KMS chave AWS Key Management Service (AWS KMS)

Se você estiver usando uma [AWS KMS chave gerenciada pelo cliente](#) para criptografia em repouso, o uso dessa chave será registrado. AWS CloudTrail CloudTrail fornece visibilidade da atividade do usuário ao registrar as ações realizadas em sua conta. 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 AWS serviço. Essas informações ajudam você a monitorar as alterações feitas em seus AWS recursos e solucionar problemas operacionais. CloudTrail facilita a garantia da conformidade com políticas internas e padrões regulatórios.

Você pode usar CloudTrail para auditar o uso da chave. CloudTrail cria arquivos de log que contêm um histórico de chamadas de AWS API e eventos relacionados à sua conta. Esses

arquivos de log incluem todas as solicitações de AWS KMS API feitas usando o console, AWS os SDKs e as ferramentas de linha de comando, além das feitas por meio de AWS serviços integrados. Você pode usar esses arquivos de log para obter informações sobre quando a AWS KMS chave 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 CloudTrail para monitorar as operações da linguagem de definição de dados (DDL) do Amazon Keyspaces

CloudTrail fornece visibilidade da atividade do usuário ao registrar as ações realizadas em sua conta. 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 AWS serviço. Essas informações ajudam você a monitorar as alterações feitas em seus AWS recursos e a solucionar problemas operacionais. CloudTrail facilita a garantia da conformidade com políticas internas e padrões regulatórios.

Todas as [operações de DDL](#) do Amazon Keyspaces são registradas automaticamente. CloudTrail As operações DDL permitem criar e gerenciar tabelas e espaços de chaves do Amazon Keyspaces.

Quando a atividade ocorre no Amazon Keyspaces, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço 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 no seu Conta da AWS. Para obter mais informações, consulte [Visualização de CloudTrail eventos com histórico](#) de eventos no Guia AWS CloudTrail do usuário.

[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 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, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros.

Marque seus recursos do Amazon Keyspaces para identificação e automação

Você pode atribuir metadados aos seus AWS recursos 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: usada 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, 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).

- **Elementos do CQL:** esta seção abrange os elementos fundamentais do CQL com suporte no Amazon Keyspaces, incluindo identificadores, constantes, termos e tipos de dados. Ele explica conceitos como tipos de sequências de caracteres, tipos numéricos, tipos de coleção e muito mais.
- **Data Definition Language (DDL):** as instruções DDL são usadas para gerenciar estruturas de dados como keyspaces e tabelas no Amazon Keyspaces. Esta seção aborda instruções para criar, alterar e eliminar tabelas e espaços de chave, bem como restaurar tabelas a partir de um backup pontual.
- **Linguagem de manipulação de dados (DML):** as instruções DML são usadas para gerenciar dados em tabelas. Esta seção abrange instruções para selecionar, inserir, atualizar e excluir dados. Também explica os recursos avançados de consulta, como o uso do operador do IN, os resultados do pedido e a paginação.
- **Built-in funções** — O Amazon Keyspaces oferece suporte a uma variedade de funções escalares integradas que você pode usar em instruções CQL. Esta seção fornece uma visão geral dessas funções, incluindo exemplos de seu uso.

Ao longo deste tópico, você encontrará sintaxe detalhada, exemplos e melhores práticas para usar o CQL de forma eficaz no Amazon Keyspaces.

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](#)
- [Built-in funções 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	Description
<code>ascii</code>	Representa uma cadeia de caracteres ASCII.
<code>text</code>	Representa uma string UTF-8 codificada.
<code>varchar</code>	Representa uma string UTF-8 codificada (varchar é um alias para <code>text</code>).

Tipos numéricos

Tipo de dados	Description
<code>bigint</code>	Representa um comprimento assinado de 64 bits.
<code>counter</code>	Representa um contador de números inteiros assinado de 64 bits. Para obter mais informações, consulte the section called “Contadores” .
<code>decimal</code>	Representa um decimal de precisão variável.

Tipo de dados	Description
<code>double</code>	Representa um ponto flutuante IEEE 754 de 64 bits.
<code>float</code>	Representa um ponto flutuante IEEE 754 de 32 bits.
<code>int</code>	Representa um int assinado de 32 bits.
<code>varint</code>	Representa um valor inteiro dentro do intervalo +/- 10 ³⁸ .

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	Description
<code>blob</code>	Representa bytes arbitrários.

Tipo booliano

Tipo de dados	Description
boolean	Representa true ou false.

Time-related tipos

Tipo de dados	Description
date	Uma string no formato <yyyy> - <mm> - <dd> .
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	Description
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. Client-side os timestamps e as configurações de Time to Live (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.

Client-side os timestamps e as configurações de Time to Live (TTL) se aplicam a toda a coleção congelada, não a elementos individuais. Frozencolunas de coleção podem fazer parte `PRIMARY KEY` de uma tabela.

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>>>
```

O Amazon Keyspaces suporta o agrupamento de até 8 níveis de coleções congeladas por padrão. Para obter 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	Description
inet	Uma cadeia de caracteres representando um endereço IP, no formato IPv4 ou IPv6.

Estático

Em uma tabela do Amazon Keyspaces com colunas de clustering, você pode usar a palavra-chave `STATIC` 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 trabalho com colunas estáticas, consulte [the section called “Estimar o consumo de capacidade de colunas estáticas”](#).

User-defined tipos (UDTs)

O Amazon Keyspaces oferece suporte a tipos definidos pelo usuário (UDTs). Você pode usar qualquer tipo de dados válido do Amazon Keyspaces para criar um UDT, incluindo coleções e outros UDTs existentes. Você cria UDTs em um espaço de chave e pode usá-las para definir colunas em qualquer tabela no espaço de chaves.

Para obter mais informações sobre sintaxe de CQL, consulte [the section called “Tipos”](#). Para obter mais informações sobre como trabalhar com UDTs, consulte [the section called “User-defined tipos \(UDTs\)”](#).

Para analisar quantos UDTs são suportados por espaço de chave, níveis de aninhamento suportados e outros valores e cotas padrão relacionados aos UDTs, consulte [the section called “Cotas e valores padrão para tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces”](#).

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
<code>ascii</code>	<code>string</code>	<code>string</code>	Usa <code>\u</code> escape de caracteres JSON.
<code>bigint</code>	<code>integer</code> , <code>string</code>	<code>integer</code>	A <code>string</code> deve ser um número inteiro válido de 64 bits.
<code>blob</code>	<code>string</code>	<code>string</code>	A <code>string</code> deve começar com <code>0x</code> seguida por um número par de dígitos hexadecimais.
<code>boolean</code>	<code>boolean</code> , <code>string</code>	<code>boolean</code>	<code>String</code> deve ser <code>true</code> ou <code>false</code> .
<code>date</code>	<code>string</code>	<code>string</code>	Data em formato <code>YYYY-MM-DD</code> , fuso horário UTC.
<code>decimal</code>	<code>integer</code> , <code>float</code> , <code>string</code>	<code>float</code>	Pode exceder a precisão de ponto IEEE-754 flutuante de 32 ou 64 bits no decodificador do lado do cliente.

Tipo de dados do JSON	Tipos de dados aceitos em declarações 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 representação 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 declarações INSERT JSON	Tipos de dados retornados em declarações SELECT JSON	Observações
<code>time</code>	<code>string</code>	<code>string</code>	Hora do dia em formato HH-MM-SS[.ffffffffff] .
<code>timestamp</code>	<code>integer, string</code>	<code>string</code>	Um carimbo de data/hora. As constantes de <code>string</code> permitem que você armazene carimbos de data/hora como datas. Os carimbos de data com formato YYYY-MM-DD HH:MM:SS.SSS são retornados.
<code>timeuuid</code>	<code>string</code>	<code>string</code>	Tipo 1 UUID. Consulte constants para o formato UUID.
<code>tinyint</code>	<code>integer, string</code>	<code>integer</code>	A <code>string</code> deve ser um número inteiro válido de 8 bits.
<code>tuple</code>	<code>list, string</code>	<code>list</code>	Usa a representação nativa da lista JSON.
<code>UDT</code>	<code>map, string</code>	<code>map</code>	Usa a representação do mapa JSON nativo com nomes de campo como chaves.
<code>uuid</code>	<code>string</code>	<code>string</code>	Consulte constants para o formato UUID.

Tipo de dados do JSON	Tipos de dados aceitos em declarações INSERT JSON	Tipos de dados retornados em declarações SELECT JSON	Observações
<code>varchar</code>	<code>string</code>	<code>string</code>	Usa <code>\u</code> escape de caracteres JSON.
<code>varint</code>	<code>integer</code> , <code>string</code>	<code>integer</code>	Comprimento variável; pode ultrapassar números inteiros de 32 bits ou 64 bits no decodificador 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 DDL são compatíveis:

- [CRIAR ESPAÇO DE CHAVES](#)
- [ALTERAR ESPAÇO DE CHAVES](#)
- [DESCARTAR ESPAÇO DE CHAVES](#)
- [USO](#)
- [CRIAR TABELA](#)
- [ALTER TABLE](#)

- [RESTAURAR TABELA](#)
- [DESCARTAR TABELA](#)
- [TIPO DE CRIAÇÃO](#)
- [TIPO DE GOTA](#)

Tópicos

- [Keyspaces](#)
- [Tabelas](#)
- [User-defined tipos \(UDTs\)](#)

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](#).

Instruções para espaços de chaves

- [CRIAR ESPAÇO DE CHAVES](#)
- [ALTERAR ESPAÇO DE CHAVES](#)
- [DESCARTAR ESPAÇO DE CHAVES](#)
- [USE](#)

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 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. 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

Você pode usar a `ALTER KEYSPACE WITH` declaração para as seguintes opções

- **REPLICATION**— Use essa opção para adicionar uma nova Região da AWS réplica a um keyspace. Você pode adicionar uma nova região a uma única região ou a um keyspace multirregional.
- **TAGS**— Use essa opção para adicionar ou remover tags de um keyspace.

Sintaxe

```
alter_keyspace_statement ::=  
  ALTER KEYSPACE keyspace_name  
  WITH options
```

Em que:

- *keyspace_name* é o nome do espaço de chaves a ser alterado.
- as opções são uma das seguintes:
 - **ADD | DROP TAGS**: uma lista de tags de pares chave-valor a serem adicionadas ou removidas do espaço de chaves.
 - **REPLICATION**— Um mapa que indica a estratégia de replicação para o keyspace;
 - **class**— `NetworkTopologyStrategy` define o keyspace como um keyspace multirregional.
 - **region**— Especifique um adicional Região da AWS para esse espaço de teclas. O fator de replicação para cada região é três.
 - **CLIENT_SIDE_TIMESTAMPS**— O padrão é `DISABLED`. Você só pode alterar o status para `ENABLED`.

Exemplos

Altere um espaço de teclas conforme mostrado no exemplo a seguir para adicionar tags.

```
ALTER KEYSPACE my_keyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

Para adicionar uma terceira região a um keyspace multirregional, você pode usar a seguinte declaração.

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
  'class': 'NetworkTopologyStrategy',
  'us-east-1': '3',
  'us-west-2': '3',
  'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

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 my_keyspace;
```

USE

Use a `USE` instrução para definir o espaço de teclas atual. Isso permite que você faça referência a objetos vinculados a um espaço de chave específico, por exemplo, tabelas e tipos, sem usar o nome totalmente qualificado que inclui o prefixo do espaço de teclas.

Sintaxe

```
use_statement ::=
USE keyspace_name
```

Em que:

- `keyspace_name` é o nome do keyspace a ser usado.

Exemplo

```
USE my_keyspace;
```

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 read/write de taxa de transferência de capacidade, consulte [the section called “Configurar modos read/write de capacidade”](#).

Para tabelas no modo provisionado, você pode configurar AUTOSCALING_SETTINGS opcionais. Para obter mais informações sobre o ajuste de escala automático do Amazon Keyspaces e as opções disponíveis, consulte [the section called “Configurar o ajuste de escala automático em uma tabela existente”](#).

Para informações sobre limites e restrições de cotas para tabelas do Amazon Keyspaces, consulte [Cotas](#).

Instruções 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 ]
                           | cdc
                           | 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. O nome totalmente qualificado inclui o prefixo keyspace. Como alternativa, você pode definir o espaço de teclas atual com a instrução USE keyspace.
- *column_definition* consiste no seguinte:
 - *column_name*: o nome da coluna.
 - *cql_type*: um tipo de dados do Amazon Keyspaces (consulte [Tipos de dados](#)).
 - *FROZEN*— Designa essa coluna definida pelo usuário ou do tipo collection (por exemplo, LISTSET, ou MAP) como congelada. Uma coleção congelada é serializada em um único valor imutável e tratada como a BLOB. Para obter 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.
 - *cdc*— Um booleano que especifica se o Amazon Keyspaces cria um fluxo de captura de dados de alteração (CDC) para a tabela. O padrão é *false*. Para especificar o *view_type* ao ativar um fluxo, defina *cdc_specification* com *CUSTOM_PROPERTIES*.
 - *CUSTOM_PROPERTIES*: um mapa de configurações específicas do Amazon Keyspaces.
 - *capacity_mode*: especifica o modo de capacidade de taxa de read/write transferência 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*.
 - *cdc_specification*: especifica o fluxo *view_type* do CDC. As opções são:
 - *NEW_AND_OLD_IMAGES*— ambas as versões da linha, antes e depois da alteração. Esse é o padrão.
 - *NEW_IMAGE*— a versão da linha após a alteração.
 - *OLD_IMAGE*— a versão da linha antes da alteração.
 - *KEYS_ONLY*— as chaves de partição e agrupamento da linha que foi alterada.

Para obter mais informações sobre fluxos CDC, consulte [the section called “Trabalhando com fluxos do CDC”](#). Para obter exemplos de código, consulte [the section called “Ativar um fluxo CDC para uma nova tabela”](#).

- `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 recuperação em um ponto no tempo está habilitada ou desabilitada 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. Para isso, você pode fazer isso configurando os parâmetros a seguir. Para ter mais informações e exemplos, consulte [the section called “Criar uma tabela multirregional no modo provisionado”](#).
- `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 dimensionamento automático”](#).
- `provisioned_write_capacity_autoscaling_update`:
 - `autoscaling_disabled`: para ativar o ajuste de escala automático para capacidade de gravação, defina o valor como `false`. O padrão é `true`. (Optional)

- `minimum_units`: o nível mínimo de throughput de gravação para o qual a tabela deve estar sempre pronta para dar suporte. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).
- `maximum_units`: o nível máximo de throughput de gravação para o qual a tabela deve estar sempre pronta para dar suporte. O valor deve estar entre 1 e a cota máxima de throughput 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 ajuste de escala automático 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 rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) no Guia do usuário do Application Auto Scaling.
- `target_value`: a taxa de utilização alvo da tabela. O ajuste de escala automático do Amazon Keyspaces garante que o índice de capacidade consumida para capacidade provisionada permaneça nesse valor ou próximo a ele. Você define `target_value` como uma porcentagem. Um duplo entre 20 e 90. (Obrigatória)
- `scale_in_cooldown`: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de reduzir a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)
- `scale_out_cooldown`: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de aumentar a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)
- `disable_scale_in`: um boolean que especifica se `scale-in` está desativado ou ativado para a tabela. Por padrão, esse parâmetro é desabilitado. Para ativar `scale-in`, defina o valor de boolean 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 ajuste de escala automático da capacidade de leitura, defina o valor como `false`. O padrão é `true`. (Optional)

- `minimum_units`: o 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).
- `maximum_units`: o nível máximo de throughput para o qual a tabela deve estar sempre pronta para dar suporte. O valor deve estar entre 1 e a cota máxima de throughput 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 ajuste de escala automático 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 rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) no Guia do usuário do Application Auto Scaling.
- `target_value`: a taxa de utilização alvo da tabela. O ajuste de escala automático do Amazon Keyspaces garante que o índice de capacidade consumida para capacidade provisionada permaneça nesse valor ou próximo a ele. Você define `target_value` como uma porcentagem. Um duplo entre 20 e 90. (Obrigatória)
- `scale_in_cooldown`: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de reduzir a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)
- `scale_out_cooldown`: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de aumentar a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)
- `disable_scale_in`: um boolean que especifica se `scale-in` está desativado ou ativado para a tabela. Por padrão, esse parâmetro é desabilitado. Para ativar `scale-in`, defina o valor de boolean 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. Para isso, você pode fazer isso configurando os parâmetros a seguir. Para ter mais informações e exemplos, consulte [the section called “Atualize as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#).

- **region**— A réplica Região da AWS da tabela com as seguintes configurações:
- **provisioned_read_capacity_autoscaling_update**
 - **autoscaling_disabled**: para ativar o ajuste de escala automático para a capacidade de leitura da tabela, defina o valor como `false`. O padrão é `true`. (Optional)

 Note

O ajuste de escala automático de uma tabela multirregional precisa estar ativado ou desativado para todas as réplicas da tabela.

- **minimum_units**: o nível mínimo de throughput de leitura para o qual a tabela deve estar sempre pronta para dar suporte. O valor deve estar entre 1 e a cota máxima de throughput por segundo da sua conta (40.000 por padrão).
- **maximum_units**: o nível máximo de throughput de leitura para o qual a tabela deve estar sempre pronta para dar suporte. O valor deve estar entre 1 e a cota máxima de throughput 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 ajuste de escala automático 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 rastreamento de destino e período de espera, consulte [Target Tracking Scaling Policies](#) no Guia do usuário do Application Auto Scaling.
- **target_value**: a taxa de utilização alvo da tabela. O ajuste de escala automático do Amazon Keyspaces garante que o índice de capacidade de leitura consumida para capacidade provisionada de leitura permaneça nesse valor ou próximo a ele. Você define **target_value** como uma porcentagem. Um duplo entre 20 e 90. (Obrigatória)
- **scale_in_cooldown**: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de reduzir a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)
- **scale_out_cooldown**: um período de espera em segundos entre as atividades de ajuste de escala permite que a tabela se estabilize antes que outra atividade de

aumentar a escala horizontalmente comece. Se nenhum valor for fornecido, o valor padrão será 0. (Optional)

- `disable_scale_in`: um boolean que especifica se `scale-in` está desativado ou ativado para a tabela. Por padrão, esse parâmetro é desabilitado. Para ativar `scale-in`, defina o valor de boolean 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-1:555555555555: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 “Estimar o consumo de capacidade de colunas estáticas”](#).

Exemplo

```
CREATE TABLE my_keyspace.my_table (
    id int,
    name text,
    region text,
    division text,
    project text STATIC,
    PRIMARY KEY (id,division));
```

Você pode criar uma tabela com uma coluna que usa um tipo definido pelo usuário (UDT). A primeira instrução nos exemplos cria um tipo, a segunda instrução cria uma tabela com uma coluna que usa o tipo.

Exemplo

```
CREATE TYPE my_keyspace."udt""N@ME" (my_field int);
CREATE TABLE my_keyspace.my_table (my_col1 int pri key, my_col2 "udt""N@ME");
```

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:
 - *AUTOSCALING_SETTINGS* inclui as configurações opcionais de ajuste de escala automático para tabelas provisionadas. Para obter descrições detalhadas e de sintaxe, consulte [the section called “CRIAR TABELA”](#). Para obter exemplos, consulte [the section called “Configurar o ajuste de escala automático em uma tabela existente”](#).
 - *cdc*— Um booleano que especifica se o Amazon Keyspaces cria um fluxo de captura de dados de alteração (CDC) para a tabela. O padrão é `false`. Para especificar o `view_type` ao ativar um fluxo, defina *cdc_specification* com `CUSTOM_PROPERTIES`.
 - *CUSTOM_PROPERTIES*: um mapa de configurações específicas para o Amazon Keyspaces.
 - *capacity_mode*: especifica o modo de capacidade de taxa de read/write transferência 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`.
 - *cdc_specification*: especifica o fluxo *view_type* do CDC. As opções são:
 - *NEW_AND_OLD_IMAGES*— ambas as versões da linha, antes e depois da alteração. Esse é o padrão.
 - *NEW_IMAGE*— a versão da linha após a alteração.
 - *OLD_IMAGE*— a versão da linha antes da alteração.
 - *KEYS_ONLY*— as chaves de partição e agrupamento da linha que foi alterada.

Para obter mais informações sobre fluxos CDC, consulte [the section called “Trabalhando com fluxos do CDC”](#). Para obter exemplos de código, consulte [the section called “Habilitar um fluxo CDC para uma tabela existente”](#).

- `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 recuperação em um ponto no tempo está habilitada ou desabilitada 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. Para isso, você pode fazer isso configurando os parâmetros a seguir. Para ter mais informações e exemplos, consulte [the section called “Atualize as configurações de capacidade provisionada e ajuste de escala automático para uma tabela multirregional”](#).
 - `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.
- `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>>>>);
```

O exemplo a seguir mostra como adicionar uma coluna que usa um tipo definido pelo usuário (UDT) a uma tabela.

```
ALTER TABLE my_keyspace.my_table ADD (my_column, my_udt);
```

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-1:555555555555:key/11111111-1111-111-1111-111111111111'
}
};
```

Para habilitar a restauração de ponto no tempo para 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 instrução exige que a recuperação de ponto no tempo seja habilitada em uma tabela. Para obter mais informações, consulte [the section called “Backup e restauração com recuperação para um ponto no tempo”](#).

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 taxa de read/write transferência 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 recuperação em um ponto no tempo está habilitada ou desabilitada 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. Para isso, você pode fazer isso configurando os parâmetros a seguir.
 - `region`— A réplica Região da AWS da tabela com as seguintes configurações:
 - `read_capacity_units`
 - `AUTOSCALING_SETTINGS` inclui as configurações opcionais de ajuste de escala automático para tabelas provisionadas. Para obter descrições detalhadas e de sintaxe, consulte [the 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 my_keyspace.my_table;
```

User-defined tipos (UDTs)

UDT — Um agrupamento de campos e tipos de dados que você pode usar para definir uma única coluna no Amazon Keyspaces. Os tipos de dados válidos para UDTs são todos os tipos de dados do Cassandra compatíveis, incluindo coleções e outros UDTs que você já criou no mesmo keyspace. Para obter mais informações sobre os tipos de dados do Cassandra compatíveis, consulte [the section called “Suporte ao tipo de dados do Cassandra”](#)

```

user_defined_type ::= udt_name
udt_name ::= [ keyspace_name '.' ] identifier

```

Declarações para tipos

- [CREATE TYPE](#)
- [DROP TYPE](#)

CREATE TYPE

Use a `CREATE TYPE` instrução para criar um novo tipo.

Sintaxe

```

create_type_statement ::= CREATE TYPE [ IF NOT EXISTS ] udt_name
                        '(' field_definition ( ',' field_definition ) * ')'
                        field_definition ::= identifier cql_type

```

Em que:

- `IF NOT EXISTS` `CREATE TYPE` evita que falhe se o tipo já existir. (Optional)
- `udt_name` é o nome totalmente qualificado do UDT em formato de tipo, por exemplo. `my_keyspace.my_type` Se você definir o espaço de teclas atual com a `USE` instrução, não precisará especificar o nome do espaço de teclas.
- `field_definition` consiste em um nome e um tipo.

A tabela a seguir mostra exemplos de nomes de UDT permitidos. A primeira coluna mostra como inserir o nome ao criar o tipo, a segunda coluna mostra como o Amazon Keyspaces formata o nome internamente. O Amazon Keyspaces espera o nome formatado para operações como `GetType`

Nome inserido	Nome formatado	Observação
MY_UDT	my_udt	Sem aspas duplas, o Amazon Keyspaces converte todos os caracteres maiúsculos em minúsculas.
"MY_UDT"	MY_UDT	Com aspas duplas, o Amazon Keyspaces respeita os caracteres maiúsculos e remove as aspas duplas do nome formatado.

Nome inserido	Nome formatado	Observação
"1234"	1234	Com aspas duplas, o nome pode começar com um número, e o Amazon Keyspaces remove as aspas duplas do nome formatado.
"Special_Ch@r@cter s<>!!"	Special_Ch@r@cters<>!!	Com aspas duplas, o nome pode conter caracteres especiais, e o Amazon Keyspaces remove as aspas duplas do nome formatado.
"nested""""quote s"	nested""""quotes	O Amazon Keyspaces remove as aspas duplas externas e as aspas duplas de escape do nome formatado.

Exemplos

```
CREATE TYPE my_keyspace.phone (
    country_code int,
    number text
);
```

Você pode aninhar UDTs se o UDT aninhado estiver congelado. Para obter mais informações sobre valores padrão e cotas para tipos, consulte [the section called “Cotas de UDT e valores padrão do Amazon Keyspaces”](#).

```
CREATE TYPE my_keyspace.user (
    first_name text,
    last_name text,
    phones FROZEN<phone>
);
```

Para obter mais exemplos de código que mostram como criar UDTs, consulte [the section called “User-defined tipos \(UDTs\)”](#).

DROP TYPE

Use a `DROP TYPE` instrução para excluir um UDT. Você só pode excluir um tipo que não esteja sendo usado por outro tipo ou tabela.

Sintaxe

```
drop_type_statement ::= DROP TYPE [ IF EXISTS ] udt_name
```

Em que:

- `IF EXISTS` `DROP TYPE` evita que falhe se o tipo não existir. (Optional)
- *udt_name* é o nome totalmente qualificado do UDT em formato de tipo, por exemplo.
`my_keyspace.my_type` Se você definir o espaço de teclas atual com a `USE` instrução, não precisará especificar o nome do espaço de teclas.

Exemplo

```
DROP TYPE udt_name;
```

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 JSON-encoded os tipos de dados 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 “Usar o IN SELECT”](#).

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 agrupamento nas cláusulas de ordenação. Non-clustering colunas 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 “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 obter mais informações, consulte [the section called “Client-side carimbos de data/hora”](#).

```
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 obter 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 JSON-encoded os tipos de dados 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",
"division": "Engineering",
"role": "IC",
"pay_scale": 3,
"vacation_hrs": 12.5,
"manager_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 obter 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.

Built-in funções 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	Description
<code>blobAsType</code>	Retorna um valor do tipo de dados especificado.

Função	Description
<code>cast</code>	Converte um tipo de dados nativo em outro tipo de dados nativo.
<code>currentDate</code>	Retorna o atual date/time como uma data.
<code>currentTime</code>	Retorna a corrente date/time como uma hora.
<code>currentTimestamp</code>	Retorna a corrente date/time como um carimbo de data/hora.
<code>currentTimeUUID</code>	Retorna a corrente date/time como timeuuid a.
<code>fromJson</code>	Converte a string JSON no tipo de dados da coluna selecionada.
<code>maxTimeuuid</code>	Retorna o maior timeuuid possível para carimbo de data/hora ou string de data.
<code>minTimeuuid</code>	Retorna o menor timeuuid possível para carimbo de data/hora ou string de data.
<code>now</code>	Retorna um novo timeuuid exclusivo.
<code>toDate</code>	Converte um timeuuid ou carimbo de data/hora em um tipo de data.
<code>toJson</code>	Retorna o valor da coluna selecionada no formato JSON.
<code>token</code>	Retorna o valor de hash da chave de partição.
<code>toTimestamp</code>	Converte um timeuuid ou uma data em um carimbo de data/hora.
<code>TTL</code>	Retorna o tempo de expiração em segundos para uma coluna.

Função	Description
<code>typeAsBlob</code>	Converte o tipo de dados especificado em um blob.
<code>toUnixTimestamp</code>	Converte um <code>timeuuid</code> ou carimbo de data/hora em um <code>bigInt</code> .
<code>uuid</code>	Retorna um UUID randomizado da versão 4.
<code>writetime</code>	Retorna o carimbo de data/hora do valor da coluna especificada.
<code>dateOf</code>	(Obsoleto) Extrai o carimbo de data/hora de um <code>timeuuid</code> e retorna o valor como uma data.
<code>unixTimestampOf</code>	(Obsoleto) Extrai o carimbo de data/hora de um <code>timeuuid</code> 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](#)
- [Cotas e valores padrão para tipos definidos pelo usuário \(UDTs\) no 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.

Quota	Description	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. É possível 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. É possível ajustar esse valor padrão no console Service Quotas .	256

Quota	Description	Padrão do Amazon Keyspaces
Quantidade máxima de dados restaurados usando ADD REGION operações	O tamanho máximo dos dados que podem ser restaurados simultaneamente pelas ADD REGION operações. Para aumentar a quantidade de dados a serem restaurados simultaneamente, entre em contato com Suporte.	10 TB
Tamanho máx. do esquema de tabela	O tamanho máximo de um esquema de tabela.	350 KB
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 “Estimar o tamanho da linha” .	1 MB

Quota	Description	Padrão do Amazon Keyspaces
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
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 “Calcular o tamanho da coluna estática 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. É possível ajustar esse valor padrão no console Service Quotas .	100

Quota	Description	Padrão do Amazon Keyspaces
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 Suporte.	8
Número máximo de declarações em um lote registrado	O número máximo de instruções CQL suportadas quando você usa uma instrução em lote registrada. Para obter mais informações sobre declarações de lote, consulte the section called “Use declarações em lote” .	100
Número máximo de declarações em um lote registrado usando colunas estáticas	O número máximo de instruções CQL suportadas quando você usa colunas estáticas em uma declaração em lote registrada. Para obter mais informações sobre declarações de lote, consulte the section called “Use declarações em lote” .	50

Quota	Description	Padrão do Amazon Keyspaces
Tamanho máximo da declaração de lote registrada	O tamanho máximo de uma declaração de lote registrada. Para obter mais informações sobre declarações de lote, consulte the section called “Use declarações em lote” .	4 MB
Número máximo de declarações em um lote não registrado	O número máximo de instruções CQL suportadas quando você usa uma instrução em lote não registrada. Para obter mais informações sobre declarações de lote, consulte the section called “Use declarações em lote” .	30
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. É possível ajustar esse valor padrão no console Service Quotas .	40.000

Quota	Description	Padrão do Amazon Keyspaces
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. É possível ajustar esse valor padrão no console Service Quotas .	40.000
Taxa de transferência máxima de gravação para uma tabela usando change-data-capture (CDC)	A capacidade máxima de gravação para uma tabela do Amazon Keyspaces com o CDC ativado. Essa cota padrão só se aplica às tabelas no modo de capacidade provisionada. read/write É possível ajustar esse valor padrão no console Service Quotas .	40.000
Account-level taxa de transferência de leitura (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 read/write capacidade e provisionada. É possível ajustar esse valor padrão no console Service Quotas .	80.000

Quota	Description	Padrão do Amazon Keyspaces
Account-level taxa de transferência de gravação (provisionada)	O número máximo de unidades de capacidade de escrita agregada (WCUs) alocadas para a conta por Região. Isso é aplicável somente para tabelas no modo de read/write capacidade e provisionada. É possível 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 o ajuste de escala automático estiver habilitado para capacidade de leitura e como outra meta escalável se o ajuste de escala automático estiver habilitado para capacidade de gravação. É possível ajustar esse valor padrão no console Service Quotas para o Application Auto Scaling escolhendo metas escaláveis para o Amazon Keyspaces.	1.500

Quota	Description	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 Point-in-time Recuperação (PITR)	O número máximo de restaurações simultâneas de tabelas usando PITR por assinante é 4. É possível ajustar esse valor padrão no console Service Quotas .	4
Quantidade máx. de dados restaurados usando a recuperação para um ponto no tempo (PITR)	O tamanho máximo dos dados que podem ser restaurados usando PITR em 24 horas. É possível 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` sempre que necessário, usando o console ou a instrução `ALTER TABLE`. As novas configurações não têm efeito até que a operação `ALTER TABLE` seja concluída.

Você não pode exceder as cotas por conta ao adicionar a capacidade provisionada. E você pode aumentar a capacidade provisionada de suas tabelas quantas vezes 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 AWS entre 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 da criação da tabela. 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 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 de cota de serviço, consulte [Suporte](#).

Cotas e valores padrão para tipos definidos pelo usuário (UDTs) no Amazon Keyspaces

Cotas de UDT e valores padrão do Amazon Keyspaces

A tabela a seguir contém cotas e valores padrão relacionados aos UDTs no Amazon Keyspaces. 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 essas cotas, entre em contato com AWS Support.

Quota	Description	Padrão do Amazon Keyspaces
Número máximo de UDTs por Região da AWS	O número máximo de UDTs em todos os espaços chave desse assinante por região.	256
Número máximo de tabelas por UDT	O número máximo de tabelas que podem fazer referência ao mesmo UDT.	100
Número máximo de UDTs por tabela	O número máximo de UDTs que uma tabela pode referenciar.	50
Nível máximo de aninhamento de UDTs	A profundidade máxima de aninhamento suportada por UDTs.	8
Quantidade máxima de UDTs diretos para crianças por UDT	O número máximo de UDTs secundários suportados por um UDT.	10
Quantidade máxima de UDTs de pais diretos por UDT	O número máximo de UDTs principais suportados por um UDT.	10

Quota	Description	Padrão do Amazon Keyspaces
Tamanho máximo do esquema UDT	O tamanho máximo do esquema para um UDT.	25 KB
Tamanho máximo do nome UDT	O número máximo de caracteres no nome do UDT.	48
Tamanho máximo do nome do campo UDT	O número máximo de caracteres em um nome de campo UDT.	128

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, é possível inscrever-se em um feed RSS.

- Última atualização da documentação: 28 de janeiro de 2026

Alteração	Descrição	Data
Agora você pode pré-aquecer tabelas novas e existentes no Amazon Keyspaces	Agora você pode configurar tabelas novas e existentes no Amazon Keyspaces warm-throughput para pré-aquecê-las de acordo com os picos previstos na capacidade máxima.	28 de janeiro de 2026
O Amazon Keyspaces agora oferece suporte a uma experiência integrada AWS CloudShell	Agora você pode iniciar um shell pré-autenticado diretamente do console do Amazon Keyspaces.	26 de janeiro de 2026
O Amazon Keyspaces agora oferece suporte a lotes registrados	Agora você pode combinar várias declarações em lotes registrados no Amazon Keyspaces.	11 de novembro de 2025
Alterações nos certificados TLS do Amazon Keyspaces	Instruções atualizadas para incluir certificados emitidos pela cadeia Amazon Trust Services (ATS).	25 de outubro de 2025

[O Amazon Keyspaces agora oferece suporte a solicitações IPv4 e IPv6](#)

Os endpoints de pilha dupla do Amazon Keyspaces oferecem suporte a configurações IPv4 e IPv6.

30 de setembro de 2025

[NOWe agora UUID são compatíveis com todas as instruções CQL no Amazon Keyspaces](#)

Agora você pode usar as UUID funções NOW e em todas as instruções CQL no Amazon Keyspaces.

8 de setembro de 2025

[O adaptador Amazon Keyspaces Streams Kinesis já está disponível](#)

Agora você pode processar streams CDC do Amazon Keyspaces com a Kinesis Client Library (KCL).

02 de julho de 2025

[Nova política gerenciada do Amazon Keyspaces](#)

O Amazon Keyspaces adicionou uma nova política gerenciada Keyspaces `CDCServiceRolePolicy` que concede as permissões necessárias à função vinculada ao serviço para publicar dados de métricas de stream do CDC do `AWSServiceRoleForAmazonKeyspacesCDC` Amazon Keyspaces em seu nome. CloudWatch Para obter mais informações, consulte [Uso de funções para streams CDC do Amazon Keyspaces](#).

02 de julho de 2025

[Atualização da política gerenciada pelo Amazon Keyspaces](#)

O Amazon Keyspaces adicionou novas permissões às políticas `AmazonKeyspacesReadOnlyAccess` gerenciadas pelo `AmazonKeyspacesReadOnlyAccess_v2` e às políticas para permitir que os diretores do IAM visualizassem streams. O Amazon Keyspaces atualizou a política `AmazonKeyspacesFullAccess` gerenciada com permissões para criar uma função vinculada ao serviço `AWSServiceRoleForAmazonKeyspacesCDC` que publica CloudWatch métricas em nome do diretor.

02 de julho de 2025

[Os streams de captura de dados de alteração \(CDC\) do Amazon Keyspaces agora estão disponíveis](#)

Permita que os streams do CDC registrem alterações em suas tabelas do Amazon Keyspaces.

02 de julho de 2025

[A replicação multirregional do Amazon Keyspaces agora oferece suporte à região da África \(Cidade do Cabo\).](#)

Agora você pode definir a região da África (Cidade do Cabo) para espaços chave multirregionais.

31 de março de 2025

[A replicação multirregional do Amazon Keyspaces agora oferece suporte ilimitado. Regiões da AWS](#)

Agora você pode definir quantas regiões quiser para espaços de chave multirregionais.

25 de março de 2025

[Suporte do Amazon Keyspaces para tipos definidos pelo usuário \(UDTs\) em espaços chave multirregionais](#)

Agora você pode criar e usar UDTs em espaços chave de várias regiões.

24 de março de 2025

[Atualização da política gerenciada pelo Amazon Keyspaces](#)

O Amazon Keyspaces adicionou novas permissões à política AmazonKey spacesFullAccess gerenciada para permitir que os diretores do IAM adicionassem novas regiões a um keyspace existente. Isso inclui uma atualização na função vinculada ao serviço. [AWSServiceRoleForAmazonKeyspacesReplication](#)

19 de novembro de 2024

[Adicione réplicas a tabelas multirregionais no Amazon Keyspaces](#)

Agora você pode adicionar novos espaços Regiões da AWS de chave únicos e multirregionais existentes.

19 de novembro de 2024

[Support para tipos definidos pelo usuário \(UDTs\) no Amazon Keyspaces](#)

Com o suporte nativo para UDTs no Amazon Keyspaces , agora você pode definir estruturas de dados em seus aplicativos que representem hierarquias de dados do mundo real.

30 de outubro de 2024

[ADD COLUMNsuporte para replicação multirregional do Amazon Keyspaces](#)

O Amazon Keyspaces agora oferece suporte a alterações de esquema para tabelas multirregionais.

17 de setembro de 2024

Orientações de migração atualizadas para Amazon Keyspaces	A orientação de migração atualizada descreve como criar um plano de migração para migrar com sucesso do Apache Cassandra para o Amazon Keyspaces, incluindo estratégias diferentes para migrações offline e online.	28 de junho de 2024
Conexão com Amazon Keyspaces a partir do Amazon Elastic Kubernetes Service	Agora você pode seguir um tutorial passo a passo para se conectar ao Amazon Keyspaces a partir do Amazon EKS.	7 de fevereiro 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
APIs de ajuste de escala automático 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
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 em várias regiões](#)

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

[Cross-account acesso para Amazon Keyspaces e endpoints de interface VPC](#)

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 timestamps do lado do cliente do Amazon Keyspaces são timestamps em Cassandra-compatible nível de célula 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 tutorial passo a passo, 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 de FedRAMP-High conformidade. 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 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.	Step-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 `system.peers` tabela com as informações disponíveis do VPC endpoint da interface para melhorar o balanceamento de carga e aumentar a taxa de transferência. `read/write`

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 eficientemente 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

[Recuperação para um ponto no tempo \(PITR\) do Amazon Keyspaces](#)

O Amazon Keyspaces agora oferece recuperação para um ponto no tempo (PITR) para ajudar a proteger tabelas de operações de gravação ou exclusão acidentais, fornecendo o backups contínuos dos dados de tabelas.

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

<u>ajuste de escala automático do Amazon Keyspaces</u>	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 workloads variáveis em resposta ao tráfego real do aplicativo, ajustando a capacidade de throughput automaticamente.	23 de abril de 2020
<u>Endpoints da interface VPC (Nuvem Privada Virtual) do Amazon Keyspaces</u>	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
<u>Tag-based políticas de acesso</u>	Você agora pode usar tags de recursos em políticas do IAM para gerenciar o acesso ao Amazon Keyspaces.	8 de abril de 2020
<u>Tipo de dados do contador</u>	O Amazon Keyspaces agora ajuda você a coordenar incrementos e decréscimos nos valores das colunas usando contadores.	7 de abril de 2020
<u>Marcar recursos</u>	O Amazon Keyspaces agora permite que você rotule e categorize recursos usando tags.	31 de março de 2020

CloudFormation apoio	O Amazon Keyspaces agora ajuda você a automatizar a criação e o gerenciamento dos recursos da usando CloudFormation.	25 de março de 2020
Suporte a perfis do IAM e políticas do IAM e autenticação SigV4	Foram adicionadas informações sobre como você pode usar o AWS Identity and Access Management (IAM) para gerenciar permissões de acesso e implementar políticas de segurança para o Amazon Keyspaces e como usar o plug-in de autenticação do DataStax Java Driver for Cassandra para acessar programaticamente o Amazon Keyspaces usando funções do IAM e identidades federadas.	17 de março de 2020
Read/write modo de capacidade	O Amazon Keyspaces agora oferece suporte a dois modos de capacidade de read/write taxa de transferência. O modo read/write de capacidade controla como você é cobrado pela taxa de transferência de leitura e gravação e como a capacidade de transferência da tabela é gerenciada.	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.