aws

Guia do Desenvolvedor

Amazon MemoryDB para Redis

Amazon MemoryDB para Redis: Guia do Desenvolvedor

Table of Contents

O que é o MemoryDB para Redis?

O MemoryDB para Redis é um serviço de banco de dados na memória, durável, que oferece um desempenho ultrarrápido. Foi desenvolvido especificamente para aplicativos modernos com arquiteturas de microsserviços.

O MemoryDB é compatível com o Redis, um armazenamento de dados de código aberto popular, que permite que você crie aplicativos rapidamente usando as mesmas estruturas de dados, APIs e comandos flexíveis e fáceis de usar do Redis que eles já usam atualmente. Com o MemoryDB, todos os seus dados são armazenados na memória, o que permite que você obtenha latência de leitura de microssegundos e de gravação de um dígito de milissegundo, além do alto throughput. O MemoryDB também armazena dados de forma duradoura em várias zonas de disponibilidades (AZs) usando um log transacional Multi-AZ para permitir failover rápido, recuperação de banco de dados e reinicialização de nós.

Oferecendo desempenho na memória e durabilidade Multi-AZ, o MemoryDB pode ser usado como um banco de dados primário de alto desempenho para seus aplicativos de microsserviços, eliminando a necessidade de gerenciar separadamente um cache e um banco de dados durável.

Tópicos

Atributos do MemoryDB

O MemoryDB para Redis é um serviço de banco de dados na memória, durável, que oferece um desempenho ultrarrápido. Os atributos do MemoryDB incluem:

- Forte consistência para nós primários e consistência eventual garantida para nós de réplica. Para obter mais informações, consulte Consistência.
- Latências de leitura em microssegundos e gravação de um dígito em milissegundos com até 160 milhões de TPS por cluster.

- Estruturas de dados e APIs flexíveis e amigáveis do Redis. Crie facilmente novos aplicativos ou migre aplicativos Redis existentes com quase nenhuma modificação.
- Durabilidade de dados usando um log transacional Multi-AZ que fornece recuperação e reinicialização rápidas do banco de dados.
- Disponibilidade Multi-AZ com failover automático e detecção e recuperação de falhas nos nós.
- Ajuste a escala facilmente na horizontal ao adicionar e remover nós ou na vertical ao mudar para tipos de nós maiores ou menores. Você pode escalar o throughput de gravação adicionando fragmentos e escalar o throughput de leitura adicionando réplicas.
- Consistência de leitura após gravação para nós primários e consistência eventual garantida para nós de réplica.
- O MemoryDB suporta criptografia em trânsito, criptografia em repouso e autenticação de usuários por meio de Autenticação de usuários com listas de controle de acesso (ACLs).
- Snapshots automáticos no Amazon S3 com retenção por até 35 dias.
- Support para até 500 nós e mais de 100 TB de armazenamento por cluster (com 1 réplica por fragmento).
- Criptografia em trânsito com TLS e criptografia em repouso com chaves AWS KMS.
- Autenticação e autorização do usuário com o Redis Autenticação de usuários com listas de controle de acesso (ACLs).
- Support para tipos de instância Graviton2 da AWS.
- Integração com outros serviços AWS, como CloudWatch, Amazon VPC, CloudTrail e Amazon SNS, para monitoramento, segurança e notificações.
- Atualizações e patches de software totalmente gerenciados.
- Integração do Identity and Access Management (IAM) da AWS e controle de acesso baseado em tags para APIs de gerenciamento.

Componentes principais do MemoryDB

A seguir, encontre uma visão geral dos principais componentes de uma implantação do MemoryDB.

Tópicos

- Fragmentos
- Grupos de parâmetros
- Grupos de sub-redes
- Listas de controle de acesso
- Usuários

Clusters

Um cluster é uma coleção de um ou mais nós, servindo um único conjunto de dados. Um conjunto de dados do MemoryDB é particionado em fragmentos, e cada fragmento tem um nó primário e até 5 nós de réplica. Um nó primário atende solicitações de leitura e gravação, enquanto uma réplica atende somente solicitações de leitura. Um nó primário pode fazer o failover para um nó de réplica, promovendo essa réplica para o novo nó primário desse fragmento. O MemoryDB executa o Redis como seu mecanismo de banco de dados e, ao criar um cluster, você especifica a versão do Redis para seu cluster. Você pode modificar um cluster usando o AWS Management Console, a AWS CLI ou a API do MemoryDB.

Cada cluster do MemoryDB executa uma versão do mecanismo do Redis. Cada versão do mecanismo do Redis tem suporte a seus próprios recursos. Além disso, cada versão do mecanismo do Redis tem um conjunto de parâmetros em um grupo de parâmetros que controlam o comportamento dos clusters que ele gerencia.

A capacidade de computação e memória de um cluster é determinada por seu tipo de nó. Você pode selecionar o tipo de nó que melhor atenda às suas necessidades. Se as suas necessidades mudarem com o passar do tempo, você poderá alterar os tipos de nós. Para obter mais informações, consulte Tipos de nó compatíveis.

> **Note**
>
> Para obter informações sobre preços dos tipos de nós do MemoryDB, consulte Precificação do MemoryDB.

É possível executar um cluster em uma nuvem privada virtual (VPC) usando o serviço Amazon Virtual Private Cloud (Amazon VPC). Ao usar uma VPC, você tem controle sobre o ambiente de rede virtual. É possível escolher seu próprio intervalo de endereços IP, criar sub-redes e configurar o roteamento e listas de controle de acesso. O MemoryDB gerencia snapshots, patches de software,

detecção automática de falhas e recuperação. Não há custos adicionais para executar seu cluster em uma VPC. Para obter mais informações sobre como usar a Amazon VPC com o MemoryDB, consulte MemoryDB e Amazon VPC.

Muitas operações do MemoryDB são direcionadas a clusters:

- Criar um cluster
- Modificar um cluster
- Tirando snapshots de um cluster
- Excluir um cluster
- Visualizar os elementos em um cluster
- Adicionar ou remover tags de alocação de custos para e de um cluster

Para obter informações mais detalhadas, consulte os seguintes tópicos relacionados:

- Gerenciamento de clusters e Gerenciamento de nós

  Informações sobre clusters, nós e operações relacionadas.
- Resiliência no MemoryDB para Redis

  Informações sobre como melhorar a tolerância a falhas de seus clusters.

Nós

Um nó é o menor componente básico de uma implantação do MemoryDB e é executado usando uma instância do Amazon EC2. Cada nó executa a versão do Redis que foi escolhida quando você criou o cluster. Um nó pertence a um fragmento, que pertence a um cluster.

Cada nó executa uma instância do mecanismo e da versão escolhidos ao criar o cluster. Se necessário, você pode escalar os nós em um cluster para um tipo de instância diferente. Para obter mais informações, consulte Escalabilidade .

Cada nó em um cluster é do mesmo tipo de nó. Há suporte para vários tipos de nós, cada um com quantidades variadas de memória. Para obter uma lista dos tipos de nó compatíveis, consulte Tipos de nó compatíveis.

Para obter mais informações sobre nós, consulte Gerenciamento de nós.

Fragmentos

Um fragmento é um agrupamento de um a seis nós, com um deles servindo como nó de gravação principal e os outros cinco servindo como réplicas de leitura. Um cluster do MemoryDB sempre tem pelo menos um fragmento.

Os clusters do MemoryDB podem ter até 500 fragmentos, com seus dados particionados entre os fragmentos. Por exemplo, você pode optar por configurar um cluster de 500 nós que varia entre 83 fragmentos (uma primária e 5 réplicas por fragmento) e 500 fragmentos (primário único e sem réplicas). Verifique se existem endereços IP disponíveis suficientes para acomodar o aumento. As armadilhas comuns incluem as sub-redes no grupo de sub-redes têm um intervalo CIDR muito pequeno ou as sub-redes são compartilhadas e fortemente usadas por outros clusters.

Um fragmento de vários nós implementa a replicação por ter um nó primário de leitura/gravação e de 1 a 5 nós de réplicas. Para obter mais informações, consulte Noções básicas sobre a replicação do MemoryDB.

Para obter mais informações sobre estilhaços, consulte Operação com fragmentos.

Grupos de parâmetros

Os grupos de parâmetros são uma maneira fácil de gerenciar as configurações de runtime do Redis em seu cluster. Os parâmetros são usados para controlar o uso da memória, o tamanho dos itens e muito mais. Um grupo de parâmetros do MemoryDB é uma coleção nomeada de parâmetros específicos do mecanismo que você pode aplicar a um cluster, e todos os nós desse cluster são configurados exatamente da mesma maneira.

Para obter informações mais detalhadas sobre os grupos de parâmetros do MemoryDB, consulte Configuração de parâmetros do mecanismo usando grupos de parâmetros.

Grupos de sub-redes

Um grupo de sub-redes é um conjunto de sub-redes (normalmente privadas) que você pode designar para seus clusters em execução em um ambiente Amazon Virtual Private Cloud (VPC).

Ao criar um cluster em na Amazon VPC, você pode especificar um grupo de sub-rede ou usar o padrão fornecido. O MemoryDB usa esse grupo de sub-rede para escolher uma sub-rede e endereços IP dentro dessa sub-rede para associar aos seus nós.

Para obter informações mais detalhadas sobre os grupos de sub-rede do MemoryDB, consulte Sub-redes e grupos de sub-redes.

Listas de controle de acesso

Uma lista de controle de acesso é uma coleção de um ou mais usuários. As strings de acesso seguem as regras de ACL do Redis para autorizar o acesso do usuário aos comandos e dados do Redis.

Para obter informações mais detalhadas sobre as listas de controle de acesso do MemoryDB, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

Usuários

Um usuário tem um nome de usuário e uma senha e é usado para acessar dados e emitir comandos em seu cluster do MemoryDB. Um usuário é membro de uma Lista de Controle de Acesso (ACL), que pode ser usada para determinar as permissões para esse usuário nos clusters do MemoryDB. Para obter mais informações, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

Serviços relacionados

ElastiCache para Redis

Ao decidir se deseja usar o ElastiCache para Redis ou o MemoryDB para Redis, considere as seguintes comparações:

- O MemoryDB para Redis é um banco de dados na memória, durável para workloads que exigem um banco de dados primário ultrarrápido. Você deve considerar o uso do MemoryDB se sua workload exigir um banco de dados durável que ofereça desempenho ultrarrápido (leitura de microssegundos e latência de gravação em menos de 10 milissegundos). O MemoryDB também pode ser uma boa opção para seu caso de uso se você quiser criar uma aplicação usando estruturas de dados e APIs do Redis com um banco de dados primário e durável. Finalmente, você deve considerar o uso do MemoryDB para simplificar a arquitetura da aplicação e reduzir os custos substituindo o uso de um banco de dados por um cache para maior durabilidade e desempenho.
- O ElastiCache para Redis é um serviço comumente usado para armazenar dados de outros bancos de dados e armazenamentos de dados usando o Redis. Você deve considerar o ElastiCache para Redis para armazenar workloads em cache nas quais você deseja acelerar o acesso aos dados com seu banco de dados principal ou armazenamento de dados existente (desempenho de leitura e gravação em microssegundos). Você também deve considerar o ElastiCache para Redis para casos de uso em que você deseja usar as estruturas de dados e APIs

do Redis para acessar dados armazenados em um banco de dados ou armazenamento de dados primário.

Escolher regiões e zonas de disponibilidade

Os recursos de computação em nuvem da AWS são abrigados em instalações de datacenters de alta disponibilidade. Para fornecer escalabilidade e confiabilidade adicionais, estas instalações do datacenter estão localizadas em diferentes locais físicos. Esses locais são categorizados por regiões e zonas de disponibilidade.

As regiões da AWS são grandes e amplamente dispersas em locais geográficos separados. As zonas de disponibilidade são diferentes localizações dentro de uma região da AWS que são projetadas para serem isoladas de falhas em outras zonas de disponibilidade. Elas fornecem conectividade de rede de baixa latência e custo reduzido para outras zonas de disponibilidade na mesma região da AWS.

> **Important**
>
> Cada região é totalmente independente. Qualquer atividade do MemoryDB iniciada (por exemplo, criação de clusters) é executada somente na região padrão atual.

Para criar ou trabalhar com um cluster em uma região específica, use o endpoint do serviço regional correspondente. Para os endpoints de serviço, consulte Regiões e endpoints com suporte.

Localização dos seus nós

Qualquer cluster que tenha pelo menos uma réplica deve estar distribuído entre as AZs. A única maneira de localizar tudo em uma única AZ é com um cluster composto por fragmentos de nó único.

Ao localizar os nós em diferentes AZs, o MemoryDB elimina a chance de que uma falhaem uma AZ, como uma queda de energia, cause perda de disponibilidade.

- Criação de um cluster do MemoryDB
- Modificar um cluster do MemoryDB

Regiões e endpoints com suporte

O MemoryDB para Redis está disponível em várias regiões da AWS. Isso significa que você pode iniciar clusters do MemoryDB nos locais que atendem às suas necessidades. Por exemplo, você pode ativá-los na região da AWS mais próxima de seus clientes ou ativá-los em determinada região da AWS para atender a determinados requisitos legais. Além disso, à medida que o MemoryDB expande a disponibilidade para uma nova região da AWS, o MemoryDB oferece suporte às duas versões `MAJOR.MINOR` mais recentes à época para a nova região. Para obter mais informações sobre versões do MemoryDB, consulte Versões do mecanismo Redis.

Por padrão, os SDKs da AWS, a AWS CLI, a API do MemoryDB e o console do MemoryDB fazem referência à região Leste dos EUA (Norte da Virgínia). À medida que o MemoryDB expandir a disponibilidade para novas regiões, novos endpoints para essas regiões também estarão disponíveis para uso nas suas solicitações HTTP, nos SDKs da AWS, na AWS CLI e no console.

Cada região é projetada para ser completamente isolada das outras. Dentro de cada região há várias zonas de disponibilidade (AZ). Ao iniciar seus nós em diferentes AZs, você obtém a maior tolerância possível a falhas. Para obter mais informações sobre regiões e zonas de disponibilidade, consulte Escolher regiões e zonas de disponibilidade no início deste tópico.

Regiões em que o MemoryDB tem suporte

Nome da região/região	Endpoint	Protocolo	
Região Leste dos EUA (Ohio) `us-east-2`	`memory-db.us-east-2.amazonaws.com`	HTTPS	
Região Leste dos EUA (N. da Virgínia) `us-east-1`	`memory-db.us-east-1.amazonaws.com`	HTTPS	
Região Leste dos EUA (Norte da Califórnia) `us-west-1`	`memory-db.us-west-1.amazonaws.com`	HTTPS	

Nome da região/região	Endpoint	Protocolo	
Região Oeste dos EUA (Oregon) `us-west-2`	`memory-db.us-west-2.amazonaws.com`	HTTPS	
Região do Canadá (Central) `ca-central-1`	`memory-db.ca-central-1.amazonaws.com`	HTTPS	
Região Ásia-Pacífico (Hong Kong) `ap-east-1`	`memory-db.ap-eastl-1.amazonaws.com`	HTTPS	
Região Ásia-Pacífico (Mumbai) `ap-south-1`	`memory-db.ap-south-1.amazonaws.com`	HTTPS	
Região Ásia-Pacífico (Tóquio) `ap-northeast-1`	`memory-db.ap-northeast-1.amazonaws.com`	HTTPS	
Região Ásia-Pacífico (Seul) `ap-northeast-2`	`memory-db.ap-northeast-2.amazonaws.com`	HTTPS	
Região Ásia-Pacífico (Singapura) `ap-southeast-1`	`memory-db.ap-southeast-1.amazonaws.com`	HTTPS	

Nome da região/região	Endpoint	Protocolo	
Região Ásia-Pacífico (Sydney) `ap-southeast-2`	`memory-db.ap-southeast-2.amazonaws.com`	HTTPS	
Região Europa (Frankfurt) `eu-central-1`	`memory-db.eu-central-1.amazonaws.com`	HTTPS	
Região Europa (Irlanda) `eu-west-1`	`memory-db.eu-west-1.amazonaws.com`	HTTPS	
Região Europa (Londres) `eu-west-2`	`memory-db.eu-west-2.amazonaws.com`	HTTPS	
Região Europa (Paris) `eu-west-3`	`memory-db.eu-west-3.amazonaws.com`	HTTPS	
Região Europa (Estocolmo) `eu-north-1`	`memory-db.eu-north-1.amazonaws.com`	HTTPS	
Região Europa (Milão) `eu-south-1`	`memory-db.eu-south-1.amazonaws.com`	HTTPS	

Nome da região/região	Endpoint	Protocolo	
Região América do Sul (São Paulo) `sa-east-1`	`memory-db.sa-east-1.amazonaws.com`	HTTPS	
Região da China (Pequim) `cn-north-1`	`memory-db.cn-north-1.amazonaws.com.cn`	HTTPS	
Região da China (Ningxia) `cn-northwest-1`	`memory-db.cn-northwest-1.amazonaws.com.cn`	HTTPS	

Para obter uma tabela de produtos e serviços da AWS por região, consulte Produtos e serviços por região.

Para obter uma tabela das zonas de disponibilidade compatíveis dentro das regiões, consulte Sub-redes e grupos de sub-redes.

Acessando o MemoryDB

Cada endpoint do cluster do MemoryDB contém um endereço e uma porta. Esse endpoint de cluster oferece suporte ao protocolo Redis Cluster para permitir que os clientes descubram as funções, endereços IP e slots específicos de cada nó no cluster. Quando um nó primário falha e uma réplica é promovida em seu lugar, você pode se conectar ao endpoint do cluster para descobrir o novo primário usando o protocolo Redis Cluster.

Você precisa se conectar ao endpoint do cluster para descobrir os endpoints do nó usando os comandos cluster nodes ou cluster slots. Depois de descobrir o nó certo para uma chave, você pode se conectar diretamente ao nó para solicitações de leitura/gravação. Um cliente do Redis pode usar o endpoint do cluster para se conectar automaticamente ao nó correto.

Para solucionar problemas de nós específicos em um cluster, você também pode usar endpoints específicos de nós, mas eles não são necessários para o uso normal.

Para localizar os endpoints do cluster, consulte o seguinte:

- Localização do endpoint de um cluster do MemoryDB (AWS CLI)
- Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Para conectar-se a nós ou clusters, consulteConectando-se aos nós do MemoryDB usando redis-cli.

Segurança do MemoryDB

A segurança do MemoryDB é gerenciada em três níveis:

- Para controlar quem pode executar ações de gerenciamento em clusters e nós do MemoryDB, use o Gerenciamento de Identidade e Acesso (IAM)da AWS. Quando você se conecta à AWS usando credenciais do IAM, sua conta da AWS deve ter políticas do IAM que concedam as permissões necessárias para realizar operações. Para obter mais informações, consulte Gerenciamento de identidade e acesso no MemoryDB para Redis.
- Para controlar os níveis de acesso aos clusters, você cria usuários com permissões específicas e os atribui às listas de controle de acesso (ACL). A ACL, por sua vez, é então associada a um ou mais clusters. Para obter mais informações, consulte Autenticação de usuários com listas de controle de acesso (ACLs).
- Os clusters do MemoryDB devem ser criados em uma nuvem privada virtual (VPC) com base no serviço da Amazon VPC. Para controlar quais dispositivos e instâncias do Amazon EC2 podem abrir conexões com o endpoint e a porta do nó para clusters do MemoryDB em uma VPC, use um grupo de segurança da VPC. É possível estabelecer essas conexões de endpoint e porta usando Transport Layer Security (TLS)/Secure Sockets Layer (SSL). Além disso, as regras de firewall da sua empresa podem controlar se os dispositivos em execução na sua empresa podem abrir conexões com um cluster do MemoryDB. Para ter mais informações sobre VPCs, consulte MemoryDB e Amazon VPC.

Para obter informações sobre a configuração de segurança, consulte Segurança no MemoryDB para Redis.

Conceitos básicos do MemoryDB

Este exercício mostra as etapas para criar, conceder acesso, conectar-se e, finalmente, excluir um cluster do MemoryDB usando o console de gerenciamento do MemoryDB.

> **Note**
>
> Para este exercício, recomendamos que você use a opção Criação fácil ao criar um cluster e retorne às outras duas opções depois de explorar mais os atributos do MemoryDB.

Tópicos

Configuração

A seguir, você encontrará tópicos que descrevem as ações únicas que devem ser executadas para começar a usar o MemoryDB.

Tópicos

Crie sua AWS conta

Inscreva-se para um Conta da AWS

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

Para se inscrever em um Conta da AWS

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

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

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

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

Crie um usuário com acesso administrativo

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

Proteja seu Usuário raiz da conta da AWS

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

   Para obter ajuda ao fazer login usando o usuário raiz, consulte Fazer login como usuário raiz no Guia do usuário do Início de Sessão da AWS .
2. Ative a autenticação multifator (MFA) para o usuário raiz.c

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

Crie um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

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

2. No IAM Identity Center, conceda acesso administrativo a um usuário.

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

Faça login como usuário com acesso administrativo

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

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

Atribua acesso a usuários adicionais

1. No IAM Identity Center, crie um conjunto de permissões que siga as melhores práticas de aplicação de permissões com privilégios mínimos.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia AWS IAM Identity Center do usuário.

2. Atribua usuários a um grupo e, em seguida, atribua acesso de login único ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia AWS IAM Identity Center do usuário.

Conceder acesso programático

Os usuários precisam de acesso programático se quiserem interagir com pessoas AWS fora do AWS Management Console. A forma de conceder acesso programático depende do tipo de usuário que está acessando AWS.

Para conceder acesso programático aos usuários, selecione uma das seguintes opções:

Qual usuário precisa de acesso programático?	Para	Por
Identificação da força de trabalho (Usuários gerenciados no Centro de Identidade do IAM)	Use credenciais temporári as para assinar solicitações programáticas para AWS SDKs ou APIs. AWS CLI AWS	Siga as instruções da interface que deseja utilizar. • Para o AWS CLI, consulte Configurando o AWS CLI para uso AWS IAM Identity Center no Guia do AWS Command Line Interface usuário. • Para AWS SDKs, ferrament as e AWS APIs, consulte a autenticação do IAM Identity Center no Guia de referência de AWS SDKs e ferramentas.
IAM	Use credenciais temporári as para assinar solicitações programáticas para AWS SDKs ou APIs. AWS CLI AWS	Siga as instruções em Como usar credenciais temporárias com AWS recursos no Guia do usuário do IAM.
IAM	(Não recomendado) Use credenciais de longo prazo para assinar solicitaç ões programáticas para AWS SDKs AWS CLI ou APIs. AWS	Siga as instruções da interface que deseja utilizar. • Para isso AWS CLI, consulte Autenticação usando credenciais de usuário do IAM no Guia do AWS Command Line Interface usuário. • Para AWS SDKs e ferramentas, consulte Autenticar usando credencia is de longo prazo no Guia

Qual usuário precisa de acesso programático?	Para	Por
		de referência de AWS SDKs e ferramentas. • Para AWS APIs, consulte Gerenciamento de chaves de acesso para usuários do IAM no Guia do usuário do IAM.

Tópicos relacionados:

- O que é o IAM no Guia do usuário do IAM
- AWS Credenciais de segurança em referência AWS geral.

Configuração de permissões (somente novos usuários do MemoryDB)

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em AWS IAM Identity Center:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criar um perfil para um provedor de identidades de terceiros (federação) no Guia do usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

O MemoryDB para Redis cria e usa funções vinculadas a serviços para provisionar recursos e acessar outros recursos e serviços da AWS em seu nome. Para que o MemoryDB crie uma função vinculada ao serviço para você, use a AWS política -managed chamada. `AmazonMemoryDBFullAccess` Essa função é pré-provisionada com uma permissão que o serviço requer para criar uma função vinculada a serviço em seu nome.

Talvez você decida usar uma política gerenciada personalizada, em vez de uma política padrão. Nesse caso, confirme se você tem permissões para chamar a `iam:createServiceLinkedRole` ou se criou a função vinculada ao serviço MemoryDB.

Para mais informações, consulte:

- Criar uma política (IAM)
- Políticas (predefinidas) gerenciadas pela AWS do MemoryDB para Redis
- Uso de funções vinculadas ao serviço para o Amazon MemoryDB para Redis

Baixando e configurando a CLI AWS

O AWS CLI está disponível em http://aws.amazon.com/cli. Ela roda em 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 da interface de linha de comando da AWS
2. Siga as instruções para instalar a AWS CLI e configurar a CLI. AWS

Etapa 1: criar um cluster

Antes de criar um cluster para uso em produção, é óbvio que você precisa considerar como configurar o cluster para atender às suas necessidades de negócios. Esses problemas são abordados na seção Preparação de um cluster. Para os fins deste exercício de introdução, você pode aceitar os valores de configuração padrão onde se aplicarem.

O cluster que você criará estará ativo, e não em execução em uma sandbox. Você pagará as taxas de utilização padrão do MemoryDB pela instância até que a exclua. As cobranças totais serão mínimas (geralmente menos de um dólar) se você concluir o exercício descrito aqui em uma única sessão e excluir seu cluster quando terminar. Para obter mais informações sobre taxas de uso do MemoryDB, consulte MemoryDB.

Seu cluster é iniciado em uma nuvem privada virtual (VPC) com base no serviço da Amazon VPC.

Criação de um cluster do MemoryDB

Os exemplos a seguir mostram como criar um cluster usando a API AWS Management Console AWS CLI e MemoryDB.

Criação de um cluster (console)

Para criar um cluster usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, selecione Clusters e depois Criar.

Easy create

1. Preencha a seção Configuração. Isso define o tipo de nó e a configuração padrão do seu cluster. Selecione entre as seguintes opções o tamanho de memória e o desempenho de rede adequados que você precisa usar:
   - Produção
   - Dev/Teste
   - Demonstração
2. Preencha a seção Informações do cluster.

a. Em Nome, insira um nome para o cluster.

   As restrições de nomenclatura de cluster são as seguintes:

   - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
   - Deve começar com uma letra.
   - Não podem conter dois hifens consecutivos.
   - Não podem terminar com um hífen.

b. Na caixa Descrição, insira uma descrição para esse cluster.

3. Preencha a seção Grupos de sub-rede:

   - Para Grupos de sub-rede, crie um novo grupo de sub-rede ou escolha um existente na lista disponível que você deseja aplicar a esse cluster. Se você estiver criando um novo:

     - Insira um Nome
     - Insira uma Descrição
     - Se você habilitou o Multi-AZ, o grupo de sub-redes deve conter pelo menos duas sub-redes que residem em zonas de disponibilidade diferentes. Para ter mais informações, consulte Sub-redes e grupos de sub-redes.
     - Se você estiver criando um novo grupo de sub-redes e não tiver uma VPC existente, você deverá criar uma VPC. Para obter mais informações, consulte O que é a Amazon VPC? no Guia do usuário da Amazon VPC.

4. Em Pesquisa vetorial, você pode Habilitar o recurso de pesquisa vetorial para armazenar incorporações vetoriais e realizar pesquisas vetoriais. Isso fixará os valores de Compatibilidade de versões do Redis, Grupos de parâmetros e Fragmentos. Para ter mais informações, consulte Pesquisa vetorial.

5. Visualizar configurações padrão:

   Ao usar a Criação fácil, as configurações restantes do cluster são definidas por padrão. Algumas dessas configurações podem ser alteradas após a criação, conforme indicado em Editável após a criação.

6. Para Tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar seus clusters ou monitorar seus AWS custos.

7. Revise todas as suas entradas e opções e faça as correções necessárias. Quando estiver pronto, escolha Criar para executar seu cluster ou Cancelar para cancelar a operação.

Assim que o status do seu cluster for available, você poderá conceder acesso ao EC2 a ele, conectar-se a ele e começar a usá-lo. Para mais informações, consulte Etapa 2: autorizar o acesso ao cluster.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver ativo, mesmo que você não o esteja usando ativamente. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Create new cluster

1. Preencha a seção Informações do cluster.
   a. Em Nome, insira um nome para o cluster.

      As restrições de nomenclatura de cluster são as seguintes:

      - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
      - Deve começar com uma letra.
      - Não podem conter dois hifens consecutivos.
      - Não podem terminar com um hífen.

   b. Na caixa Descrição, insira uma descrição para esse cluster.
2. Preencha a seção Grupos de sub-rede:
   - Para Grupos de sub-rede, crie um novo grupo de sub-rede ou escolha um existente na lista disponível que você deseja aplicar a esse cluster. Se você estiver criando um novo:
     - Insira um Nome
     - Insira uma Descrição
     - Se você habilitou o Multi-AZ, o grupo de sub-redes deve conter pelo menos duas sub-redes que residem em zonas de disponibilidade diferentes. Para ter mais informações, consulte Sub-redes e grupos de sub-redes.

- Se você estiver criando um novo grupo de sub-redes e não tiver uma VPC existente, você deverá criar uma VPC. Para obter mais informações, consulte O que é a Amazon VPC? no Guia do usuário da Amazon VPC.

3. Complete a seção Configurações de Cluster:

   a. Em Habilitar recurso de pesquisa vetorial, você pode habilitar esse recurso para armazenar incorporações vetoriais e fazer pesquisas vetoriais. Isso fixará os valores de Compatibilidade de versões do Redis, Grupos de parâmetros e Fragmentos. Para ter mais informações, consulte Pesquisa vetorial.

   b. Para compatibilidade com a versão do Redis, aceite o padrão`6.2`.

   c. Em Porta, aceite a porta Redis padrão 6379 ou, se tiver um motivo para usar uma porta diferente, insira o número da porta.

   d. Em Grupo de parâmetros, se tiver habilitado a pesquisa vetorial, use `default.memorydb-redis7.search.preview`. Caso contrário, aceite o grupo de parâmetros `default.memorydb-redis7`.

      Os grupo de parâmetros controlam os parâmetros de runtime do seu cluster. Para ter mais informações sobre grupos de parâmetros, consulte Parâmetros específicos do Redis.

   e. Para Tipo de nó, escolha um valor para o tipo de nó (junto com o tamanho de memória associado) que você deseja.

      Se você escolher um tipo de nó da família r6gd, a classificação de dados em níveis será ativada automaticamente, dividindo o armazenamento de dados entre memória e SSD. Para ter mais informações, consulte Classificação de dados em níveis.

   f. Em Número de fragmentos, escolha o número de fragmentos desejado para este cluster. Para maior disponibilidade de seus clusters, recomendamos que você adicione pelo menos 2 fragmentos.

      É possível alterar dinamicamente o número de fragmentos no cluster. Para ter mais informações, consulte Escalabilidade de clusters do MemoryDB.

   g. Em Réplicas por fragmento, escolha o número de nós de réplica de leitura desejados em cada fragmento.

      Existem as seguintes restrições:

- Se você tiver o Multi-AZ habilitado, verifique se tem pelo menos uma réplica por fragmento.
- O número de réplicas é o mesmo para cada fragmento ao criar o cluster usando o console.

h. Escolha Avançar.

i. Conclua a seção Configurações avançadas:

i. Em Grupos de segurança, escolha os grupos de segurança desejados para esse cluster. Um grupo de segurança atua como um firewall para controlar o acesso à rede ao cluster. É possível usar o grupo de segurança padrão para sua VPC ou criar um novo.

Para obter mais informações sobre grupos de segurança, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC.

ii. Para criptografar seus dados, você tem as seguintes opções:

- Criptografia em repouso: permite a criptografia de dados armazenados em disco. Para obter mais informações, consulte Criptografia em repouso.

> **Note**
>
> Você tem a opção de fornecer uma chave de criptografia diferente da padrão escolhendo a chave KMS de AWS propriedade gerenciada pelo cliente e escolhendo a chave.

- Criptografia em trânsito: permite a criptografia de dados na conexão. Se você selecionar nenhuma criptografia, será criada uma lista de controle de acesso aberta chamada “acesso aberto” com um usuário padrão. Para ter mais informações, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

iii. Para Snapshot, opcionalmente, especifique um período de retenção de snapshot e uma janela de snapshot. Por padrão, a opção Ativar snapshots automáticos está pré-selecionada.

iv. Para Janela de manutenção, opcionalmente, especifique uma janela de manutenção. A Janela de manutenção é o tempo, geralmente de uma hora de duração, a cada semana quando o MemoryDB agenda a manutenção do sistema

para seu cluster. É possível permitir que o MemoryDB escolha o dia e a hora da sua janela de manutenção (Sem preferência) ou é possível escolher o dia, a hora e a duração por conta própria (Especificar janela de manutenção). Se você escolher Especificar janela de manutenção, nas listas, escolha Dia de início, Hora de início e Duração (em horas) para sua janela de manutenção. Todos os horários são em UCT.

Para ter mais informações, consulte Gerenciamento da manutenção.

v. Em Notificações, escolha um tópico existente do Amazon Simple Notification Service (Amazon SNS) ou escolha a entrada de ARN manual e insira o nome de recurso da Amazon (ARN) do tópico. O Amazon SNS permite que você envie notificações para dispositivos inteligentes conectados à Internet. O padrão é desabilitar notificações. Para obter mais informações, consulte https://aws.amazon.com/sns/.

vi. Para Tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar seus clusters ou monitorar seus AWS custos.

j. Revise todas as suas entradas e opções e faça as correções necessárias. Quando estiver pronto, escolha Criar para executar seu cluster ou Cancelar para cancelar a operação.

Assim que o status do seu cluster for available, você poderá conceder acesso ao EC2 a ele, conectar-se a ele e começar a usá-lo. Para mais informações, consulte Etapa 2: autorizar o acesso ao cluster.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver ativo, mesmo que você não o esteja usando ativamente. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Restore from snapshots

Em Fonte do snapshot, escolha o snapshot de origem do qual os dados serão migrados. Para ter mais informações, consulte Snapshots e restauração .

> **Note**
>
> Se quiser que seu novo cluster tenha a pesquisa vetorial habilitada, o snapshot de origem também deverá ter a pesquisa vetorial habilitada.

O cluster de destino usa por padrão as configurações do cluster de origem. Outra opção é alterar as seguintes configurações no cluster de destino:

1. Informações do cluster

   a. Em Nome, insira um nome para o cluster.

      As restrições de nomenclatura de cluster são as seguintes:

      - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
      - Deve começar com uma letra.
      - Não podem conter dois hifens consecutivos.
      - Não podem terminar com um hífen.

   b. Na caixa Descrição, insira uma descrição para esse cluster.

2. Grupos de sub-redes

   - Para Grupos de sub-rede, crie um novo grupo de sub-rede ou escolha um existente na lista disponível que você deseja aplicar a esse cluster. Se você estiver criando um novo:

     - Insira um Nome
     - Insira uma Descrição
     - Se você habilitou o Multi-AZ, o grupo de sub-redes deve conter pelo menos duas sub-redes que residem em zonas de disponibilidade diferentes. Para ter mais informações, consulte Sub-redes e grupos de sub-redes.
     - Se você estiver criando um novo grupo de sub-redes e não tiver uma VPC existente, você deverá criar uma VPC. Para obter mais informações, consulte O que é a Amazon VPC? no Guia do usuário da Amazon VPC.

3. Configurações do cluster

   a. Em Habilitar recurso de pesquisa vetorial, você pode habilitar esse recurso para armazenar incorporações vetoriais e fazer pesquisas vetoriais. Isso fixará os valores de

Compatibilidade de versões do Redis, Grupos de parâmetros e Fragmentos. Para ter mais informações, consulte Pesquisa vetorial.

b. Para compatibilidade com a versão do Redis, aceite o padrão `6.2`.

c. Em Porta, aceite a porta Redis padrão 6379 ou, se tiver um motivo para usar uma porta diferente, insira o número da porta.

d. Em Grupo de parâmetros, se tiver habilitado a pesquisa vetorial, use `default.memorydb-redis7.search.preview`. Caso contrário, aceite o grupo de parâmetros `default.memorydb-redis7`.

Os grupo de parâmetros controlam os parâmetros de runtime do seu cluster. Para ter mais informações sobre grupos de parâmetros, consulte Parâmetros específicos do Redis.

e. Para Tipo de nó, escolha um valor para o tipo de nó (junto com o tamanho de memória associado) que você deseja.

Se você escolher um tipo de nó da família r6gd, a classificação de dados em níveis será ativada automaticamente, dividindo o armazenamento de dados entre memória e SSD. Para ter mais informações, consulte Classificação de dados em níveis.

f. Em Número de fragmentos, escolha o número de fragmentos desejado para este cluster. Para maior disponibilidade de seus clusters, recomendamos que você adicione pelo menos 2 fragmentos.

É possível alterar dinamicamente o número de fragmentos no cluster. Para ter mais informações, consulte Escalabilidade de clusters do MemoryDB.

g. Em Réplicas por fragmento, escolha o número de nós de réplica de leitura desejados em cada fragmento.

Existem as seguintes restrições:

- Se você tiver o Multi-AZ habilitado, verifique se tem pelo menos uma réplica por fragmento.
- O número de réplicas é o mesmo para cada fragmento ao criar o cluster usando o console.

h. Escolha Avançar.

i. Configurações avançadas

i. Em Grupos de segurança, escolha os grupos de segurança desejados para esse cluster. Um grupo de segurança atua como um firewall para controlar o acesso à rede ao cluster. É possível usar o grupo de segurança padrão para sua VPC ou criar um novo.

   Para obter mais informações sobre grupos de segurança, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC.

ii. Para criptografar seus dados, você tem as seguintes opções:

   - Criptografia em repouso: permite a criptografia de dados armazenados em disco. Para obter mais informações, consulte Criptografia em repouso.

     > **Note**
     >
     > Você tem a opção de fornecer uma chave de criptografia diferente da padrão escolhendo a chave KMS de AWS propriedade gerenciada pelo cliente e escolhendo a chave.

   - Criptografia em trânsito: permite a criptografia de dados na conexão. Se você selecionar nenhuma criptografia, será criada uma lista de controle de acesso aberta chamada “acesso aberto” com um usuário padrão. Para ter mais informações, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

iii. Para Snapshot, opcionalmente, especifique um período de retenção de snapshot e uma janela de snapshot. Por padrão, a opção Ativar snapshots automáticos está pré-selecionada.

iv. Para Janela de manutenção, opcionalmente, especifique uma janela de manutenção. A Janela de manutenção é o tempo, geralmente de uma hora de duração, a cada semana quando o MemoryDB agenda a manutenção do sistema para seu cluster. É possível permitir que o MemoryDB escolha o dia e a hora da sua janela de manutenção (Sem preferência) ou é possível escolher o dia, a hora e a duração por conta própria (Especificar janela de manutenção). Se você escolher Especificar janela de manutenção, nas listas, escolha Dia de início, Hora de início e Duração (em horas) para sua janela de manutenção. Todos os horários são em UCT.

   Para ter mais informações, consulte Gerenciamento da manutenção.

v. Em Notificações, escolha um tópico existente do Amazon Simple Notification Service (Amazon SNS) ou escolha a entrada de ARN manual e insira o nome de recurso da Amazon (ARN) do tópico. O Amazon SNS permite que você envie notificações para dispositivos inteligentes conectados à Internet. O padrão é desabilitar notificações. Para obter mais informações, consulte https://aws.amazon.com/sns/.

vi. Para Tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar seus clusters ou monitorar seus AWS custos.

j. Revise todas as suas entradas e opções e faça as correções necessárias. Quando estiver pronto, escolha Criar para executar seu cluster ou Cancelar para cancelar a operação.

Assim que o status do seu cluster for available, você poderá conceder acesso ao EC2 a ele, conectar-se a ele e começar a usá-lo. Para mais informações, consulte Etapa 2: autorizar o acesso ao cluster.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver ativo, mesmo que você não o esteja usando ativamente. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Criação de um cluster (AWS CLI)

Para criar um cluster usando o AWS CLI, consulte `create-cluster`. Veja um exemplo a seguir:

Para Linux, macOS ou Unix:

```
aws memorydb create-cluster \
    --cluster-name my-cluster \
    --node-type db.r6g.large \
    --acl-name my-acl \
    --subnet-group my-sg
```

Para Windows:

```
aws memorydb create-cluster ^
   --cluster-name my-cluster ^
   --node-type db.r6g.large ^
   --acl-name my-acl ^
   --subnet-group my-sg
```

Você deve obter a seguinte resposta em JSON:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "creating",
        "NumberOfShards": 1,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxxxxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
```

```
        "ACLName": "my-acl",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
}
```

Você pode começar a usar o cluster quando seu status mudar para `available`.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver ativo, mesmo que você não o esteja usando ativamente. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Criação de um cluster (API do MemoryDB)

Para criar um cluster usando a API MemoryDB, use a CreateClusteração.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver, mesmo que você não o esteja usando. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Configuração de autenticação

Para obter informações sobre como configurar a autenticação para seu cluster, consulte Autenticação com o IAM e Autenticação de usuários com listas de controle de acesso (ACLs).

Etapa 2: autorizar o acesso ao cluster

Esta seção supõe que você esteja familiarizado com a execução e a conexão com instâncias do Amazon EC2. Para obter mais informações, consulte o Guia de conceitos básicos do Amazon EC2.

Todos os clusters do MemoryDB foram criados para serem acessados a partir de uma instância do Amazon EC2. Eles também podem ser acessados por aplicativos em contêineres ou de tecnologia sem servidor executados no Amazon Elastic Container Service ou AWS Lambda. O cenário mais comum é acessar um cluster do MemoryDB a partir de uma instância do Amazon EC2 na mesma Amazon Virtual Private Cloud (Amazon VPC), o que será o caso para esse exercício.

Antes de poder se conectar a um cluster a partir de uma instância do EC2, você deve autorizar a instância do EC2 para acessar o cluster.

O caso de uso mais comum é quando uma aplicação implantada em uma instância do EC2 precisa se conectar a um cluster na mesma VPC. A maneira mais simples de gerenciar o acesso entre instâncias do EC2 e clusters na mesma VPC é fazer o seguinte:

1. Crie um grupo de segurança de VPC para o seu cluster. Esse grupo de segurança pode ser usado para restringir o acesso aos clusters. Por exemplo, é possível criar uma regra personalizada para esse grupo de segurança que permite o acesso TCP usando a porta atribuída ao cluster quando você o criou e um endereço IP que será usado para acessar o cluster.

   A porta padrão dos clusters do MemoryDB é 6379.
2. Crie um grupo de segurança de VPC para suas instâncias do EC2 (servidores Web e de aplicativos). Esse grupo de segurança pode, se necessário, permitir o acesso à instância do EC2 da Internet através da tabela de rotas da VPC. Por exemplo, você pode definir regras nesse grupo de segurança para permitir o acesso TCP à instância do EC2 pela porta 22.
3. Crie regras personalizadas no grupo de segurança para o seu cluster que permitam conexões do grupo de segurança que você criou para suas instâncias do EC2. Isso permitiria que qualquer membro de grupo de segurança acessasse os clusters.

Para criar uma regra em um grupo de segurança de VPC que permita conexões de outro grupo de segurança

1. Faça login no AWS Management Console e abra o console da Amazon VPC em https://console.aws.amazon.com/vpc.

2. No painel de navegação esquerdo, escolha Security Groups.
3. Selecione ou crie um grupo de segurança que você usará para seus clusters. Em Regras de entrada, selecione Editar regras de entrada e escolha Adicionar regra. Esse grupo de segurança permitirá o acesso a membros de outro grupo de segurança.
4. Em Tipo, escolha Regra TCP personalizada.

   a. Para Port Range, especifique a porta que você usou quando criou seu cluster.

      A porta padrão dos clusters do MemoryDB é 6379.

   b. Na caixa Source, comece a digitar o ID do grupo de segurança. Na lista, selecione o grupo de segurança que você usará para o suas instâncias do Amazon EC2.
5. Escolha Save quando terminar.

Depois de habilitar o acesso, você agora estará pronto para se conectar ao cluster, conforme discutido na próxima seção.

Para obter informações sobre como acessar seu cluster MemoryDB de uma Amazon VPC diferente, de uma AWS região diferente ou até mesmo de sua rede corporativa, consulte o seguinte:

- Padrões de acesso para acessar um cluster do MemoryDB em uma Amazon VPC
- Acessar recursos do MemoryDB de fora da AWS

Etapa 3: Conectar-se ao cluster

Antes de continuar, conclua Etapa 2: autorizar o acesso ao cluster.

Esta seção supõe que você tenha criado uma instância do Amazon EC2 e possa conectar-se a ela. Para obter instruções sobre como fazer isso, consulte o Guia de conceitos básicos do Amazon EC2.

Uma instância do Amazon EC2 pode se conectar a um cluster apenas se você tiver autorização para isso.

Localize o endpoint de seu cluster

Quando seu cluster estiver no estado disponível e você tiver autorizado o acesso a ele, será possível fazer login em uma instância do Amazon EC2 e se conectar a ela. Para isso, primeiro você deve determinar o endpoint.

Para explorar mais sobre como localizar os endpoints, consulte o seguinte:

- Localização do endpoint para um cluster do MemoryDB (AWS Management Console)
- Localização do endpoint de um cluster do MemoryDB (AWS CLI)
- Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Conecte-se a um cluster do MemoryDB (Linux)

Agora que você tem o endpoint necessário, você pode fazer o login em uma instância do EC2 e se conectar ao cluster. No exemplo a seguir, você usa o serviço cli para se conectar a um cluster usando o Ubuntu 22. A versão mais recente da cli também oferece suporte a SSL/TLS para conexão com clusters habilitados para criptografia e autenticação.

Conectando-se aos nós do MemoryDB usando redis-cli

Para acessar dados dos nós do MemoryDB, use clientes que trabalhem com Secure Socket Layer (SSL). Você também pode usar a redis-cli com TLS/SSL no Amazon Linux e no Amazon Linux 2.

Para usar a redis-cli para se conectar a um cluster do MemoryDB no Amazon Linux 2 ou no Amazon Linux

1. Baixe e compile o utilitário redis-cli. Esse utilitário está incluído na distribuição do software Redis.

2. No prompt de comando da sua instância do EC2, digite os comandos apropriados para a versão do Linux que você está usando.

   Amazon Linux 2023

   Se estiver usando o Amazon Linux 2023, digite o seguinte:

   ```
   sudo yum install redis6 -y
   ```

   Em seguida, digite o comando a seguir, substituindo o endpoint do cluster e da porta pelo que é mostrado neste exemplo.

   ```
   redis-cli -h Primary or Configuration Endpoint --tls -p 6379
   ```

   Para obter mais informações sobre como localizar o endpoint, consulte Localize seus endpoints de nó.

   Amazon Linux 2

   Se estiver usando o Amazon Linux 2, digite o seguinte:

   ```
   sudo yum -y install openssl-devel gcc
   wget http://download.redis.io/redis-stable.tar.gz
   tar xvzf redis-stable.tar.gz
   cd redis-stable
   make distclean
   make redis-cli BUILD_TLS=yes
   sudo install -m 755 src/redis-cli /usr/local/bin/
   ```

   Amazon Linux

   Se estiver usando o Amazon Linux, digite o seguinte:

   ```
   sudo yum install gcc jemalloc-devel openssl-devel tcl tcl-devel clang wget
   wget http://download.redis.io/redis-stable.tar.gz
   tar xvzf redis-stable.tar.gz
   cd redis-stable
   make redis-cli CC=clang BUILD_TLS=yes
   sudo install -m 755 src/redis-cli /usr/local/bin/
   ```

No Amazon Linux, também pode ser necessário executar as seguintes etapas adicionais:

```
sudo yum install clang
CC=clang make
sudo make install
```

3. Depois de baixar e instalar o utilitário redis-cli, é recomendável executar o comando opcional. `make-test`
4. Para se conectar a um cluster com criptografia e autenticação ativadas, digite este comando:

```
redis-cli -h Primary or Configuration Endpoint --tls -a 'your-password' -p 6379
```

> **Note**
>
> Se você instalar o redis6 no Amazon Linux 2023, agora poderá usar `redis6-cli` o comando em vez de: `redis-cli`
>
> ```
> redis6-cli -h Primary or Configuration Endpoint --tls -p 6379
> ```

Etapa 4: excluir um cluster

Enquanto um cluster estiver no estado disponível, você será cobrado por ele, independentemente de o estar ou não. Para interromper as cobranças, exclua o cluster.

> **Warning**
>
> Ao excluir um cluster do MemoryDB, seus snapshots manuais são retidos. Também é possível criar um snapshot final antes que o cluster seja excluído. Os snapshots automáticos não são retidos. Para ter mais informações, consulte Snapshots e restauração .

Usando o AWS Management Console

O procedimento a seguir exclui um único cluster da sua implantação. Para excluir vários clusters, repita o procedimento para cada cluster que deseja excluir. Você não precisa esperar a finalização da exclusão de um cluster antes de iniciar o procedimento para excluir outro.

Para excluir um cluster

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para escolher o cluster a ser excluído, selecione o botão de opção ao lado do nome do cluster na lista de clusters. Nesse caso, o nome do cluster do que você criou em Etapa 1: criar um cluster.
3. Em Ações, escolha Excluir.
4. Primeiro, escolha se deseja criar um snapshot do cluster antes de excluí-lo e, em seguida, insira `delete` na caixa de confirmação e Excluir para excluir o cluster, ou escolha Cancelar para manter o cluster.

   Se você escolheu Excluir, o status do cluster muda para excluindo.

Assim que o cluster não estiver mais relacionado na lista de clusters, você para de ser cobrado por ele.

Usando o AWS CLI

O código a seguir exclui o cluster `my-cluster`. Neste caso, substitua `my-cluster` pelo nome do cluster do que você criou em Etapa 1: criar um cluster.

```
aws memorydb delete-cluster --cluster-name my-cluster
```

A operação `delete-cluster` da CLI exclui apenas um cluster. Para excluir vários clusters, chame `delete-cluster` para cada cluster que você deseja excluir. Você não precisa esperar a finalização da exclusão de um cluster antes de excluir outro.

Para Linux, macOS ou Unix:

```
aws memorydb delete-cluster \
    --cluster-name my-cluster \
    --region us-east-1
```

Para Windows:

```
aws memorydb delete-cluster ^
    --cluster-name my-cluster ^
```

```
    --region us-east-1
```

Para ter mais informações, consulte `delete-cluster`.

Usando a API do MemoryDB

O código a seguir exclui o cluster `my-cluster`. Neste caso, substitua `my-cluster` pelo nome do cluster do que você criou em Etapa 1: criar um cluster.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DeleteCluster
    &ClusterName=my-cluster
    &Region=us-east-1
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210802T220302Z
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210802T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210802T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

A operação `DeleteCluster` da API exclui apenas um cluster. Para excluir vários clusters, chame `DeleteCluster` para cada cluster que você deseja excluir. Você não precisa esperar a finalização da exclusão de um cluster antes de excluir outro.

Para obter mais informações, consulte DeleteCluster.

O que faço agora?

Agora que tentou o exercício de Conceitos básicos, você pode explorar as seções a seguir para saber mais sobre o MemoryDB e as ferramentas disponíveis:

- Começando com AWS
- Ferramentas para a Amazon Web Services
- AWS Command Line Interface
- Referência da API do MemoryDB para Redis.

Gerenciamento de nós

Um nó é o menor bloco de construção de uma implantação do MemoryDB para Redis. Um nó pertence a um fragmento, que pertence a um cluster. Cada nó executa o mecanismo que foi escolhido quando o cluster foi criado ou modificado pela última vez. Cada nó possui seu próprio nome DNS (Serviço de Nomes de Domínio) e porta. Há suporte para vários tipos de nós do MemoryDB, cada um com quantidades variáveis de memória associada e potência computacional.

Tópicos

Algumas operações importantes que envolvem nós são as seguintes:

- Adicição e Remoção de nós de um cluster
- Escalabilidade
- Encontrar endpoints de conexão

Nós e fragmentos do MemoryDB

Um fragmento é um arranjo hierárquico de nós, cada um envolvido em um cluster. Fragmentos oferecem suporte para replicação. Dentro de um fragmento, um nó funciona como o nó primário de leitura/gravação. Todos os outros nós em um fragmento funcionam como réplicas somente leitura do nó primário. O MemoryDB oferece suporte a vários fragmentos em um cluster. Esse suporte permite particionar os dados em um cluster do MemoryDB.

O MemoryDB oferece suporte à replicação por meio de fragmentos. A operação da API DescribeClusterslista os fragmentos com os nós membros, os nomes dos nós, os endpoints e também outras informações.

Depois que um cluster do MemoryDB é criado, ele pode ser alterado (reduzido ou aumentado). Para obter mais informações, consulte Substituição de nós e Escalabilidade .

Ao criar um novo cluster, você pode preenchê-lo com dados do cluster antigo para que ele não fique vazio. Fazer isso pode ser útil se você precisar alterar o tipo de nó, a versão do mecanismo ou migrar da Amazon ElastiCache para o Redis. Para obter mais informações, consulte Restauração a partir de um snapshot e Obtenção manual de snapshots.

Tipos de nó compatíveis

O MemoryDB suporta os seguintes tipos de nós.

Otimizado para memória

Tipo de instância	Largura de banda da linha de base (Gbps)	Expansão da largura de banda (Gbps)	Multiplexação de E/S aprimorada (Redis 7.0.4+)	Versão mínima do motor
db.r7g.large	0,937	12,5	Não	6.2
db.r7g.xlarge	1.876	12,5	Não	6.2
db.r7g.2xlarge	3,75	15	Sim	6.2
db.r7g.4xlarge	7,5	15	Sim	6.2
db.r7g.8xlarge	15	N/D	Sim	6.2
db.r7g.12xlarge	22,5	N/D	Sim	6.2
db.r7g.16xlarge	30	N/D	Sim	6.2
db.r6g.large	0.75	10.0	Não	6.2
db.r6g.xlarge	1,25	10.0	Não	6.2
db.r6g.2xlarge	2,5	10.0	Sim	6.2
db.r6g.4xlarge	5,0	10.0	Sim	6.2
db.r6g.8xlarge	12	N/D	Sim	6.2
db.r6g.12xlarge	20	N/D	Sim	6.2
db.r6g.16xlarge	25	N/D	Sim	6.2

Otimizada para memória com classificação de dados em níveis

Tipo de instância	Largura de banda da linha de base (Gbps)	Expansão da largura de banda (Gbps)	Multiplexação de E/S aprimorada (Redis 7.0.4+)	Versão mínima do motor
db.r6gd.xlarge	1,25	10	Não	6.2
db.r6gd.2xlarge	2,5	10	Não	6.2
db.r6gd.4xlarge	5,0	10	Não	6.2
db.r6gd.8xlarge	12	N/D	Não	6.2

Nodos de uso geral

Tipo de instância	Largura de banda da linha de base (Gbps)	Expansão da largura de banda (Gbps)	Multiplexação de E/S aprimorada (Redis 7.0.4+)	Versão mínima do motor
db.t4g.small	0,128	5,0	Não	6.2
db.t4g.medium	0,256	5,0	Não	6.2

Para saber a disponibilidade AWS da região, consulte os preços do MemoryDB for Redis

Todos os tipos de nó são criados em uma nuvem privada virtual (VPC).

Nós reservados do MemoryDB

Os nós reservados fornecem um desconto significativo em comparação com os preços de nós sob demanda. Os nós reservados não são nós físicos, mas um desconto na fatura aplicado na sua conta pelo uso de nós sob demanda. Os descontos para nós reservados estão vinculados ao tipo de nó e à região da AWS .

O processo geral para trabalhar com nós reservados é o seguinte:

- Analise as informações sobre ofertas de nós reservados disponíveis
- Compre uma oferta de nó reservado usando o AWS Management Console, AWS Command Line Interface ou SDK
- Analise as informações sobre seus nós reservados existentes

Tópicos

Visão geral de nós reservados

Ao comprar um nó reservado do MemoryDB, você adquire um compromisso de obter uma taxa com desconto sobre um tipo específico de nó pela duração do nó reservado. Para usar um nó reservado do MemoryDB, crie um nó como você faria para um nó sob demanda. O novo nó que você criar deve corresponder exatamente às especificações do nó reservado. Se as especificações do novo nó corresponderem a um nó reservado existente em sua conta, você será cobrado de acordo com a tarifa com desconto oferecida para o nó reservado. Caso contrário, uma taxa sob demanda será cobrada para o nó. Você pode usar a API AWS Management Console AWS CLI, a ou MemoryDB para listar e comprar ofertas de nós reservados disponíveis.

O MemoryDB oferece nós reservados para os nós R7g, R6g e R6gd (com classificação de dados em camadas) otimizados para memória. Para obter informações sobre preços, consulte Definição de preços de MemoryDB para Redis.

Tipos de oferta

Os nós reservados estão disponíveis em três variedades: sem pagamento adiantado, com pagamento adiantado parcial e com pagamento adiantado integral. Esses tipos permitem otimizar os custos do MemoryDB para Redis com base no uso esperado.

Sem entrada – essa opção fornece acesso ao nó reservado sem a necessidade de entrada de pagamento. Seu nó reservado sem entrada de pagamento será cobrado de acordo com uma taxa horária com desconto por cada hora dentro do período de vigência, independentemente do uso, e não é necessária entrada.

Pagamento adiantado parcial – essa opção requer que uma parte do nó reservado seja paga antecipadamente. As horas restantes do período de vigência serão cobradas com base em uma taxa horária com desconto, independentemente do uso.

Pagamento adiantado integral – o pagamento integral é feito no início do período de vigência, sem outros custos ou cobranças por hora incorridos pelo restante do período, independentemente do número de horas usadas.

Todos os três tipos de ofertas estão disponíveis para períodos de vigência de um e três anos.

Tamanho de nós reservados flexíveis

Ao adquirir um nó reservado, uma das especificações feitas é o tipo de nó, por exemplo, db.r6g.xlarge. Para obter mais informações sobre os tipos de nós, consulte precificação do MemoryDB para Redis.

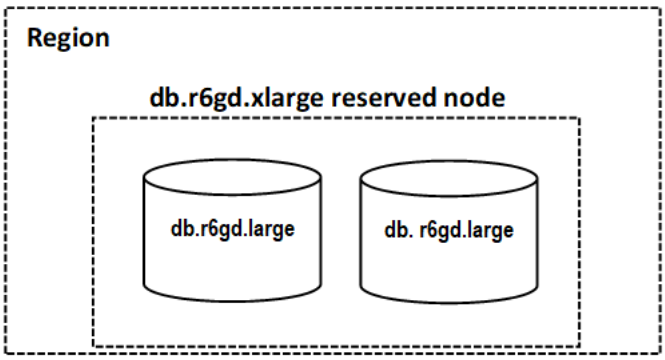
Se você tiver um nó e precisar escalá-lo para uma capacidade maior, o nó reservado será automaticamente aplicado ao nó escalado. Ou seja, seus nós reservados são automaticamente aplicados ao uso de qualquer tamanho na mesma família de nós. Os nós reservados de tamanho flexível estão disponíveis para nós com a mesma região. AWS Nós reservados de tamanho flexível só podem reduzir a escala horizontalmente em suas famílias de nós. Por exemplo, um nó reservado para um db.r6g.xlarge pode ser aplicado a um db.r6g.2xlarge, mas não a um db.r6gd.large, porque db.r6g e db.r6gd são famílias de nós diferentes.

Flexibilidade de tamanho significa que você pode se mover livremente entre configurações dentro da mesma família de nós. Por exemplo, você pode passar de um nó reservado r6g.xlarge (8 unidades normalizadas) para dois nós reservados r6g.large (8 unidades normalizadas) (2*4 = 8 unidades normalizadas) na mesma região sem custo adicional. AWS

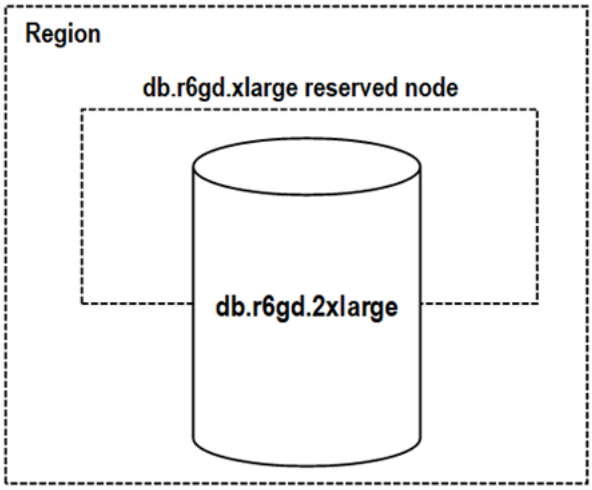
Você pode comparar o uso de diferentes tamanhos de nós reservados usando unidades normalizadas. Por exemplo, uma unidade de uso em dois nós db.r6g.4xlarge é equivalente a 16 unidades normalizadas de uso em um db.r6g.large. A tabela a seguir mostra o número de unidades normalizadas para cada tamanho de nó:

Tamanho do nó	Unidades normalizadas
pequeno	1
médio	2
grande	4
xlarge	8
2xlarge	16
4xlarge	32
6xlarge	48
8xlarge	64
10xlarge	80
12xlarge	96
16xlarge	128

Por exemplo, você compra um nó reservado db.r6gd.xlarge e tem dois nós reservados db.r6gd.large em execução em sua conta na mesma região. AWS Nesse caso, o benefício de faturamento é aplicado integralmente a ambos os nós.



Como alternativa, se você tiver uma instância db.r6gd.2xlarge em execução na sua conta na mesma AWS região, o benefício de cobrança será aplicado a 50% do uso do nó reservado.



Excluir um nó reservado

Os períodos de vigência de um nó reservado envolvem um compromisso de um ou três anos. Você não pode cancelar um nó reservado. No entanto, você pode excluir um nó coberto por um desconto de nó reservado. O processo de exclusão de um nó coberto por um desconto de nó reservado é o mesmo que o de qualquer outro nó.

Se excluir um nó coberto por um desconto de nó reservado, você poderá iniciar outro nó com especificações compatíveis. Neste caso, você continua recebendo a taxa com desconto durante o período de vigência da reserva (um ou três anos).

Trabalhar com nós reservados

Você pode usar a API AWS Management Console AWS Command Line Interface, the e MemoryDB para trabalhar com nós reservados.

Console

Para obter preços e informações sobre as ofertas de nós reservados disponíveis

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, selecione nós reservados.
3. Escolha comprar nós reservados.
4. Em tipo de nó, escolha o tipo de nó que você deseja implantar.
5. Em quantidade, escolha a quantidade de nós que você deseja implantar.
6. Em prazo, escolha quanto tempo você deseja que o nó do banco de dados seja reservado.

7. Em Tipo de oferta, escolha o tipo de oferta.

Após fazer essas seleções, você pode visualizar as informações de preço em Resumo da reserva.

> **Important**
>
> Escolha Cancelar para evitar a compra desses nós e gerar cobranças.

Assim que tiver informações sobre as ofertas de nós reservados disponíveis, você poderá usá-las para comprar uma oferta, conforme mostrado no procedimento a seguir:

Para comprar um nó reservado

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, selecione nós reservados.
3. Escolha comprar nós reservados.
4. Em tipo de nó, escolha o tipo de nó que você deseja implantar.
5. Em quantidade, escolha a quantidade de nós que você deseja implantar.
6. Em prazo, escolha quanto tempo você deseja que o nó do banco de dados seja reservado.
7. Em Tipo de oferta, escolha o tipo de oferta.
8. (Opcional) Você pode atribuir seu próprio identificador aos nós reservados adquiridos, para ajudá-lo a rastreá-los. Em ID da reserva, digite um identificador para o nó reservado.

   Após fazer essas seleções, você pode visualizar as informações de preço em Resumo da reserva.
9. Escolha comprar nós reservados.
10. Seus nós reservados são comprados e exibidos na lista nós reservados.

Para obter informações sobre nós reservados para sua AWS conta

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, selecione nós reservados.

3. Os nós reservados para sua conta são exibidos. Para ver informações detalhadas sobre um nó reservado específico, escolha esse nó na lista. Você pode, então, visualizar informações detalhadas sobre esse nó.

AWS Command Line Interface

O exemplo `describe-reserved-nodes-offerings` a seguir retorna detalhes das ofertas de nós reservados.

```
aws memorydb describe-reserved-nodes-offerings
```

Isso gera uma saída semelhante à seguinte:

```
{
    "ReservedNodesOfferings": [
        {
            "ReservedNodesOfferingId": "0193cc9d-7037-4d49-b332-xxxxxxxxxxxx",
            "NodeType": "db.xxx.large",
            "Duration": 94608000,
            "FixedPrice": $xxx.xx,
            "OfferingType": "Partial Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": $xx.xx,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
        }
    ]
}
```

Você também pode passar os seguintes parâmetros para limitar o escopo do que é retornado:

- `--reserved-nodes-offering-id` – o ID da oferta que você deseja comprar.
- `--node-type` – o valor do filtro do tipo de nó. Use esse parâmetro para mostrar somente as reservas que correspondem ao tipo de nó especificado.
- `--duration` – o valor do filtro de duração, especificado em anos ou segundos. Use esse parâmetro para mostrar somente reservas para esse período.

- `--offering-type` – use esse parâmetro para mostrar somente as ofertas disponíveis que correspondem ao tipo de oferta especificado.

Depois de obter informações sobre as ofertas de nós reservados disponíveis, você pode usar essas informações para comprar uma oferta.

O exemplo `purchase-reserved-nodes-offering` a seguir mostra a compra de novos nós reservados

Para Linux, macOS ou Unix:

```
aws memorydb purchase-reserved-nodes-offering \

    --reserved-nodes-offering-id 0193cc9d-7037-4d49-b332-d5e984f1d8ca \
    --reservation-id reservation \
    --node-count 2
```

Para Windows:

```
aws memorydb purchase-reserved-nodes-offering ^
    --reserved-nodes-offering-id 0193cc9d-7037-4d49-b332-d5e984f1d8ca ^
    --reservation-id MyReservation
```

- `--reserved-nodes-offering-id` representa o nome dos nós reservados oferecidos para compra.
- `--reservation-id` é um identificador especificado pelo cliente para rastrear essa reserva.

> **Note**
>
> O ID da reserva é um identificador exclusivo especificado pelo cliente para rastrear essa reserva. Se esse parâmetro não for especificado, o MemoryDB gerará automaticamente um identificador para a reserva.

- `--node-count` é o número de nós a serem reservados. Ele assume 1 como padrão.

Isso gera uma saída semelhante à seguinte:

```
{
```

```
    "ReservedNode": {
        "ReservationId": "reservation",
        "ReservedNodesOfferingId": "0193cc9d-7037-4d49-b332-xxxxxxxxxxxx",
        "NodeType": "db.xxx.large",
        "StartTime": 1671173133.982,
        "Duration": 94608000,
        "FixedPrice": $xxx.xx,
        "NodeCount": 2,
        "OfferingType": "Partial Upfront",
        "State": "payment-pending",
        "RecurringCharges": [
            {
                "RecurringChargeAmount": $xx.xx,
                "RecurringChargeFrequency": "Hourly"
            }
        ],
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxx:reservednode/reservation"
    }
}
```

Depois de comprar nós reservados, você pode obter informações sobre seus nós reservados.

O exemplo `describe-reserved-nodes` a seguir retorna informações sobre nós reservados para essa conta.

```
aws memorydb describe-reserved-nodes
```

Isso gera uma saída semelhante à seguinte:

```
{
    "ReservedNodes": [
        {
            "ReservationId": "ri-2022-12-16-00-28-40-600",
            "ReservedNodesOfferingId": "0193cc9d-7037-4d49-b332-xxxxxxxxxxxx",
            "NodeType": "db.xxx.large",
            "StartTime": 1671150737.969,
            "Duration": 94608000,
            "FixedPrice": $xxx.xx,
            "NodeCount": 1,
            "OfferingType": "Partial Upfront",
            "State": "active",
```

```
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": $xx.xx,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxx:reservednode/
ri-2022-12-16-00-28-40-600"
        }
    ]
}
```

Você também pode passar os seguintes parâmetros para limitar o escopo do que é retornado:

- `--reservation-id` – você pode atribuir seu próprio identificador aos nós reservados adquiridos, para ajudá-lo a rastreá-los.
- `--reserved-nodes-offering-id` – o valor do filtro identificador da oferta. Use esse parâmetro para mostrar somente as reservas compradas que correspondam ao identificador de oferta especificado.
- `--node-type` – o valor do filtro do tipo de nó. Use esse parâmetro para mostrar somente as reservas que correspondem ao tipo de nó especificado.
- `--duration` – o valor do filtro de duração, especificado em anos ou segundos. Use esse parâmetro para mostrar somente reservas para esse período.
- `--offering-type` – use esse parâmetro para mostrar somente as ofertas disponíveis que correspondem ao tipo de oferta especificado.

API do MemoryDB

Os exemplos a seguir demonstram como usar o MemoryDB Query API para nós reservados:

DescribeReservedNodesOfferings

Retorna detalhes das ofertas de nós reservados.

```
https://memorydb.us-west-2.amazonaws.com/
    ?Action=DescribeReservedNodesOfferings
    &ReservedNodesOfferingId=649fd0c8-xxxx-xxxx-xxxx-06xxxx75e95f
 &"Duration": 94608000,
    &NodeType="db.r6g.large"
```

```
    &OfferingType="Partial Upfront"
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Os parâmetros a seguir limitam o escopo do que é retornado:

- `ReservedNodesOfferingId` representa o nome dos nós reservados oferecidos para compra.
- `Duration` – o valor do filtro de duração, especificado em anos ou segundos. Use esse parâmetro para mostrar somente reservas para esse período.
- `NodeType` – o valor do filtro do tipo de nó. Use esse parâmetro para mostrar somente as ofertas que correspondem ao tipo de nó especificado.
- `OfferingType` – use esse parâmetro para mostrar somente as ofertas disponíveis que correspondem ao tipo de oferta especificado.

Depois de obter informações sobre as ofertas de nós reservados disponíveis, você pode usar essas informações para comprar uma oferta.

PurchaseReservedNodesOffering

Permite que você compre uma oferta de nó reservado.

```
https://memorydb.us-west-2.amazonaws.com/
    ?Action=PurchaseReservedCacheNodesOffering
    &ReservedNodesOfferingId=649fd0c8-xxxx-xxxx-xxxx-06xxxx75e95f
    &ReservationID=myreservationID
    &NodeCount=1
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
```

```
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

- `ReservedNodesOfferingId` representa o nome dos nós reservados oferecidos para compra.
- `ReservationID` é um identificador especificado pelo cliente para rastrear essa reserva.

> **Note**
>
> O ID da reserva é um identificador exclusivo especificado pelo cliente para rastrear essa reserva. Se esse parâmetro não for especificado, o MemoryDB gerará automaticamente um identificador para a reserva.

- `NodeCount` é o número de nós a serem reservados. Ele assume 1 como padrão.

Depois de comprar nós reservados, você pode obter informações sobre seus nós reservados.

DescribeReservedNodes

Retorna informações sobre nós reservados para essa conta.

```
https://memorydb.us-west-2.amazonaws.com/
 ?Action=DescribeReservedNodes
 &ReservedNodesOfferingId=649fd0c8-xxxx-xxxx-xxxx-06xxxx75e95f
 &ReservationID=myreservationID
 &NodeType="db.r6g.large"
 &Duration=94608000
 &OfferingType="Partial Upfront"
 &Version=2021-01-01
 &SignatureVersion=4
 &SignatureMethod=HmacSHA256
 &Timestamp=20141201T220302Z
 &X-Amz-Algorithm
 &X-Amz-SignedHeaders=Host
 &X-Amz-Expires=20141201T220302Z
 &X-Amz-Credential=<credential>
 &X-Amz-Signature=<signature>
```

Os parâmetros a seguir limitam o escopo do que é retornado:

- `ReservedNodesOfferingId` representa o nome do nó reservado.

- `ReservationID` – você pode atribuir seu próprio identificador aos nós reservados adquiridos, para ajudá-lo a rastreá-los.
- `NodeType` – o valor do filtro do tipo de nó. Use esse parâmetro para mostrar somente as reservas que correspondem ao tipo de nó especificado.
- `Duration` – o valor do filtro de duração, especificado em anos ou segundos. Use esse parâmetro para mostrar somente reservas para esse período.
- `OfferingType` – use esse parâmetro para mostrar somente as ofertas disponíveis que correspondem ao tipo de oferta especificado.

Visualização do faturamento de seus nós reservados

É possível visualizar o faturamento dos seus nós reservados no Painel de cobrança no AWS Management Console.

Para visualizar o faturamento de nós reservados

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No botão Pesquisar na parte superior do console, escolha Faturamento.
3. Escolha Faturas no lado esquerdo do painel.
4. Em Taxas de serviço da AWS , expanda o MemoryDB.
5. Expanda a AWS região onde estão seus nós reservados, por exemplo, Leste dos EUA (Norte da Virgínia).

Seus nós reservados e suas cobranças por hora do mês atual são mostrados em Instâncias CreateCluster reservadas do Amazon MemoryDB.

Amazon MemoryDB CreateCluster Reserved Instances		
AmazonMemoryDB, db.r6g.large reserved instance applied	81.000 Hrs	
AmazonMemoryDB, db.r6g.4xlarge reserved instance applied	324.000 Hrs	
AmazonMemoryDB, db.r6g.4xlarge reserved instance applied	162.000 Hrs	
USD hourly fee per AmazonMemoryDB, db.r6g.large instance	1,488.000 Hrs	
USD hourly fee per AmazonMemoryDB, db.r6gd.2xlarge instance	744.000 Hrs	
USD hourly fee per AmazonMemoryDB, db.r6g.4xlarge instance	744.000 Hrs	
USD hourly fee per AmazonMemoryDB, db.r6gd.xlarge instance	744.000 Hrs	
USD hourly fee per AmazonMemoryDB, db.r6gd.4xlarge instance	2,976.000 Hrs	

Substituição de nós

O MemoryDB atualiza frequentemente sua frota com patches e upgrades, geralmente sem interrupções. No entanto, de tempos em tempos, precisamos reiniciar seus nós do MemoryDB para

aplicar atualizações obrigatórias do sistema operacional ao host subjacente. Essas substituições são necessárias para aplicar atualizações que fortalecem a segurança, a confiabilidade e o desempenho operacional.

Você tem a opção de gerenciar essas substituições a qualquer momento antes da janela agendada para a substituição do nó. Ao gerenciar uma substituição sozinho, sua instância recebe a atualização do sistema operacional quando você executa novamente o nó e a substituição de nó agendada é cancelada. Você pode continuar recebendo alertas que indicam que a substituição do nó ocorrerá. Caso já tenha atenuado manualmente a necessidade da manutenção, você pode ignorar esses alertas.

> **Note**
>
> Os nós de substituição gerados automaticamente pelo MemoryDB para Redis podem ter endereços IP diferentes. Você é responsável por revisar a configuração do aplicativo para garantir que os nós estejam associados aos endereços IP apropriados.

A lista a seguir identifica as ações que você pode tomar quando o MemoryDB programar um de seus nós para substituição:

Opções de substituição de nós do MemoryDB

- Não fazer nada – Se você não fizer nada, o MemoryDB substituirá o nó conforme programado.

  Se o nó for membro de um cluster Multi-AZ, o MemoryDB oferece maior disponibilidade durante a aplicação de patches, atualizações e outras substituições de nós relacionadas à manutenção.

  A substituição é concluída enquanto o cluster atende às solicitações de gravação recebidas.
- Mudar sua janela de manutenção – Para eventos de manutenção programados, você recebe um e-mail ou um evento de notificação do MemoryDB. Nesses casos, se você mudar sua janela de manutenção antes da hora de substituição programada, o nó será substituído no novo horário. Para ter mais informações, consulte Modificar um cluster do MemoryDB.

  > **Note**
  >
  > A possibilidade de alterar sua janela de substituição movendo a janela de manutenção só está disponível quando a notificação do MemoryDB inclui uma janela de manutenção.

Se a notificação não inclui uma janela de manutenção, não é possível alterar a janela de substituição.

Por exemplo, digamos que seja quinta-feira, 9 de novembro, às 15h e a próxima janela de manutenção seja sexta-feira, 10 de novembro, às 17h. Veja estes três cenários e seus resultados:

- Você altera sua janela de manutenção para sexta-feira, 16h (após a data e hora atual e antes da próxima janela de manutenção programada). O nó é substituído na sexta-feira, 10 de novembro, às 16h.
- Você altera sua janela de manutenção para sábado, 16h (após a data e hora atual e a próxima janela de manutenção programada). O nó é substituído no sábado, 11 de novembro, às 16h.
- Você altera sua janela de manutenção para quarta-feira às 16:00, mais cedo na semana do que a data e a hora atuais. O nó é substituído na próxima quarta-feira, 15 de novembro, às 16h.

Para obter instruções, consulte Gerenciamento da manutenção.

Gerenciamento de clusters

A maioria das operações do MemoryDB é realizada no nível do cluster. Você pode configurar um cluster com um número específico de nós e um parameter group que controla as propriedades de cada nó. Todos os nós de um cluster são do mesmo tipo e têm as mesmas configurações de parameter group e security group.

Cada cluster deve ter um identificador de cluster. O identificador de cluster é um nome fornecido pelo cliente para o cluster. Esse identificador especifica um cluster específico ao interagir com os comandos da API do MemoryDB e da AWS CLI. O identificador de cluster deve ser exclusivo para esse cliente em uma região da AWS.

Os clusters do MemoryDB foram criados para serem acessados usando uma instância do Amazon EC2. Você só pode iniciar o cluster do MemoryDB em uma nuvem privada virtual (VPC) com base no serviço Amazon VPC, mas pode acessá-lo de fora de AWS. Para obter mais informações, consulte Acessar recursos do MemoryDB de fora da AWS.

Classificação de dados em níveis

Os clusters que usam um tipo de nó da família r6gd têm seus dados classificados em níveis entre a memória e o armazenamento local em unidades de estado sólido (Solid state Drives, SSD). A classificação de dados em níveis fornece uma nova opção de performance de preço para workloads do Redis com SSDs de menor custo em cada nó de cluster, além do datastore na memória. Semelhante a outros tipos de nós, os dados gravados nos nós r6gd são armazenados de forma durável em um log de transações Multi-AZ. A classificação de dados em níveis é ideal para workloads que acessam regularmente até 20% do conjunto de dados geral e para aplicações que podem tolerar latência adicional ao acessar dados em SSD.

Em clusters com classificação de dados em níveis, o MemoryDB monitora o último horário de acesso de cada item armazenado. Quando a memória disponível (DRAM) é totalmente consumida, o MemoryDB usa um algoritmo usado menos recentemente (Least-Recently Used, LRU) para mover automaticamente da memória para o SSD os itens acessados com pouca frequência. Quando os dados em SSD são acessados posteriormente, o MemoryDB os move de modo automático e assíncrono de volta para a memória antes de processar a solicitação. Se você tiver uma workload que acessa regularmente apenas um subconjunto de dados, a classificação de dados em níveis é uma maneira ideal de dimensionar sua capacidade de modo econômico.

Observe que, ao usar a classificação por níveis, as próprias chaves sempre permanecem na memória, enquanto a LRU controla a colocação de valores na memória versus disco. Em geral, recomendamos que seus tamanhos de chave sejam menores do que seus tamanhos de valor ao usar a classificação por níveis de dados.

A classificação de dados em níveis foi projetada para causar impacto mínimo na performance das workload da aplicação. Por exemplo, supondo valores de string de 500 bytes, você pode esperar um adicional de 450 microssegundos de latência para solicitações de leitura de dados armazenados em SSD em comparação com solicitações de leitura de dados na memória.

Com o maior tamanho de nó de classificação de dados (db.r6gd.8xlarge), é possível armazenar até mais ou menos 500 TB em um único cluster de 500 nós (250 TB ao usar 1 réplica de leitura). Para a classificação de dados em níveis, o MemoryDB reserva 19% da memória (DRAM) por nó para uso não relacionado a dados. A classificação de dados em níveis é compatível com todos os comandos e estruturas de dados do Redis compatíveis com o MemoryDB. Para usar esse recurso, não é necessário promover alterações no lado do cliente.

Tópicos

Práticas recomendadas

Recomendamos seguir estas práticas recomendadas:

- A classificação de dados em níveis é ideal para workloads que acessam regularmente até 20% do conjunto de dados geral e para aplicações que podem tolerar latência adicional ao acessar dados em SSD.
- Ao usar a capacidade SSD disponível em nós em níveis de dados, recomendamos que o tamanho do valor seja maior do que o tamanho da chave. O tamanho do valor não pode ser maior que 128 MB, caso contrário, não será movido para o disco. Quando os itens são movidos entre DRAM e SSD, as chaves sempre permanecerão na memória e somente os valores serão movidos para a camada SSD.

Limitações

A classificação de dados em níveis tem as seguintes limitações:

- O tipo de nó usado deve ser da família r6gd, que está disponível nas seguintes regiões: `us-east-2`, `us-east-1`, `us-west-2`, `us-west-1`, `eu-west-1`, `eu-west-3`, `eu-central-1`, `ap-northeast-1`, `ap-southeast-1`, `ap-southeast-2`, `ap-south-1`, `ca-central-1` e `sa-east-1`.
- Não é possível restaurar um snapshot de um cluster r6gd em outro cluster, a menos que ele também use r6gd.
- Não é possível exportar um snapshot para o Amazon S3 para clusters de classificação de dados em níveis.
- Não há compatibilidade com salvamento sem bifurcação.

- Não há compatibilidade com escalabilidade de um cluster de classificação de dados em níveis (p. ex., um cluster que use um tipo de nó r6gd) para um cluster sem classificação de dados em níveis (p. ex., um cluster que use um tipo de nó r6g).
- A classificação de dados em níveis só é compatível com as políticas maxmemory `volatile-lru`, `allkeys-lru` e `noeviction`.
- Itens maiores que 128 MiB não são movidos para o SSD.

Preços para a classificação de dados em níveis

Os nós R6gd têm 5 vezes mais capacidade total (memória + SSD) e podem ajudá-lo a obter mais de 60% de economia de custos de armazenamento ao serem executados na utilização máxima em comparação com os nós R6g (somente memória). Para obter mais informações, consulte Preços do MemoryDB.

Monitorar

O MemoryDB oferece métricas especificamente projetadas para monitorar os clusters de desempenho que usam a classificação de dados em níveis. Para monitorar a proporção de itens na DRAM em comparação com o SSD, é possível usar a métrica de `CurrItems` em Métricas para MemoryDB. Você pode calcular a porcentagem como: `(CurrItems with Dimension: Tier = Memory * 100) / (CurrItems with no dimension filter)`. Quando a porcentagem de itens na memória cair abaixo de 5%, recomendamos que você considere Escalabilidade de clusters do MemoryDB.

Para mais informações, consulte Métricas para clusters do MemoryDB que usam classificação de dados em níveis em Métricas para MemoryDB.

Como usar a classificação de dados em níveis

Como usar a classificação de dados em níveis usando o AWS Management Console

Ao criar um cluster, você usa a classificação de dados em níveis selecionando um tipo de nó da família r6gd, como o db.r6gd.xlarge. A seleção desse tipo de nó ativa automaticamente a classificação de dados em níveis.

Para mais informações sobre como criar um cluster, consulte Etapa 1: criar um cluster.

Como habilitar a classificação de dados em níveis usando a AWS CLI

Ao criar um cluster usando o AWS CLI, você usa a classificação de dados em níveis selecionando um tipo de nó da família r6gd, como db.r6gd.xlarge, e configurando o parâmetro `--data-tiering`.

Você não pode optar por não usar a classificação de dados em níveis ao selecionar um tipo de nó da família r6gd. Se você configurar o parâmetro `--no-data-tiering`, a operação falhará.

Para Linux, macOS ou Unix:

```
aws memorydb create-cluster \
   --cluster-name my-cluster \
   --node-type db.r6gd.xlarge \
   --acl-name my-acl \
   --subnet-group my-sg \
   --data-tiering
```

Para Windows:

```
aws memorydb create-cluster ^
   --cluster-name my-cluster ^
   --node-type db.r6gd.xlarge ^
   --acl-name my-acl ^
   --subnet-group my-sg
   --data-tiering
```

Após executar essa operação, você verá uma resposta semelhante ao seguinte:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "creating",
        "NumberOfShards": 1,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Port": 6379
        },
        "NodeType": "db.r6gd.xlarge",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
```

```
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxxxxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "ACLName": "my-acl",
        "DataTiering":"true",
        "AutoMinorVersionUpgrade": true
    }
}
```

Restauração de dados de um snapshot em clusters com a classificação de dados em níveis ativada

Você pode restaurar um snapshot em um novo cluster com a classificação de dados em níveis ativada usando o (Console), a (AWS CLI) ou a (API do MemoryDB). Ao criar um cluster usando tipos de nós na família r6gd, a classificação de dados em níveis é ativada.

Restauração de dados do snapshot para clusters com a classificação de dados em níveis ativada (console)

Restaurar um snapshot para um novo cluster com a classificação de dados em níveis ativada (console), siga as etapas em Restauração a partir de um snapshot (Console)

Observe que, para ativar a classificação de dados em níveis, você precisa selecionar um tipo de nó da família r6gd.

Restauração de dados do snapshot para clusters com a classificação de dados em níveis ativada (AWS CLI)

Ao criar um cluster usando a AWS CLI, a classificação de dados em níveis é usada por padrão ao selecionar um tipo de nó da família r6gd, por exemplo, o db.r6gd.xlarge, e configurando o parâmetro `--data-tiering`.

Você não pode optar por não usar a classificação de dados em níveis ao selecionar um tipo de nó da família r6gd. Se você configurar o parâmetro `--no-data-tiering`, a operação falhará.

Para Linux, macOS ou Unix:

```
aws memorydb create-cluster \
```

```
   --cluster-name my-cluster \
   --node-type db.r6gd.xlarge \
   --acl-name my-acl \
   --subnet-group my-sg \
   --data-tiering \
   --snapshot-name my-snapshot
```

Para Linux, macOS ou Unix:

```
aws memorydb create-cluster ^
   --cluster-name my-cluster ^
   --node-type db.r6gd.xlarge ^
   --acl-name my-acl ^
   --subnet-group my-sg ^
   --data-tiering ^
   --snapshot-name my-snapshot
```

Após executar essa operação, você verá uma resposta semelhante ao seguinte:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "creating",
        "NumberOfShards": 1,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Port": 6379
        },
        "NodeType": "db.r6gd.xlarge",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxxxxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "ACLName": "my-acl",
        "DataTiering": "true"
}
```

Preparação de um cluster

Veja a seguir instruções sobre como criar um cluster usando o console do MemoryDB, a AWS CLI ou a API do MemoryDB.

Sempre que você criar um cluster, é uma boa ideia fazer algum trabalho preparatório para que você não precise atualizar nem fazer alterações imediatamente.

Tópicos

Determinação dos seus requisitos

Preparação

Conhecer as respostas às seguintes perguntas ajuda a tornar a criação do cluster mais simples:

- Verifique se criou um grupo de sub-rede na mesma VPC antes de começar a criar um cluster. Como alternativa, você pode usar o grupo de sub-rede padrão fornecido. Para obter mais informações, consulte Sub-redes e grupos de sub-redes.

  O MemoryDB foi projetado para ser acessado de dentro da AWS usando o Amazon EC2. No entanto, ao iniciá-lo em uma VPC com base na Amazon VPC, você pode fornecer acesso de fora da AWS. Para obter mais informações, consulte Acessar recursos do MemoryDB de fora da AWS.
- Você precisa personalizar qualquer valor de parâmetro?

  Se você fizer isso, crie um grupo de parâmetro personalizado. Para obter mais informações, consulte Criar um parameter group.
- Você precisa criar um grupo de segurança de VPC?

  Para obter mais informações, consulte Segurança na sua VPC.
- Como você pretende implementar a tolerância a falhas?

  Para obter mais informações, consulte Atenuar falhas.

Tópicos

- Configuração do cluster do MemoryDB
- Multiplexação de E/S aprimorada
- Requisitos de escalabilidade
- Requisitos de acesso
- Regiões e zonas de disponibilidade

Requisitos de memória e processador

O bloco de construção básico do MemoryDB para Redis é o nó. Os nós são configurados em fragmentos para formar clusters. Ao determinar o tipo de nó a ser usado para o seu cluster, considere a configuração do nó do cluster e a quantidade de dados que você deve armazenar.

Configuração do cluster do MemoryDB

Os clusters do MemoryDB são compostos de 1 a 500 fragmentos. Os dados em um cluster do MemoryDB são particionados nos fragmentos no cluster. Seu aplicativo conecta-se a um cluster do MemoryDB usando um endereço de rede chamado de Endpoint. Além dos pontos de extremidade do nó, o cluster do MemoryDB em si tem um endpoint chamado cluster endpoint. Seu aplicativo pode usar esse endpoint para ler ou gravar no cluster, deixando a determinação de qual nó deve ser lido ou gravado a cargo do MemoryDB.

Multiplexação de E/S aprimorada

Se você estiver executando o Redis versão 7.0 ou superior, você obterá aceleração adicional com multiplexação de E/S aprimorada, em que cada thread de E/S de rede dedicado canaliza comandos de vários clientes para o mecanismo Redis, aproveitando a capacidade da Redis de processar comandos em lotes com eficiência. Para obter mais informações, consulte Desempenho ultrarrápido e the section called “Tipos de nó compatíveis”.

Requisitos de escalabilidade

Todos os clusters podem aumentar a escala verticalmente para um tipo de nó maior. Ao aumentar a escala verticalmente de um cluster do MemoryDB, você pode fazer isso on-line para que o cluster permaneça disponível ou você pode semear um novo cluster a partir de um snapshot e evitar que o novo cluster comece vazio.

Para obter mais informações, consulte Escalabilidade neste guia.

Requisitos de acesso

Por design, os clusters do MemoryDB são acessados a partir de instâncias do Amazon EC2. O acesso via rede a um cluster do MemoryDB é limitado à conta que criou esse cluster. Portanto, antes de poder acessar um cluster de uma instância do Amazon EC2, você deve autorizar a acessar o cluster. Para obter instruções detalhadas, consulte Etapa 2: autorizar o acesso ao cluster neste guia.

Regiões e zonas de disponibilidade

Ao colocar seus clusters do MemoryDB em uma região da AWS próxima ao seu aplicativo, é possível reduzir a latência. Se o seu cluster tiver vários nós, a localização deles em diferentes zonas de disponibilidade poderá reduzir o impacto das falhas no cluster.

Para ver mais informações, consulte:

- Escolher regiões e zonas de disponibilidade
- Atenuar falhas

Criar um cluster

O MemoryDB para Redis oferece três maneiras de criar um cluster. Para obter mais informações, consulte Etapa 1: criar um cluster.

Visualização dos detalhes de um cluster

Você pode visualizar informações detalhadas sobre um ou mais clusters usando o console do MemoryDB, a AWS CLI ou a API do MemoryDB.

Visualização de detalhes de um cluster do MemoryDB (console)

O procedimento a seguir detalha como visualizar os detalhes de um cluster do MemoryDB usando o console do MemoryDB.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver os detalhes de um cluster, escolha o botão de opção à esquerda do nome do cluster e escolha Exibir detalhes. Você também pode clicar diretamente no cluster para ver a página de detalhes do cluster.

   A página de detalhes do cluster exibe detalhes sobre o cluster, incluindo o endpoint do cluster. Você pode ver mais detalhes usando as várias guias disponíveis na página de detalhes do cluster.
3. Selecione a guia fragmentos e nós para ver uma lista dos fragmentos do cluster e o número de nós em cada fragmento.
4. Para visualizar informações específicas sobre um nó, expanda o fragmento na tabela abaixo. Como alternativa, você também pode pesquisar o fragmento usando a caixa de pesquisa.

   Isso exibe informações sobre cada nó, incluindo sua zona de disponibilidade, slots/keyspaces e status.
5. Escolha a guia Métricas para monitorar seus respectivos processos, como a utilização da CPU e a utilização da CPU do mecanismo. Para obter mais informações, consulte Métricas para MemoryDB.
6. Escolha a guia Rede e segurança para ver detalhes do grupo de sub-rede e dos grupos de segurança.

   a. Em Grupo de sub-redes, você pode ver o nome do grupo de sub-redes, um link para a VPC à qual a sub-rede pertence e o nome do recurso da Amazon (ARN) do grupo de sub-redes.

   b. Em Grupos de segurança, você pode ver o ID, o nome e a descrição do grupo de segurança.

7. Escolha a guia Manutenção e snapshot para ver detalhes das configurações do snapshot.

   a. Em Snapshot, você pode ver se os snapshots automatizados estão ativados, o período de retenção do snapshot e a janela do snapshot.

   b. Em Snapshots, você verá uma lista de todos os snapshots desse cluster, incluindo o nome, o tamanho, o número de fragmentos e o status do snapshot.

   Para obter mais informações, consulte Snapshots e restauração .

8. Escolha a guia Manutenção e snapshot para ver os detalhes da janela de manutenção, junto com quaisquer atualizações pendentes de ACL, refragmentação ou serviço. Para obter mais informações, consulte Gerenciamento da manutenção.

9. Escolha a guia Atualizações de serviços para ver detalhes de todas as atualizações de serviço aplicáveis a esse cluster. Para obter mais informações, consulte Atualizações de serviço no MemoryDB para Redis.

10. Escolha a guia Tags para ver detalhes de quaisquer tags de alocação de recursos ou custos associados a esse cluster. Para obter mais informações, consulte Marcação de snapshots.

Visualização dos detalhes de um cluster (CLI da AWS)

Você pode visualizar os detalhes de um cluster usando o comando AWS CLI `describe-clusters`. Se o parâmetro `--cluster-name` for omitido, os detalhes para vários clusters, até `--max-results`, serão retornados. Se o parâmetro `--cluster-name` estiver incluído, os detalhes do cluster especificado serão retornados. Você pode limitar o número de registros retornados com o parâmetro `--max-results`.

O código a seguir lista os detalhes para `my-cluster`.

```
aws memorydb describe-clusters --cluster-name my-cluster
```

O código a seguir lista os detalhes para até 25 clusters.

```
aws memorydb describe-clusters --max-results 25
```

Example

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
    --cluster-name my-cluster \
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster ^
    --show-shard-details
```

A saída JSON a seguir mostra a resposta:

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Description": "my cluster",
            "Status": "available",
            "NumberOfShards": 1,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": 1629230643.961,
                            "Endpoint": {
                                "Address": "my-cluster-0001-001.my-
cluster.abcdef.memorydb.us-east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "CreateTime": 1629230644.025,
                            "Endpoint": {
```

```
                                "Address": "my-cluster-0001-002.my-
cluster.abcdef.memorydb.us-east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.abcdef.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "default",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:000000000:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "sat:06:30-sat:07:30",
            "SnapshotWindow": "04:00-05:00",
            "ACLName": "open-access",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true,
        }
```

Para obter mais informações, consulte a AWS CLI para o tópico do MemoryDB `describe-clusters`.

Visualizar os detalhes de um cluster: (API do MemoryDB)

Você pode visualizar os detalhes de um cluster usando a ação `DescribeClusters` da API do MemoryDB. Se o parâmetro `ClusterName` estiver incluído, os detalhes do cluster especificado serão retornados. Se o parâmetro `ClusterName` for omitido, os detalhes para até `MaxResults` (padrão 100) clusters serão retornados. O valor para `MaxResults` não pode ser inferior a 20 ou superior a 100.

O código a seguir lista os detalhes para `my-cluster`.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeClusters
   &ClusterName=my-cluster
   &Version=2021-01-01
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

O código a seguir lista os detalhes para até 25 clusters.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeClusters
   &MaxResults=25
   &Version=2021-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte o tópico `DescribeClusters` de referência da API do MemoryDB.

Modificar um cluster do MemoryDB

Além de adicionar ou remover nós de um cluster, pode haver momentos em que você precisará fazer outras alterações em um cluster existente, como adicionar um grupo de segurança, alterar a janela de manutenção ou um grupo de parâmetros.

Recomendamos que você tenha sua janela de manutenção cair no momento da menor utilização. Assim, talvez seja necessário modificá-la de tempos em tempos.

Quando você altera os parâmetros de um cluster, a alteração é aplicada ao cluster imediatamente. Isso é verdadeiro se você alterar o próprio grupo de parâmetro do cluster ou um valor do parâmetro dentro do grupo do parâmetro do cluster.

Você também pode atualizar a versão do mecanismo de seus clusters. Por exemplo, você pode selecionar uma nova versão secundária do mecanismo e o MemoryDB começará a atualizar seu cluster imediatamente.

Usar a AWS Management Console

Como modificar um cluster

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista localizada no canto superior direito, escolha a região da AWS em que o cluster que você deseja modificar está localizado.
3. No painel de navegação à esquerda, acesse Clusters. Em Detalhes dos clusters, selecione o cluster usando o botão de opções e vá até Ações e depois Modificar.
4. A página Modificar é exibida.
5. Na janela Modificar, faça as modificações desejadas. Entre as opções estão:

    - Descrição
    - Grupos de sub-rede
    - Grupos de segurança da VPC
    - Tipo de nó

> **Note**
>
> Se o cluster estiver usando um tipo de nó da família r6gd, você só poderá escolher um tamanho de nó diferente nessa família. Se você escolher um tipo de nó da família r6gd, a classificação de dados em níveis será ativada automaticamente. Para obter mais informações, consulte Classificação de dados em níveis.

- Compatibilidade da versão do Redis
- Habilitar snapshots automáticos
- Período de retenção de snapshot
- Janela do Snapshot
- Janela de manutenção
- Tópico para notificação do SNS

6. Escolha Salvar alterações.

Você também pode acessar a página de detalhes do cluster e clicar em modificar para fazer modificações no cluster. Se você quiser modificar seções específicas do cluster, acesse a respectiva guia na página de detalhes do cluster e clique em Modificar.

Usar a AWS CLI

Você pode modificar um cluster existente usando a operação AWS CLI `update-cluster`. Para modificar o valor de configuração de um cluster, especifique o ID do cluster, o parâmetro a ser alterado e o novo valor do parâmetro. O exemplo a seguir altera a janela de manutenção para um cluster chamado `my-cluster` e aplica a alteração imediatamente.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --preferred-maintenance-window sun:23:00-mon:02:00
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
```

```
    --preferred-maintenance-window sun:23:00-mon:02:00
```

Para obter mais informações, consulte update-cluster na referência de comandos da AWS CLI.

Usando a API do MemoryDB

Você pode modificar um cluster existente usando a operação UpdateCluster na API do MemoryDB. Para modificar o valor de configuração de um cluster, especifique o ID do cluster, o parâmetro a ser alterado e o novo valor do parâmetro. O exemplo a seguir altera a janela de manutenção para um cluster chamado `my-cluster` e aplica a alteração imediatamente.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateCluster
    &ClusterName=my-cluster
    &PreferredMaintenanceWindow=sun:23:00-mon:02:00
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210802T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Adição e Remoção de nós de um cluster

Você pode adicionar ou remover nós de um cluster usando o AWS Management Console, AWS CLI ou a API do MemoryDB.

Usar a AWS Management Console

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista de clusters, escolha o nome do cluster do qual você deseja remover um nó.
3. Na guia fragmentos e nós, escolha Adicionar/Excluir nós
4. em Número de nós, insira o número de nós desejado.
5. Selecione a opção Confirmar.

> **Important**
>
> Se você definir o número de nós como 1, não estará mais habilitado para Multi-AZ. Você também pode optar por ativar o failover automático.

Usar a AWS CLI

1. Identifique os nomes dos nós que você deseja remover. Para obter mais informações, consulte Visualização dos detalhes de um cluster.
2. Use a operação `update-cluster` da CLI com uma lista dos nós a serem removidos, como no exemplo a seguir.

   Para remover nós de um cluster usando a interface da linha de comando, use o comando `update-cluster` com os seguintes parâmetros:

   - `--cluster-name` o ID do cluster de cache do qual você deseja remover nós.
   - `--replica-configuration` – permite que você defina o número de réplicas:
     - `ReplicaCount` – defina essa propriedade para especificar o número de nós de réplica desejado.
   - `--region` especifica a região da AWS do cluster da qual você deseja remover nós.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --replica-configuration \
        ReplicaCount=1 \
    --region us-east-1
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --replica-configuration ^
        ReplicaCount=1 ^
    --region us-east-1
```

Para obter mais informações, consulte os tópicos da AWS CLI `update-cluster`.

Usando a API do MemoryDB

Para remover nós usando a API do MemoryDB, chame a operação `UpdateCluster` da API com o nome do cluster e uma lista de nós para remoção, conforme mostrado:

- `ClusterName` o ID do cluster de cache do qual você deseja remover nós.
- `ReplicaConfiguration` – permite que você defina o número de réplicas:
  - `ReplicaCount` – defina essa propriedade para especificar o número de nós de réplica desejado.
- `Region`: especifica a região da AWS do cluster da qual você deseja remover um nó.

Para obter mais informações, consulte UpdateCluster.

Acessar o cluster

Suas instâncias do MemoryDB para Redis são projetadas para acesso por meio de uma instância do Amazon EC2.

Você pode acessar seu nó do MemoryDB de uma instância do Amazon EC2 na mesma Amazon VPC. Ou, usando o emparelhamento da VPC, você pode acessar seu nó do MemoryDB de um Amazon EC2 em uma Amazon VPC diferente.

Tópicos

Conceder acesso a seus clusters

Você pode se conectar ao seu cluster do MemoryDB somente a partir de uma instância do Amazon EC2 que esteja sendo executada na mesma Amazon VPC. Nesse caso, você precisará conceder entrada de rede ao cluster.

Para conceder entrada na rede de um grupo de segurança da Amazon VPC para um cluster

1. Faça login no AWS Management Console e abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. No painel de navegação esquerdo, em Network & Security, escolha Security Groups.
3. Na lista de grupos de segurança, escolha o de segurança para a sua Amazon VPC. A menos que você tenha criado um grupo de segurança para uso com o MemoryDB, esse grupo de segurança será chamado default.
4. Escolha a guia Inbound e faça o seguinte:
    a. Escolha Editar.
    b. Escolha Adicionar regra.
    c. Na coluna Type, escolha Custom TCP rule.
    d. Na caixa Port range, digite o número da porta para o nó do cluster. Esse número deve ser o mesmo que você especificou quando você executou o cluster. A porta padrão para o Redis é **6379**.

e. Na caixa Fonte, escolha Qualquer lugar, que tem o intervalo de porta (0.0.0.0/0) para que qualquer instância do Amazon EC2 que você inicie na sua Amazon VPC possa se conectar aos seus nós do MemoryDB.

> **Important**
>
> Abrir o cluster do MemoryDB para 0.0.0.0/0 não expõe o cluster à Internet, pois ele não possui um endereço IP público e, portanto, não pode ser acessado de fora da VPC. No entanto, o grupo de segurança padrão pode ser aplicado a outras instâncias do Amazon EC2 na conta do cliente, e essas instâncias podem ter um endereço IP público. Se eles estiverem executando algo na porta padrão, esse serviço poderá ser exposto involuntariamente. Portanto, recomendamos criar um grupo de segurança de VPC que será usado exclusivamente pelo MemoryDB. Para obter mais informações, consulte Grupos de segurança personalizados.

f. Escolha Salvar.

Quando você ativa uma instância do Amazon EC2 na sua Amazon VPC, essa instância poderá se conectar ao seu cluster do MemoryDB.

Acessar recursos do MemoryDB de fora da AWS

MemoryDB é um serviço projetado para ser usado internamente em sua VPC. O acesso externo não é recomendado devido à latência do tráfego da Internet e preocupações de segurança. No entanto, se o acesso externo ao MemoryDB for necessário para fins de teste ou desenvolvimento, poderá ser feito por meio de uma VPN.

Usando o cliente VPN da AWS, você permite o acesso externo aos seus nós do MemoryDB com os seguintes benefícios:

- Acesso restrito a usuários aprovados ou chaves de autenticação;
- Tráfego criptografado entre o cliente de VPN e o endpoint da VPN da AWS;
- Acesso limitado a sub-redes ou nós específicos;
- Fácil revogação do acesso de usuários ou chaves de autenticação;
- Conexões de auditoria;

Os procedimentos a seguir demonstram como:

Tópicos

Criar uma autoridade de certificação

É possível criar uma Autoridade de certificação (CA) usando diferentes técnicas ou ferramentas. Sugerimos o utilitário easy-rsa, fornecido pelo projeto OpenVPN. Independentemente da opção escolhida, mantenha as chaves seguras. O procedimento a seguir faz download dos scripts easy-rsa, cria a Autoridade de certificação e as chaves para autenticar o primeiro cliente de VPN:

- Para criar os certificados iniciais, abra um terminal e faça o seguinte:
  - `git clone` https://github.com/OpenVPN/easy-rsa
  - `cd easy-rsa`
  - `./easyrsa3/easyrsa init-pki`
  - `./easyrsa3/easyrsa build-ca nopass`

- `./easyrsa3/easyrsa build-server-full server nopass`
- `./easyrsa3/easyrsa build-client-full client1.domain.tld nopass`

Um subdiretório pki com os certificados será criado sob easy-rsa.

- Envie o certificado do servidor para o AWS Certificate Manager (ACM):
  - No console do ACM, selecione Gerenciador de certificados.
  - Selecione Importar certificado.
  - Informe o certificado de chave pública disponível no arquivo `easy-rsa/pki/issued/server.crt` no campo Corpo do certificado.
  - Cole a chave privada disponível no `easy-rsa/pki/private/server.key` no campo Chave privada do certificado. Selecione todas as linhas entre `BEGIN AND END PRIVATE KEY` (incluindo as linhas `BEGIN` e `END`).
  - Cole a chave pública da CA disponível no arquivo `easy-rsa/pki/ca.crt` no campo Cadeia de certificados.
  - Selecione Revisar e importar.
  - Selecione Importar.

  Para enviar os certificados do servidor ao ACM usando a CLI da AWS, execute o seguinte comando: `aws acm import-certificate --certificate fileb://easy-rsa/pki/issued/server.crt --private-key file://easy-rsa/pki/private/server.key --certificate-chain file://easy-rsa/pki/ca.crt --region region`

  Anote o ARN do certificado para uso futuro.

Configuração de componentes do cliente VPN da AWS

Usar o AWS Console

No console da AWS, selecione Serviços e, depois, VPC.

Em Rede privada virtual (VPN), selecione Endpoints do Client VPN e faça o seguinte:

Configuração de componentes do cliente VPN da AWS

- Selecione Criar endpoint do Client VPN.
- Especifique as seguintes opções:

- CIDR de IPv4 de cliente: use uma rede privada com uma máscara de rede de pelo menos intervalo /22. Verifique se a sub-rede selecionada não entra em conflito com os endereços das redes da VPC. Exemplo: 10.0.0.0/22.
- Em ARN do certificado de servidor, selecione o ARN do certificado importado anteriormente.
- Selecione Usar autenticação mútua.
- Em ARN do certificado de cliente, selecione o ARN do certificado importado anteriormente.
- Selecione Criar endpoint do Client VPN.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 create-client-vpn-endpoint --client-cidr-block
"10.0.0.0/22" --server-certificate-arn arn:aws:acm:us-
east-1:012345678912:certificate/0123abcd-ab12-01a0-123a-123456abcdef --
authentication-options Type=certificate-
authentication,,MutualAuthentication={ClientRootCertificateChainArn=arn:aws:acm:
east-1:012345678912:certificate/123abcd-ab12-01a0-123a-123456abcdef} --
connection-log-options Enabled=false
```

Exemplos de resultado:

```
"ClientVpnEndpointId": "cvpn-endpoint-0123456789abcdefg",
"Status": { "Code": "pending-associate" }, "DnsName": "cvpn-
endpoint-0123456789abcdefg.prod.clientvpn.us-east-1.amazonaws.com" }
```

Associar as redes de destino ao endpoint de VPN

- Selecione o novo endpoint de VPN e, depois, selecione a guia Associações.
- Selecione Associar e especifique as opções a seguir.
  - VPC: selecione a VPC do cluster do MemoryDB.
  - Selecione uma das redes do cluster do MemoryDB. Em caso de dúvida, revise as redes nos Grupos de sub-redes no painel do MemoryDB.
  - Selecione Associar. Se necessário, repita as etapas para as redes restantes.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 associate-client-vpn-target-network --client-vpn-endpoint-id cvpn-
endpoint-0123456789abcdefg --subnet-id subnet-0123456789abdcdef
```

Exemplos de resultado:

```
"Status": { "Code": "associating" }, "AssociationId": "cvpn-
assoc-0123456789abdcdef" }
```

Analisar o grupo de segurança de VPN

O endpoint de VPN adotará automaticamente o grupo de segurança padrão da VPC. Verifique as regras de entrada e saída e confirme se o grupo de segurança permite o tráfego da rede VPN (definido nas configurações de endpoint de VPN) para as redes do MemoryDB nas portas de serviço (por padrão, 6379 para Redis).

Se você precisar alterar o grupo de segurança atribuído ao endpoint de VPN, faça o seguinte:

- Selecione o grupo de segurança atual.
- Selecione Apply Security Group (Aplicar grupo de segurança).
- Selecione o novo grupo de segurança.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 apply-security-groups-to-client-vpn-target-network --
client-vpn-endpoint-id cvpn-endpoint-0123456789abcdefga  --vpc-id
vpc-0123456789abdcdef --security-group-ids sg-0123456789abdcdef
```

Exemplos de resultado:

```
"SecurityGroupIds": [ "sg-0123456789abdcdef" ] }
```

> **Note**
>
> O grupo de segurança do MemoryDB também precisa permitir o tráfego proveniente dos clientes de VPN. Os endereços dos clientes serão mascarados com o endereço do endpoint

de VPN, de acordo com a rede VPC. Portanto, considere a rede VPC (não a rede dos clientes de VPN) ao criar a regra de entrada no grupo de segurança do MemoryDB.

Autorizar o acesso de VPN às redes de destino

Na guia Autorização, selecione Autorizar entrada e especifique o seguinte:

- Rede de destino para habilitar o acesso: use 0.0.0.0/0 para permitir o acesso a qualquer rede (incluindo a Internet) ou restringir as redes/hosts do MemoryDB.
- Em Conceder acesso a:, selecione Permitir acesso a todos os usuários.
- Selecione Adicionar regras de autorização.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 authorize-client-vpn-ingress --client-vpn-endpoint-id cvpn-endpoint-0123456789abcdefg --target-network-cidr 0.0.0.0/0 --authorize-all-groups
```

Exemplos de resultado:

```
{ "Status": { "Code": "authorizing" } }
```

Permitir o acesso à Internet dos clientes de VPN

Se você precisar navegar na Internet por meio da VPN, será necessário criar uma rota adicional. Selecione a guia Tabela de rotas e, depois, selecione Criar rota:

- Destino da rota: 0.0.0.0/0
- ID de sub-rede da VPC de destino: selecione uma das sub-redes associadas com acesso à Internet.
- Selecione Criar rota.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 create-client-vpn-route --client-vpn-endpoint-id cvpn-
endpoint-0123456789abcdefg --destination-cidr-block 0.0.0.0/0 --target-vpc-
subnet-id subnet-0123456789abdcdef
```

Exemplos de resultado:

```
{ "Status": { "Code": "creating" } }
```

Configurar o cliente de VPN

No painel do cliente VPN da AWS, selecione o endpoint de VPN criado recentemente e selecione Baixar configuração do cliente. Copie o arquivo de configuração e os arquivos `easy-rsa/pki/issued/client1.domain.tld.crt` e `easy-rsa/pki/private/client1.domain.tld.key`. Edite o arquivo de configuração e altere ou adicione os seguintes parâmetros:

- cert: adicione uma nova linha com o parâmetro cert apontando para o arquivo `client1.domain.tld.crt`. Use o caminho completo para o arquivo. Exemplo: `cert /home/user/.cert/client1.domain.tld.crt`
- cert: key: adicione uma nova linha com a chave de parâmetro apontando para o arquivo `client1.domain.tld.key`. Use o caminho completo para o arquivo. Exemplo: `key /home/user/.cert/client1.domain.tld.key`

Estabeleça a conexão VPN com o comando: `sudo openvpn --config downloaded-client-config.ovpn`

Revogar acesso

Se você precisar invalidar o acesso de uma chave de cliente específica, a chave precisará ser revogada na CA. Depois, envie a lista de revogação para o cliente VPN da AWS.

Revogar a chave com easy-rsa:

- `cd easy-rsa`
- `./easyrsa3/easyrsa revoke client1.domain.tld`
- Digite "sim" para continuar ou qualquer outra entrada para cancelar.

  ```
  Continue with revocation: `yes` ... * `./easyrsa3/easyrsa gen-crl
  ```
- Uma CRL atualizada foi criada. Arquivo de CRL: `/home/user/easy-rsa/pki/crl.pem`

Importação da lista de revogação para o cliente VPN da AWS:

- No AWS Management Console, selecione SServiços e, depois, VPC.
- Selecione Endpoints do Client VPN.
- Selecione o endpoint do Client VPN e, depois, selecione Ações -> IImportar CRL de certificado de cliente.
- Cole o conteúdo do arquivo `crl.pem`.

Como usar a AWS CLI

Execute o seguinte comando :

```
aws ec2 import-client-vpn-client-certificate-revocation-list --certificate-revocation-list file://./easy-rsa/pki/crl.pem --client-vpn-endpoint-id cvpn-endpoint-0123456789abcdefg
```

Exemplos de resultado:

```
Example output: { "Return": true }
```

Encontrar endpoints de conexão

Seu aplicativo conecta-se ao seu cluster usando endpoints. Um endpoint é o endereço exclusivo de um nó ou cluster. Use o endpoint de cluster do cluster para todas as operações.

As seções a seguir o guiarão na descoberta do endpoint necessário.

Localização do endpoint para um cluster do MemoryDB (AWS Management Console)

Para encontrar os endpoints de um cluster do MemoryDB

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, escolha Clusters.

   A tela de clusters será exibida com uma lista de clusters. Escolha o cluster ao qual você deseja se conectar.
3. Para encontrar o endpoint do cluster, escolha o nome do cluster (não o botão de opção).
4. O endpoint do cluster é exibido em detalhes do cluster. Para copiá-lo, selecione o ícone copiar à esquerda do endpoint.

Localização do endpoint de um cluster do MemoryDB (AWS CLI)

Você pode usar o comando `describe-clusters` para descobrir o endpoint de um cluster. O comando retorna o endpoint do cluster.

A operação a seguir recupera o endpoint, que neste exemplo é representado como uma *`amostra`*, para o cluster `mycluster`.

Retorna a seguinte resposta em JSON:

```
aws memorydb describe-clusters \
  --cluster-name mycluster
```

Para Windows:

```
aws memorydb describe-clusters ^
   --cluster-name mycluster
```

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 1,
```

```
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.4",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:zzzexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```

Para obter mais informações, consulte describe-clusters.

Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Você pode usar a API do MemoryDB para Redis para descobrir o endpoint de um cluster.

Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Você pode usar a API do MemoryDB para descobrir o endpoint de um cluster com a ação `DescribeClusters`. A ação retorna o endpoint do cluster.

A operação a seguir recupera o endpoint do cluster. `mycluster`

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeClusters
    &ClusterName=mycluster
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210802T192317Z
    &Version=2021-01-01
    &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte DescribeClusters.

Operação com fragmentos

Um fragmento é uma coleção de um a 6 nós. É possível criar um cluster com alto número de fragmentos e baixo número de réplicas totalizando até 500 nós por cluster. Essa configuração do cluster pode variar de 500 fragmentos e 0 réplicas para 100 fragmentos e 4 réplicas, que é o número máximo de réplicas permitidas. Os dados do cluster são particionados entre todos os fragmentos do cluster. Se houver mais de um nó em um fragmento, este implementará a replicação com um nó sendo o nó primário de leitura/gravação e os outros nós como nós de réplica somente leitura.

Ao criar um cluster do MemoryDB usando o AWS Management Console, você especifica o número de fragmentos no cluster e o número de nós nos fragmentos. Para obter mais informações, consulte Criação de um cluster do MemoryDB.

Cada nó em um fragmento tem as mesmas especificações de computação, armazenamento e memória. A API do MemoryDB permite que você controle os atributos de todo o cluster, como o número de nós, as configurações de segurança e as janelas de manutenção do sistema.

Para obter mais informações, consulte Refragmentação offline e rebalanceamento de fragmentos do MemoryDB e Refragmentação online e rebalanceamento de fragmentos do MemoryDB.

Localização do nome de um fragmento

Você pode encontrar o nome de um fragmento usando o AWS Management Console, a AWS CLI ou a API do MemoryDB.

Usar a AWS Management Console

O procedimento a seguir usa o AWS Management Console para encontrar os nomes dos fragmentos de um cluster do MemoryDB.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Clusters.
3. Escolha o cluster em Nome cujos nomes de fragmentos você deseja encontrar.
4. Na guia Fragmentos e nós, visualize a lista de fragmentos em Nome. Você também pode expandir cada um para ver detalhes de seus nós.

Usar a AWS CLI

Para encontrar nomes de fragmentos (fragmentos) para clusters do MemoryDB, use a operação AWS CLI da `describe-clusters` com o seguinte parâmetro opcional.

- **`--cluster-name`**: um parâmetro opcional que, quando usado, limita a saída aos detalhes do cluster especificado. Se esse parâmetro for omitido, serão retornados os detalhes de até 100 clusters.
- **`--show-shard-details`**: retorna detalhes dos fragmentos, incluindo seus nomes.

Esse comando retorna os detalhes do `my-cluster`.

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
    --cluster-name my-cluster
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster
    --show-shard-details
```

Retorna a seguinte resposta em JSON:

As quebras de linha foram adicionadas para legibilidade.

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 1,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
```

```
                    ],
                    "NumberOfNodes": 2
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```


Usando a API do MemoryDB

Para localizar ids de fragmentos para clusters do MemoryDB, use a operação `DescribeClusters` da API com o seguinte parâmetro opcional.

- **`ClusterName`**: um parâmetro opcional que, quando usado, limita a saída aos detalhes do cluster especificado. Se esse parâmetro for omitido, serão retornados os detalhes de até 100 clusters.
- **`ShowShardDetails`**: retorna detalhes dos fragmentos, incluindo seus nomes.

Example

Esse comando retorna os detalhes do `my-cluster`.

Para Linux, macOS ou Unix:

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeClusters
   &ClusterName=sample-cluster
   &ShowShardDetails=true
   &Version=2021-01-01
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

Gerenciando sua implementação do MemoryDB

Nesta seção, você pode encontrar detalhes sobre como gerenciar os vários componentes da sua implantação do MemoryDB.

Tópicos

Versões do mecanismo Redis

Esta seção aborda as versões compatíveis do mecanismo Redis.

Tópicos

MemoryDB para Redis versão 7.1 (aprimorado)

A versão 7.1 do MemoryDB for Redis adiciona suporte para recursos de pesquisa vetorial em pré-visualização para regiões selecionadas, bem como correções de bugs críticos e aprimoramentos de desempenho.

- Recurso de pesquisa vetorial: a pesquisa vetorial pode ser usada com a funcionalidade existente do MemoryDB. Os aplicativos que não usam a pesquisa vetorial não serão afetados por sua presença. A visualização prévia da pesquisa vetorial está disponível no MemoryDB para Redis, versão 7.1 em diante, nas seguintes regiões: Leste dos EUA (Norte da Virgínia e Ohio), Oeste dos EUA (Oregon), UE (Irlanda) e Ásia-Pacífico (Tóquio). Consulte a documentação aqui para saber como ativar a visualização da pesquisa vetorial e os recursos relacionados.

> **Note**
>
> O MemoryDB for Redis versão 7.1 é compatível com o OSS Redis v7.0. Para obter mais informações sobre a versão do Redis 7.0, consulte as notas de lançamento do Redis 7.0 em Redis on. GitHub

MemoryDB para Redis versão 7.0 (aprimorado)

O MemoryDB para Redis 7.0 adiciona várias melhorias e suporte para novas funcionalidades:

- Funções do Redis: o MemoryDB para Redis 7 adiciona suporte às funções do Redis e fornece uma experiência gerenciada que permite que os desenvolvedores executem scripts LUA com a lógica da aplicação armazenada no cluster do MemoryDB, sem exigir que os clientes reenviem os scripts para o servidor com cada conexão.
- Melhorias na ACL: o MemoryDB para Redis 7 adiciona suporte para a próxima versão das listas de controle de acesso (ACLs) do Redis. Com o MemoryDB para Redis 7, os clientes agora podem especificar vários conjuntos de permissões em chaves ou espaços de chave específicos no Redis.
- Pub/Sub fragmentado: o MemoryDB para Redis 7 adiciona suporte para executar a funcionalidade Redis Pub/Sub de forma fragmentada ao executar o MemoryDB no modo de cluster habilitado (CME). Os recursos do Redis Pub/Sub Redis permitem que os editores enviem mensagens para qualquer número de assinantes em um canal. Com o Amazon MemoryDB para Redis 7, os canais são vinculados a um fragmento no cluster do MemoryDB, eliminando a necessidade de propagar as informações do canal entre os fragmentos. Isso resulta em melhor escalabilidade.
- Multiplexação de E/S aprimorada: o MemoryDB para Redis versão 7 apresenta a multiplexação de E/S aprimorada, que proporciona maior throughput e menor latência para workloads de alto throughput com muitos clientes conectados simultaneamente a um cluster do MemoryDB. Por exemplo, ao usar um cluster de nós r6g.4xlarge e executar 5.200 clientes simultâneos, você

pode obter até 46% de aumento no throughput (operações de leitura e gravação por segundo) e redução de até 21% na latência P99, em comparação com o MemoryDB para Redis versão 6.

Para obter mais informações sobre a versão do Redis 7.0, consulte as notas de lançamento do Redis 7.0 em Redis on. GitHub

MemoryDB para Redis versão 6.2 (aprimorado)

O MemoryDB apresenta a próxima versão do mecanismo Redis, que inclui Autenticação de usuários com listas de controle de acesso (ACLs), suporte à atualização automática de versão, cache do lado do cliente e melhorias operacionais significativas.

A versão 6.2.6 do mecanismo Redis também introduz suporte ao formato nativo de notação de JavaScript objeto (JSON), uma maneira simples e sem esquemas de codificar conjuntos de dados complexos dentro de clusters do Redis. Com o suporte ao JSON, é possível aproveitar a performance e as APIs Redis para aplicações que operam em JSON. Para ter mais informações, consulte Conceitos básicos do JSON. Também está incluída a métrica relacionada ao JSON `JsonBasedCmds` que é incorporada CloudWatch para monitorar o uso desse tipo de dados. Para ter mais informações, consulte Métricas para MemoryDB.

A partir do Redis 6, o MemoryDB oferecerá uma única versão para cada versão secundária do Redis OSS, em vez de oferecer várias versões de patch. Isso foi projetado para minimizar a confusão e a ambiguidade de ter que escolher entre várias versões secundárias. O MemoryDB também gerenciará automaticamente a versão secundária e a versão de correção de seus clusters em execução, garantindo melhor desempenho e segurança aprimorada. Isso será tratado por meio de canais padrão de notificação ao cliente por meio de uma campanha de atualização de serviço. Para ter mais informações, consulte Atualizações de serviço no MemoryDB para Redis.

Se você não especificar a versão do mecanismo durante a criação, o MemoryDB selecionará automaticamente a versão preferida do Redis para você. Por outro lado, se você especificar a versão do mecanismo usando `6.2`, o MemoryDB invocará automaticamente a versão de patch preferida do Redis 6.2 que estiver disponível.

Por exemplo, quando você criar um cluster, você definirá o parâmetro `--engine-version` como `6.2`. O cluster será iniciado com a versão de patch preferencial atual disponível no momento da criação. Qualquer solicitação com um valor de versão completa do mecanismo será rejeitada, uma exceção será lançada e o processo falhará.

Ao chamar a API `DescribeEngineVersions`, o valor do parâmetro `EngineVersion` será definido como 6.2 e a versão real completa do mecanismo será retornada no campo `EnginePatchVersion`.

Para obter mais informações sobre a versão 6.2 do Redis, consulte as notas de lançamento do Redis 6.2 em Redis on. GitHub

Atualização de versões de mecanismos

Por padrão, o MemoryDB gerencia automaticamente a versão do patch de seus clusters em execução por meio de atualizações de serviço. Além disso, você pode desativar a atualização automática de versões secundárias se definir a propriedade `AutoMinorVersionUpgrade` dos seus clusters como “false”. No entanto, você não pode cancelar a atualização automática da versão do patch.

Você pode controlar se e quando os softwares compatíveis com o protocolo que alimenta seu cluster são atualizados para novas versões com suporte pelo MemoryDB antes do início do upgrade automático. Esse nível de controle permite que você mantenha a compatibilidade com versões específicas, teste novas versões com seu aplicativo antes de implantar em produção e realize atualizações de versão em seus próprios termos e cronogramas.

Você pode iniciar os upgrades da versão do mecanismo em seu cluster das seguintes maneiras:

- Ao atualizá-lo e especificar uma nova versão do mecanismo. Para ter mais informações, consulte Modificar um cluster do MemoryDB.
- Aplicando a atualização do serviço para a versão do mecanismo correspondente. Para ter mais informações, consulte Atualizações de serviço no MemoryDB para Redis.

Observe o seguinte:

- Você pode atualizar para uma versão de mecanismo mais recente, mas não pode fazer downgrade para uma versão de mecanismo mais antiga. Se quiser usar uma versão de mecanismo mais antiga, você deverá excluir o cluster existente e criá-lo novamente com a versão mais antiga do mecanismo.
- Recomendamos atualizar periodicamente para a versão principal mais recente, já que a maioria das melhorias principais não são transferidas para versões mais antigas. À medida que o MemoryDB expande a disponibilidade para uma nova AWS região, o MemoryDB oferece suporte às duas `MAJOR.MINOR` versões mais recentes da época para a nova região. Por exemplo, se uma nova AWS região for iniciada e as versões mais recentes do `MAJOR.MINOR` MemoryDB para

Redis forem 7.0 e 6.2, o MemoryDB for Redis suportará as versões 7.0 e 6.2 na nova região. AWS À medida que novas versões `MAJOR.MINOR` do MemoryDB para Redis forem lançadas, o MemoryDB adicionará suporte às versões recém-lançadas do MemoryDB para Redis. Para saber mais sobre como escolher regiões para o MemoryDB, consulte Regiões e endpoints com suporte.

- O gerenciamento da versão do mecanismo foi desenvolvido para que você possa ter o máximo controle possível sobre a execução de patches. No entanto, o MemoryDB reserva o direito de executar patches no cluster em seu nome caso ocorra uma vulnerabilidade de segurança crítica no sistema ou software.
- O MemoryDB oferecerá uma única versão para cada versão secundária do Redis OSS, em vez de oferecer várias versões de patch. Isso foi projetado para minimizar a confusão e a ambiguidade de ter que escolher entre várias versões. O MemoryDB também gerenciará automaticamente a versão secundária e a versão de correção de seus clusters em execução, garantindo melhor desempenho e segurança aprimorada. Isso será tratado por meio de canais padrão de notificação ao cliente por meio de uma campanha de atualização de serviço. Para ter mais informações, consulte Atualizações de serviço no MemoryDB para Redis.
- Você pode atualizar a versão do cluster com o mínimo de tempo de inatividade. O cluster estará disponível para leituras durante todo o processo de atualização e para gravações durante a maior parte da atualização, exceto durante a operação de failover que dura alguns segundos.
- Recomendamos que você faça atualizações do mecanismo durante períodos de baixo tráfego de gravação de entrada.

  Os clusters com vários fragmentos são processados e corrigidos da seguinte forma:

  - Apenas uma operação de upgrade é realizada por fragmento a qualquer momento.
  - Em cada fragmento, todas as réplicas são processadas antes do processamento da primária. Caso haja menos réplicas em um fragmento, a primária nesse fragmento pode ser processada antes da conclusão do processamento das réplicas em outros fragmentos.
  - Em todos os fragmentos, os nós primários são processados em série. Somente um nó primário é atualizado por vez.

Tópicos

- Como atualizar as versões dos mecanismos
- Resolução de atualizações do mecanismo do Redis bloqueadas

Como atualizar as versões dos mecanismos

Você inicia as atualizações de versão do seu cluster modificando-o usando o console MemoryDB, o ou a API MemoryDB e AWS CLI especificando uma versão mais recente do mecanismo. Para obter mais informações, consulte os tópicos a seguir.

- Usar a AWS Management Console
- Usar a AWS CLI
- Usando a API do MemoryDB

Resolução de atualizações do mecanismo do Redis bloqueadas

Conforme mostrado na tabela a seguir, sua operação de atualização do mecanismo Redis será bloqueada se você tiver uma operação de expansão pendente.

Operações pendentes	Operações bloqueadas
Amplie a sua capacidade	Atualização imediata do mecanismo
Atualização do mecanismo	Expansão imediata
Expansão e atualização do mecanismo	Expansão imediata
	Atualização imediata do mecanismo

Conceitos básicos do JSON

O MemoryDB é compatível com o formato nativo JavaScript Object Notation (JSON), uma maneira simples e sem esquema de codificar conjuntos de dados complexos dentro de clusters Redis. É possível armazenar e acessar dados nativamente usando o formato JSON em clusters do Redis e atualizar dados JSON armazenados nesses clusters sem precisar gerenciar código personalizado para serializá-los e desserializá-los.

Além de aproveitar as APIs do Redis para aplicativos que operam com JSON, agora você pode recuperar e atualizar com eficiência partes específicas de um documento JSON sem precisar manipular o objeto inteiro, o que pode melhorar o desempenho e reduzir os custos. Também

é possível pesquisar o conteúdo do seu documento JSON usando a consulta `JSONPath` estilo Goessner.

Depois de criar um cluster com uma versão de mecanismo compatível, o tipo de dados do JSON e os comandos associados estarão disponíveis automaticamente. Isso é compatível com a API e o RDB com a versão 2 do módulo RedisJSON, para que você possa migrar facilmente aplicações Redis baseadas em JSON para o MemoryDB. Para obter mais informações sobre os comandos do Redis compatíveis, consulte Comandos compatíveis.

As métricas `JsonBasedCmds` relacionadas ao JSON são incorporadas ao CloudWatch para monitorar o uso desse tipo de dados. Para obter mais informações consulte Métricas para MemoryDB.

> **Note**
>
> Para usar o JSON, é necessário estar executando o mecanismo do Redis versão 6.2.6 ou posterior.

Tópicos

- Visão geral do tipo de dados JSON do Redis
- Comandos compatíveis

Visão geral do tipo de dados JSON do Redis

O MemoryDB é compatível com vários comandos do Redis para trabalhar com o tipo de dados do JSON. A seguir uma visão geral do tipo de dados do JSON e uma lista detalhada dos comandos do Redis que são compatíveis.

Terminologia

Prazo	Descrição
Documento JSON	Refere-se ao valor de uma chave JSON do Redis
Valor JSON	refere-se a um subconjunto de um documento JSON, incluindo a raiz que representa o

Prazo	Descrição
	documento inteiro. Um valor poderia ser um contêiner ou uma entrada dentro de um contêiner
Elemento JSON	Equivalente ao valor JSON

Padrão compatível com JSON

O formato JSON é compatível com os padrão de intercâmbio de dados do JSON RFC 7159 e ECMA-404. O padrão UTF-8 Unicode é compatível com texto do JSON.

Elemento raiz

O elemento raiz pode ser de qualquer tipo de dados do JSON. Observe que na RFC 4627 anterior, somente objetos ou matrizes eram permitidos como valores raiz. Desde a atualização para o RFC 7159, a raiz de um documento JSON pode ser de qualquer tipo de dados do JSON.

Limite de tamanho de documentos

Os documentos JSON são armazenados internamente em um formato que é otimizado para acesso e modificação rápidos. Esse formato normalmente resulta no consumo um pouco maior de memória do que a representação serializada equivalente do mesmo documento. O consumo de memória por um único documento JSON é limitado a 64MB, que é o tamanho da estrutura de dados na memória, não a string JSON. A quantidade de memória consumida por um documento JSON pode ser inspecionada usando o comando `JSON.DEBUG MEMORY`.

ACLs JSON

- O tipo de dados do JSON é totalmente integrado à capacidade da lista de controle de acesso (ACL) do Redis. Semelhante às categorias existentes por tipo de dados (@string, @hash etc.), uma nova categoria @json foi adicionada para simplificar o gerenciamento do acesso a comandos e dados do JSON. Nenhum outro comando Redis existente é membro da categoria @json. Todos os comandos JSON impõem restrições e permissões de keyspace ou de comando.
- Existem cinco categorias existentes do Redis ACL que são atualizadas para incluir os novos comandos JSON: @read, @write, @fast, @slow e @admin. A tabela abaixo indica o mapeamento de comandos JSON para as categorias apropriadas.

ACL

Comando JSON	@read	@write	@fast	@slow	@admin
JSON.ARRAPPEND		y	y		
JSON.ARRINDEX	y		y		
JSON.ARRINSERT		y	y		
JSON.ARRLEN	y		y		
JSON.ARRPOP		y	y		
JSON.ARRTRIM		y	y		
JSON.CLEAR		y	y		
JSON.DEBUG	y			y	y
JSON.DEL		y	y		
JSON.FORGET		y	y		
JSON.GET	y		y		
JSON.MGET	y		y		
JSON.NUMINCRBY		y	y		

Comando JSON	@read	@write	@fast	@slow	@admin
JSON.NUMMULTBY		y	y		
JSON.OBJKEYS	y		y		
JSON.OBJLEN	y		y		
JSON.RESP	y		y		
JSON.SET		y		y	
JSON.STRAPPEND		y	y		
JSON.STRLEN	y		y		
JSON.STRLEN	y		y		
JSON.TOGGLE		y	y		
JSON.TYPE	y		y		
JSON.NUMINCRBY		y	y		

Limite de profundidade de aninhamento

Quando um objeto ou matriz JSON tem um elemento que é outro objeto ou matriz JSON, diz-se que esse objeto interno ou matriz se “aninha” dentro do objeto ou matriz externa. O limite máximo de profundidade de aninhamento é 128. Qualquer tentativa de criar um documento que contenha uma profundidade de aninhamento maior que 128 será rejeitada com um erro.

Sintaxe de comando

A maioria dos comandos exige um nome de chave Redis como primeiro argumento. Alguns comandos também têm um argumento path (caminho). O argumento path (caminho) será padronizado para a raiz se for opcional e não fornecido.

Notação:

- Os argumentos obrigatórios são colocados entre colchetes angulares, por exemplo <key>
- Os argumentos opcionais são colocados dentro de colchetes, por exemplo [path]
- Argumentos opcionais adicionais são indicados por..., por exemplo, [json...]

Sintaxe de caminho

O JSON-Redis oferece suporte a dois tipos de sintaxes de caminho:

- Sintaxe aprimorada – veja abaixo a sintaxe JSONPath descrita por Goessner, conforme mostrado na tabela abaixo. Reordenamos e modificamos as descrições na tabela para maior clareza.
- Sintaxe restrita - Tem recursos de consulta limitados.

> **Note**
>
> Os resultados de alguns comandos são sensíveis ao tipo de sintaxe de caminho usado.

Se um caminho de consulta começar com '$', ele usará a sintaxe aprimorada. Caso contrário, a sintaxe restrita será usada.

Sintaxe aprimorada

Símbolo/Expressão	Descrição
$	o elemento raiz
. ou []	operador filho
..	descida recursiva

Símbolo/Expressão	Descrição
*	curinga. Todos os elementos em um objeto ou matriz.
[]	operador subscrito de matriz. O índice é baseado em 0.
[,]	operador da união
[start:end:step]	operador de matriz slice
?()	aplica uma expressão de filtro (script) à matriz ou objeto atual
()	expressão de filtro
@	usado em expressões de filtro que consultam o nó atual que está sendo processado
==	igual a, usado em expressões de filtro.
!=	não é igual a, usado em expressões de filtro.
>	maior que, usado em expressões de filtro.
>=	maior que ou igual a, usado em expressões de filtro.
<	menor que, usado em expressões de filtro.
<=	menor que ou igual a, usado em expressões de filtro.
&&	AND lógico, usado para combinar várias expressões de filtro.
||	OR lógico, usado para combinar várias expressões de filtro.

Exemplos

Os exemplos abaixo têm como base o exemplo de dados XML de Goessner, que modificamos acrescentando campos adicionais.

```
{ "store": {
    "book": [
      { "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95,
        "in-stock": true,
        "sold": true
      },
      { "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99,
        "in-stock": false,
        "sold": true
      },
      { "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99,
        "in-stock": true,
        "sold": false
      },
      { "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8",
        "price": 22.99,
        "in-stock": false,
        "sold": false
      }
    ],
    "bicycle": {
      "color": "red",
      "price": 19.95,
      "in-stock": true,
      "sold": false
    }
```

```
  }
}
```

Path	Descrição
$.store.book[*].author	os autores de todos os livros da loja
$..author	todos os autores
$.store.*	todos os membros da loja
$["store"].*	todos os membros da loja
$.store..price	o preço de tudo na loja
$..*	todos os membros recursivos da estrutura JSON
$..book[*]	todos os livros
$..book[0]	o primeiro livro
$..book[-1]	o último livro
$..book[0:2]	os dois primeiros livros
$..book[0,1]	os dois primeiros livros
$..book[0:4]	livros do índice 0 a 3 (o índice final não é inclusivo)
$..book[0:4:2]	livros no índice 0, 2
$..book[?(@.isbn)]	todos os livros com um número ISBN
$..book[?(@.price<10)]	todos os livros com valor inferior a US$ 10
'$..book[?(@.price < 10)]'	todos os livros com valor inferior a US$ 10. (O caminho deverá estar entre aspas se contiver espaços em branco.)

Path	Descrição
'$..book[?(@["price"] < 10)]'	todos os livros com valor inferior a US$ 10
'$..book[?(@.["price"] < 10)]'	todos os livros com valor inferior a US$ 10
$..book[?(@.price>=10&&@.price<=100)]	todos os livros na faixa de preço entre US$ 10 e US$ 100, inclusive
'$..book[?(@.price>=10 && @.price<=100)]'	todos os livros na faixa de preço entre US$ 10 e US$ 100, inclusive. (O caminho deverá estar entre aspas se contiver espaços em branco.)
$..book[?(@.sold==true||@.in-stock==false)]	todos os livros vendidos ou esgotados
'$..book[?(@.sold == true || @.in-stock == false)]'	todos os livros vendidos ou esgotados. (O caminho deverá estar entre aspas se contiver espaços em branco.)
'$.store.book[?(@.["category"] == "fiction")]'	todos os livros na categoria ficção
'$.store.book[?(@.["category"] != "fiction")]'	todos os livros nas categorias não ficção

Mais exemplos de expressões de filtro:

```
127.0.0.1:6379> JSON.SET k1 . '{"books": [{"price":5,"sold":true,"in-
stock":true,"title":"foo"}, {"price":15,"sold":false,"title":"abc"}]}'
OK
127.0.0.1:6379> JSON.GET k1 $.books[?(@.price>1&&@.price<20&&@.in-stock)]
"[{\"price\":5,\"sold\":true,\"in-stock\":true,\"title\":\"foo\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?(@.price>1 && @.price<20 && @.in-stock)]'
"[{\"price\":5,\"sold\":true,\"in-stock\":true,\"title\":\"foo\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?((@.price>1 && @.price<20) && (@.sold==false))]'
"[{\"price\":15,\"sold\":false,\"title\":\"abc\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?(@.title == "abc")]'
[{"price":15,"sold":false,"title":"abc"}]

127.0.0.1:6379> JSON.SET k2 . '[1,2,3,4,5]'
127.0.0.1:6379> JSON.GET k2 $.*.[?(@>2)]
"[3,4,5]"
127.0.0.1:6379> JSON.GET k2 '$.*.[?(@ > 2)]'
```

```
"[3,4,5]"

127.0.0.1:6379> JSON.SET k3 . '[true,false,true,false,null,1,2,3,4]'
OK
127.0.0.1:6379> JSON.GET k3 $.*.[?(@==true)]
"[true,true]"
127.0.0.1:6379> JSON.GET k3 '$.*.[?(@ == true)]'
"[true,true]"
127.0.0.1:6379> JSON.GET k3 $.*.[?(@>1)]
"[2,3,4]"
127.0.0.1:6379> JSON.GET k3 '$.*.[?(@ > 1)]'
"[2,3,4]"
```

Sintaxe restrita

Símbolo/Expressão	Descrição
. ou []	operador filho
[]	operador subscrito de matriz. O índice é baseado em 0.

Exemplos

Path	Descrição
.store.book[0].author	o autor do primeiro livro
.store.book[-1].author	o autor do último livro
.address.city	nome da cidade
["store"]["book"][0]["title"]	o título do primeiro livro
["store"]["book"][-1]["title"]	o título do último livro

> **Note**
>
> Todo conteúdo de Goessner citado nesta documentação está sujeito à Licença da Creative Commons.

Prefixos de erro comuns

Cada mensagem de erro tem um prefixo. Veja a seguir uma lista de prefixos de erro comuns:

Prefixo	Descrição
ERR	um erro geral
LIMIT	erro de limite de tamanho excedido. Por exemplo, o limite de tamanho do documento ou o limite de profundidade de aninhamento excedido
NONEXISTENT	uma chave ou caminho não existe
OUTOFBOUNDARIES	índice de matriz fora dos limites
SYNTAXERR	erro de sintaxe
WRONGTYPE	tipo de valor errado

métricas relacionadas ao JSON

As seguintes métricas de informações JSON são fornecidas:

Informações	Descrição
json_total_memory_bytes	memória total alocada para objetos JSON
json_num_documents	número total de documentos no Redis

Para consultar as métricas principais, execute os comandos do Redis:

```
info json_core_metrics
```

Como o MemoryDB interage com o JSON

O exemplo a seguir ilustra como o MemoryDB interage com o tipo de dados JSON.

Precedência do operador

Ao avaliar expressões condicionais para filtragem, &&s têm precedência primeiro e, em seguida, ||s são avaliadas, como é comum na maioria das linguagens. As operações dentro dos parênteses serão executadas primeiro.

Comportamento do limite máximo de aninhamento de caminho

O limite máximo de aninhamento de caminho do MemoryDB é 128. Por isso, um valor como `$.a.b.c.d...` só pode atingir 128 níveis.

Processamento de valores numéricos

O JSON não tem tipos de dados separados para números inteiros e de ponto flutuante. Todos eles são chamados de números.

Quando um número JSON é recebido, ele é armazenado em um de dois formatos. Se o número couber em um inteiro assinado de 64 bits, será convertido para esse formato; caso contrário, será armazenado como uma string. As operações aritméticas em dois números JSON (por exemplo, JSON.NUMINCRBY e JSON.NUMMULTBY) tentam preservar o máximo de precisão possível. Se os dois operandos e o valor resultante couberem em um inteiro assinado de 64 bits, a aritmética de números inteiros será executada. Caso contrário, os operandos de entrada são convertidos em números de ponto flutuante de precisão dupla IEEE de 64 bits, a operação aritmética é executada e o resultado é convertido novamente em uma string.

Comandos aritméticos `NUMINCRBY` e `NUMMULTBY`:

- Se ambos os números forem inteiros e o resultado estiver fora da faixa de int64, ele se tornará automaticamente um número flutuante de precisão dupla.
- Se pelo menos um dos números for um ponto flutuante, o resultado será um número de ponto flutuante de precisão dupla.
- Se o resultado exceder o intervalo de dupla, o comando retornará um erro `OVERFLOW`.

> **Note**
>
> Antes da versão 6.2.6.R2 do mecanismo Redis, quando um número JSON é recebido na entrada, ele é convertido em uma das duas representações binárias internas: um inteiro assinado de 64 bits ou um ponto flutuante de precisão dupla IEEE de 64 bits. A string original e toda a sua formatação não serão retidas. Dessa forma, quando um número é gerado como parte de uma resposta JSON, ele é convertido da representação binária interna para uma string imprimível que usa regras genéricas de formatação. Essas regras podem resultar em uma string diferente da que foi recebida.
>
> - Se ambos os números forem inteiros e o resultado estiver fora da faixa de `int64`, ele se tornará automaticamente um número IEEE de ponto flutuante de precisão dupla de 64 bits.
> - Se pelo menos um dos números for um ponto flutuante, o resultado será um número IEEE ponto flutuante de precisão dupla de 64 bits.
> - Se o resultado exceder a faixa de 64 bits IEEE dupla, o comando `OVERFLOW` retornará um erro.

Para obter uma lista detalhada dos comandos disponíveis, consulte Comandos compatíveis.

Avaliação estrita da sintaxe

MemoryDB não permite caminhos JSON com sintaxe inválida, mesmo que um subconjunto do caminho contenha um caminho válido. Isso acontece para manter o comportamento correto para nossos clientes.

Comandos compatíveis

Os seguintes comandos JSON do Redis são compatíveis:

Tópicos

- JSON.ARRAPPEND
- JSON.ARRINDEX
- JSON.ARRINSERT
- JSON.ARRLEN
- JSON.ARRPOP
- JSON.ARRTRIM

- JSON.CLEAR
- JSON.DEBUG
- JSON.DEL
- JSON.FORGET
- JSON.GET
- JSON.MGET
- JSON.NUMINCRBY
- JSON.NUMMULTBY
- JSON.OBJLEN
- JSON.OBJKEYS
- JSON.RESP
- JSON.SET
- JSON.STRAPPEND
- JSON.STRLEN
- JSON.TOGGLE
- JSON.TYPE

JSON.ARRAPPEND

Anexa um ou mais valores aos valores da matriz no caminho.

Sintaxe

```
JSON.ARRAPPEND <key> <path> <json> [json ...]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- json (obrigatório) – O valor JSON a ser anexado à matriz

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de números inteiros, representando o novo comprimento da matriz em cada caminho.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.
- Erro `SYNTAXERR` se um dos argumentos de entradas json não for uma string JSON válida.
- `NONEXISTENT` erro se o caminho não existir.

Se o caminho for uma sintaxe restrita:

- Inteiro, o novo comprimento da matriz.
- Se vários valores de matriz forem selecionados, o comando retornará o novo comprimento da última matriz atualizada.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.
- Erro `SYNTAXERR` se um dos argumentos de entradas json não for uma string JSON válida.
- `NONEXISTENT` erro se o caminho não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRAPPEND  k1 $[*] '"c"'
1) (integer) 1
2) (integer) 2
3) (integer) 3
127.0.0.1:6379> JSON.GET k1
"[[\"c\"],[\"a\",\"c\"],[\"a\",\"b\",\"c\"]]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRAPPEND  k1 [-1] '"c"'
(integer) 3
127.0.0.1:6379> JSON.GET k1
"[[],[\"a\"],[\"a\",\"b\",\"c\"]]"
```

JSON.ARRINDEX

Procura a primeira ocorrência de um valor escalar JSON nas matrizes no caminho.

- Erros fora do intervalo são tratados arredondando o índice para o início e o fim da matriz.
- Se início **>** fim, retorna -1 (não encontrado).

Sintaxe

```
JSON.ARRINDEX <key> <path> <json-scalar> [start [end]]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- json-scalar (obrigatório) – valor escalar a ser pesquisado; escalar JSON se refere a valores que não são objetos ou matrizes. Ou seja, String, number, boolean e null são valores escalares.
- início (opcional) – o índice inicial, inclusive. Assumirá o padrão de 0 se não for fornecido.
- fim (opcional) – O índice final, exclusivo. Assumirá o padrão de 0 se não for fornecido, significa que o último elemento está incluído. 0 ou -1 significa que o último elemento está incluído.

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de números inteiros. Cada valor é o índice do elemento correspondente na matriz no caminho. O valor é -1, se não encontrado.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.

Se o caminho for uma sintaxe restrita:

- Inteiro, o índice do elemento correspondente ou -1 se não for encontrado.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRINDEX k1 $[*] '"b"'
1) (integer) -1
2) (integer) -1
3) (integer) 1
4) (integer) 1
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.ARRINDEX k1 .children '"Tom"'
(integer) 2
```

JSON.ARRINSERT

Insere um ou mais valores nos valores da matriz no caminho antes do índice.

Sintaxe

```
JSON.ARRINSERT <key> <path> <index> <json> [json ...]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- índice (obrigatório) – um índice de matriz antes do qual os valores são inseridos.
- json (obrigatório) – O valor JSON a ser anexado à matriz

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de números inteiros, representando o novo comprimento da matriz em cada caminho.
- Se um valor for uma matriz vazia, seu valor de retorno correspondente será nulo.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.
- Erro `OUTOFBOUNDARIES` se o argumento índice estiver fora dos limites.

Se o caminho for uma sintaxe restrita:

- Inteiro, o novo comprimento da matriz.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.
- Erro `OUTOFBOUNDARIES` se o argumento índice estiver fora dos limites.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRINSERT k1 $[*] 0 '"c"'
1) (integer) 1
2) (integer) 2
3) (integer) 3
127.0.0.1:6379> JSON.GET k1
"[[\"c\"],[\"c\",\"a\"],[\"c\",\"a\",\"b\"]]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRINSERT k1 . 0 '"c"'
(integer) 4
127.0.0.1:6379> JSON.GET k1
"[\"c\",[],[\"a\"],[\"a\",\"b\"]]"
```

JSON.ARRLEN

Obtém o comprimento dos valores da matriz no caminho.

Sintaxe

```
JSON.ARRLEN <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de inteiros que representa o comprimento da matriz em cada caminho.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.
- Nulo se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Matriz de strings em massa. Cada elemento é um nome de chave no objeto.
- Inteiro, comprimento da matriz.
- Se vários objetos forem selecionados, o comando retornará o comprimento da primeira matriz.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.
- `WRONGTYPE` erro se o caminho não existir.
- Nulo se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], [\"a\"], [\"a\", \"b\"], [\"a\", \"b\", \"c\"]]'
(error) SYNTAXERR Failed to parse JSON string due to syntax error
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRLEN k1 $[*]
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 3

127.0.0.1:6379> JSON.SET k2 . '[[], "a", ["a", "b"], ["a", "b", "c"], 4]'
OK
127.0.0.1:6379> JSON.ARRLEN k2 $[*]
1) (integer) 0
2) (nil)
3) (integer) 2
4) (integer) 3
5) (nil)
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRLEN k1 [*]
(integer) 0
127.0.0.1:6379> JSON.ARRLEN k1 $[3]
1) (integer) 3

127.0.0.1:6379> JSON.SET k2 . '[[], "a", ["a", "b"], ["a", "b", "c"], 4]'
OK
127.0.0.1:6379> JSON.ARRLEN k2 [*]
(integer) 0
127.0.0.1:6379> JSON.ARRLEN k2 $[1]
1) (nil)
127.0.0.1:6379> JSON.ARRLEN k2 $[2]
1) (integer) 2
```

JSON.ARRPOP

Remove e retorna elemento no índice da matriz. Exibir uma matriz vazia retorna nulo.

Sintaxe

```
JSON.ARRPOP <key> [path [index]]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido
- índice (opcional) – posição na matriz a partir da qual começar a exibir.
  - O padrão é -1 se não é fornecido, o que significa o último elemento.
  - O valor negativo significa posição do último elemento.
  - Os índices fora do limite são arredondados para seus respectivos limites de matriz.

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de strings em massa que representam os valores exibidos em cada caminho.
- Se um valor for uma matriz vazia, seu valor de retorno correspondente será nulo.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.

Se o caminho for uma sintaxe restrita:

- String em massa, representando o valor JSON exibido
- Nulo se a matriz estiver vazia.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k1 $[*]
1) (nil)
2) "\"a\""
3) "\"b\""
127.0.0.1:6379> JSON.GET k1
"[[],[],[\"a\"]]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k1
"[\"a\",\"b\"]"
127.0.0.1:6379> JSON.GET k1
"[[],[\"a\"]]"

127.0.0.1:6379> JSON.SET k2 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k2 . 0
"[]"
127.0.0.1:6379> JSON.GET k2
"[[\"a\"],[\"a\",\"b\"]]"
```

JSON.ARRTRIM

Reduz as matrizes no caminho para que se tornem sub matrizes [start, end], ambas inclusive.

- Se a matriz estiver vazia, não faça nada, retorne 0.
- Se início for **<** 0, trate-a como 0.
- Se fim for **>=** tamanho (tamanho da matriz), trate-a como tamanho-1.
- Se início for **>=** tamanho ou início for **>** fim, esvazie a matriz e retorne 0.

Sintaxe

```
JSON.ARRINSERT <key> <path> <start> <end>
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- início (obrigatório) – índice inicial, inclusive.
- fim (obrigatório) – índice final, inclusive.

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de números inteiros, representando o novo comprimento da matriz em cada caminho.
- Se um valor for uma matriz vazia, seu valor de retorno correspondente será nulo.
- Se um valor não for uma matriz, seu valor de retorno correspondente será nulo.
- Erro `OUTOFBOUNDARIES` se um argumento índice estiver fora dos limites.

Se o caminho for uma sintaxe restrita:

- Inteiro, o novo comprimento da matriz.
- Nulo se a matriz estiver vazia.
- Erro `WRONGTYPE` se o valor no caminho não for uma matriz.
- Erro `OUTOFBOUNDARIES` se um argumento índice estiver fora dos limites.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRTRIM k1 $[*] 0 1
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 2
   127.0.0.1:6379> JSON.GET k1
   "[[],[\"a\"],[\"a\",\"b\"],[\"a\",\"b\"]]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.ARRTRIM k1 .children 0 1
(integer) 2
127.0.0.1:6379> JSON.GET k1 .children
"[\"John\",\"Jack\"]"
```

JSON.CLEAR

Limpa as matrizes ou um objeto no caminho.

Sintaxe

```
JSON.CLEAR <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

- Inteiro, o número de contêineres limpos.
- Limpar uma matriz ou objeto vazio conta como 0 contêiner limpo.

> **Note**
>
> Antes da versão 6.2.6.R2 do Redis, limpar uma matriz ou objeto vazio conta como 1 contêiner limpo.

- Limpar um valor que não seja do contêiner retorna 0.
- Se nenhum valor de matriz ou objeto estiver localizado no caminho, o comando retorna 0.

Exemplos

```
127.0.0.1:6379> JSON.SET k1 . '[[], [0], [0,1], [0,1,2], 1, true, null, "d"]'
OK
127.0.0.1:6379>  JSON.CLEAR k1  $[*]
(integer) 6
127.0.0.1:6379> JSON.CLEAR k1  $[*]
(integer) 0
127.0.0.1:6379> JSON.SET k2 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.CLEAR k2 .children
(integer) 1
127.0.0.1:6379> JSON.GET k2 .children
"[]"
```

JSON.DEBUG

Informações do relatório. Os subcomandos compatíveis são:

- MEMORY <key> [path] – Informa o uso de memória em bytes de um valor JSON. O caminho assumirá o padrão da raiz se não for fornecido.
- DEPTH <key> [caminho] – Informa a profundidade máxima do caminho do documento JSON.

> **Note**
>
> Esse subcomando está disponível somente usando o mecanismo do Redis versão 6.2.6.R2 ou posterior.

- FIELDS <key> [path] – Informa o número de campos no caminho do documento especificado. O caminho assumirá o padrão da raiz se não for fornecido. Cada valor JSON não contêiner conta

como um campo. Objetos e matrizes contam recursivamente como um campo para cada um dos valores JSON que contêm. Cada valor de contêiner, exceto o contêiner raiz, conta como um campo adicional.

- HELP – imprime mensagens de ajuda referentes ao comando.

Sintaxe

```
JSON.DEBUG <subcommand & arguments>
```

Depende do subcomando:

MEMORY

- Se o caminho for uma sintaxe aprimorada:
  - Retorna uma matriz de inteiros, que representa o tamanho da memória (em bytes) do valor JSON em cada caminho.
  - Retorna uma matriz vazia se a chave Redis não existir.
- Se o caminho for uma sintaxe restrita:
  - retorna um número inteiro, o tamanho da memória do valor JSON em bytes.
  - retorna nulo se a chave Redis não existir.

DEPTH

- Retorna um número inteiro que representa a profundidade máxima do caminho do documento JSON.
- Retornará nulo se a chave Redis não existir.

FIELDS

- Se o caminho for uma sintaxe aprimorada:
  - Retorna uma matriz de inteiros, que representa o número de campos do valor JSON em cada caminho.
  - Retorna uma matriz vazia se a chave Redis não existir.
- Se o caminho for uma sintaxe restrita:
  - retorna um número inteiro, número de campos do valor JSON.

- retorna nulo se a chave Redis não existir.

HELP – retorna uma série de mensagens de ajuda.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[1, 2.3, "foo", true, null, {}, [], {"a":1, "b":2},
 [1,2,3]]'
OK
127.0.0.1:6379> JSON.DEBUG MEMORY k1 $[*]
1) (integer) 16
2) (integer) 16
3) (integer) 19
4) (integer) 16
5) (integer) 16
6) (integer) 16
7) (integer) 16
8) (integer) 50
9) (integer) 64
127.0.0.1:6379> JSON.DEBUG FIELDS k1 $[*]
1) (integer) 1
2) (integer) 1
3) (integer) 1
4) (integer) 1
5) (integer) 1
6) (integer) 0
7) (integer) 0
8) (integer) 2
9) (integer) 3
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
```

```
127.0.0.1:6379> JSON.DEBUG MEMORY k1
(integer) 632
127.0.0.1:6379> JSON.DEBUG MEMORY k1 .phoneNumbers
(integer) 166

127.0.0.1:6379> JSON.DEBUG FIELDS k1
(integer) 19
127.0.0.1:6379> JSON.DEBUG FIELDS k1 .address
(integer) 4

127.0.0.1:6379> JSON.DEBUG HELP
1) JSON.DEBUG MEMORY <key> [path] - report memory size (bytes) of the JSON element.
 Path defaults to root if not provided.
2) JSON.DEBUG FIELDS <key> [path] - report number of fields in the JSON element. Path
 defaults to root if not provided.
3) JSON.DEBUG HELP - print help message.
```

JSON.DEL

Exclui os valores JSON no caminho em uma chave de documento. Se o caminho é a raiz, é equivalente a excluir a chave do Redis.

Sintaxe

```
JSON.DEL <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

- Número de elementos excluídos.
- 0 se a chave Redis não existir.
- 0 se o caminho JSON for inválido ou não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.DEL k1 $.d.*
(integer) 3
127.0.0.1:6379> JSOn.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.DEL k1 $.e[*]
(integer) 5
127.0.0.1:6379> JSOn.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[]}"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.DEL k1 .d.*
(integer) 3
127.0.0.1:6379> JSON.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.DEL k1 .e[*]
(integer) 5
127.0.0.1:6379> JSON.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[]}"
```

JSON.FORGET

Um alias de JSON.DEL

JSON.GET

Retorna o JSON serializado em um ou vários caminhos.

Sintaxe

```
JSON.GET <key>
[INDENT indentation-string]
[NEWLINE newline-string]
[SPACE space-string]
```

```
[NOESCAPE]
[path ...]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- INDENT/NEWLINE/SPACE (opcional) – controla o formato da string do JSON retornada, isto é, “formatado para impressão”. O valor padrão de cada um é string vazia. Podem ser anulados em qualquer combinação. Eles podem ser especificados em qualquer ordem.
- NOESCAPE – opcional, presença permitida para compatibilidade com legado e não tem outro efeito.
- path (opcional) – zero ou mais caminhos JSON, assumirá o padrão de raiz se nenhum for fornecido. Os argumentos do caminho devem ser colocados no final.

Return

Sintaxe do caminho aprimorada:

Se um caminho for fornecido:

- Retornará a string serializada de uma matriz de valores.
- Se nenhum valor for selecionado, o comando retornará uma matriz vazia.

Se vários caminhos forem fornecidos:

- Retornará um objeto JSON em formato de string, no qual cada caminho é uma chave.
- Se houver sintaxe mista de caminho aprimorado e restrito, o resultado estará de acordo com a sintaxe aprimorada.
- Se um caminho não existir, seu valor correspondente será uma matriz vazia.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
```

```
OK
127.0.0.1:6379> JSON.GET k1 $.address.*
"[\"21 2nd Street\",\"New York\",\"NY\",\"10021-3100\"]"
127.0.0.1:6379> JSON.GET k1 indent "\t" space " " NEWLINE "\n" $.address.*
"[\n\t\"21 2nd Street\",\n\t\"New York\",\n\t\"NY\",\n\t\"10021-3100\"\n]"
127.0.0.1:6379> JSON.GET k1 $.firstName $.lastName $.age
"{\"$.firstName\":[\"John\"],\"$.lastName\":[\"Smith\"],\"$.age\":[27]}"
127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}}'
OK
127.0.0.1:6379> json.get k2 $..*
"[{},{\"a\":1},{\"a\":1,\"b\":2},1,1,2]"
```

Sintaxe do caminho restrita:

```
 127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
127.0.0.1:6379> JSON.GET k1 .address
"{\"street\":\"21 2nd Street\",\"city\":\"New York\",\"state\":\"NY\",\"zipcode\":
\"10021-3100\"}"
127.0.0.1:6379> JSON.GET k1 indent "\t" space " " NEWLINE "\n" .address
"{\n\t\"street\": \"21 2nd Street\",\n\t\"city\": \"New York\",\n\t\"state\": \"NY\",\n
\t\"zipcode\": \"10021-3100\"\n}"
127.0.0.1:6379> JSON.GET k1 .firstName .lastName .age
"{\".firstName\":\"John\",\".lastName\":\"Smith\",\".age\":27}"
```

JSON.MGET

Obtém JSONs serializados no caminho a partir de várias chaves de documento. Retorna nulo para uma chave ou caminho JSON não existente.

Sintaxe

```
JSON.MGET <key> [key ...] <path>
```

- chave (obrigatório) - Uma ou mais chaves Redis do tipo documento.

- path (obrigatório) – um caminho JSON

Return

- Matriz de Strings em Massa. O tamanho da matriz é igual ao número de chaves no comando. Cada elemento da matriz será preenchido com (a) o JSON serializado conforme localizado pelo caminho ou (b) Null se a chave não existir, o caminho não existir no documento ou o caminho for inválido (erro de sintaxe).
- Se alguma das chaves especificadas existir e não for uma chave JSON, o comando retornará o erro `WRONGTYPE`.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"address":{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021"}}'
OK
127.0.0.1:6379> JSON.SET k2 . '{"address":{"street":"5 main
 Street","city":"Boston","state":"MA","zipcode":"02101"}}'
OK
127.0.0.1:6379> JSON.SET k3 . '{"address":{"street":"100 Park
 Ave","city":"Seattle","state":"WA","zipcode":"98102"}}'
OK
127.0.0.1:6379> JSON.MGET k1 k2 k3 $.address.city
1) "[\"New York\"]"
2) "[\"Boston\"]"
3) "[\"Seattle\"]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"address":{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021"}}'
OK
127.0.0.1:6379> JSON.SET k2 . '{"address":{"street":"5 main
 Street","city":"Boston","state":"MA","zipcode":"02101"}}'
OK
127.0.0.1:6379> JSON.SET k3 . '{"address":{"street":"100 Park
 Ave","city":"Seattle","state":"WA","zipcode":"98102"}}'
```

```
OK

127.0.0.1:6379> JSON.MGET k1 k2 k3 .address.city
1) "\"New York\""
2) "\"Seattle\""
3) "\"Seattle\""
```

JSON.NUMINCRBY

Incrementa os valores numéricos no caminho por um determinado número.

Sintaxe

```
JSON.NUMINCRBY <key> <path> <number>
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- número (obrigatório) – um número

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de strings em massa que representa o valor resultante em cada caminho.
- Se um valor não for um número, seu valor de retorno correspondente será nulo.
- Erro `WRONGTYPE` se o número não puder ser analisado.
- Erro `OVERFLOW` se o resultado estiver fora do intervalo de duplo IEEE de 64 bits.
- `NONEXISTENT` se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Matriz de strings em massa que representa os valores resultantes.
- Se vários valores forem selecionados, o comando retornará o resultado do último valor atualizado.
- Erro `WRONGTYPE` se o valor no caminho não for um número.
- Erro `WRONGTYPE` se o número não puder ser analisado.
- Erro `OVERFLOW` se o resultado estiver fora do intervalo de duplo IEEE de 64 bits.

- NONEXISTENT se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 $.d[*] 10
"[11,12,13]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[11,12,13]}"

127.0.0.1:6379> JSON.SET k1 $ '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 $.a[*] 1
"[]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.b[*] 1
"[2]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.c[*] 1
"[2,3]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.d[*] 1
"[2,3,4]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[2,3,4]}"

127.0.0.1:6379> JSON.SET k2 $ '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k2 $.a.* 1
"[]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.b.* 1
"[2]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.c.* 1
"[2,3]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.d.* 1
"[2,3,4]"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":2,\"b\":3,\"c\":4}}"

127.0.0.1:6379> JSON.SET k3 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
```

```
127.0.0.1:6379> JSON.NUMINCRBY k3 $.a.* 1
"[null]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.b.* 1
"[null,2]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.c.* 1
"[null,null]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.d.* 1
"[2,null,4]"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":2,\"b\":\"b\",\"c\":4}}"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 .d[1] 10
"12"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[1,12,3]}"

127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 .a[*] 1
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMINCRBY k1 .b[*] 1
"2"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[1,2],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMINCRBY k1 .c[*] 1
"3"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMINCRBY k1 .d[*] 1
"4"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[2,3,4]}"

127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k2 .a.* 1
```

```
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMINCRBY k2 .b.* 1
"2"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":1,\"b\":2},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMINCRBY k2 .c.* 1
"3"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMINCRBY k2 .d.* 1
"4"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":2,\"b\":3,\"c\":4}}"

127.0.0.1:6379> JSON.SET k3 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k3 .a.* 1
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMINCRBY k3 .b.* 1
"2"
127.0.0.1:6379> JSON.NUMINCRBY k3 .c.* 1
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMINCRBY k3 .d.* 1
"4"
```

JSON.NUMMULTBY

Multiplica os valores numéricos no caminho por um determinado número.

Sintaxe

```
JSON.NUMMULTBY <key> <path> <number>
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON
- número (obrigatório) – um número

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de strings em massa que representa o valor resultante em cada caminho.
- Se um valor não for um número, seu valor de retorno correspondente será nulo.
- Erro `WRONGTYPE` se o número não puder ser analisado.
- Erro `OVERFLOW` se o resultado estiver fora do intervalo de duplo IEEE de 64 bits.
- `NONEXISTENT` se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Matriz de strings em massa que representa os valores resultantes.
- Se vários valores forem selecionados, o comando retornará o resultado do último valor atualizado.
- Erro `WRONGTYPE` se o valor no caminho não for um número.
- Erro `WRONGTYPE` se o número não puder ser analisado.
- Erro `OVERFLOW` se o resultado estiver fora do intervalo de duplo IEEE de 64 bits.
- `NONEXISTENT` se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 $.d[*] 2
"[2,4,6]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[2,4,6]}"

127.0.0.1:6379> JSON.SET k1 $ '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 $.a[*] 2
"[]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.b[*] 2
"[2]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.c[*] 2
"[2,4]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.d[*] 2
"[2,4,6]"
```

```
127.0.0.1:6379> JSON.SET k2 $ '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k2 $.a.* 2
"[]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.b.* 2
"[2]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.c.* 2
"[2,4]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.d.* 2
"[2,4,6]"

127.0.0.1:6379> JSON.SET k3 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k3 $.a.* 2
"[null]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.b.* 2
"[null,2]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.c.* 2
"[null,null]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.d.* 2
"[2,null,6]"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 .d[1] 2
"4"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[1,4,3]}"

127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 .a[*] 2
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMMULTBY k1 .b[*] 2
"2"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[1,2],\"d\":[1,2,3]}"
```

```
127.0.0.1:6379> JSON.NUMMULTBY k1 .c[*] 2
"4"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,4],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMMULTBY k1 .d[*] 2
"6"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,4],\"d\":[2,4,6]}"

127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k2 .a.* 2
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMMULTBY k2 .b.* 2
"2"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":1,\"b\":2},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k2 .c.* 2
"4"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":4},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k2 .d.* 2
"6"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":4},\"d\":{\"a\":2,\"b\":4,\"c\":6}}"

127.0.0.1:6379> JSON.SET k3 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k3 .a.* 2
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMMULTBY k3 .b.* 2
"2"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":1,\"b\":\"b\",\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k3 .c.* 2
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMMULTBY k3 .d.* 2
"6"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":2,\"b\":\"b\",\"c\":6}}"
```

JSON.OBJLEN

Obtém o número de chaves nos valores do objeto no caminho.

Sintaxe

```
JSON.OBJLEN <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de inteiros que representa o comprimento do objeto em cada caminho.
- Se um valor não for um objeto, seu valor de retorno correspondente será nulo.
- Nulo se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Inteiro, número de chaves no objeto.
- Se vários objetos forem selecionados, o comando retornará o comprimento do primeiro objeto.
- Erro `WRONGTYPE` se o valor no caminho não for um objeto.
- `WRONGTYPE` erro se o caminho não existir.
- Nulo se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJLEN k1 $.a
```

```
1) (integer) 0
127.0.0.1:6379> JSON.OBJLEN k1 $.a.*
(empty array)
127.0.0.1:6379> JSON.OBJLEN k1 $.b
1) (integer) 1
127.0.0.1:6379> JSON.OBJLEN k1 $.b.*
1) (nil)
127.0.0.1:6379> JSON.OBJLEN k1 $.c
1) (integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 $.c.*
1) (nil)
2) (nil)
127.0.0.1:6379> JSON.OBJLEN k1 $.d
1) (integer) 3
127.0.0.1:6379> JSON.OBJLEN k1 $.d.*
1) (nil)
2) (nil)
3) (integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 $.*
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 3
5) (nil)
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJLEN k1 .a
(integer) 0
127.0.0.1:6379> JSON.OBJLEN k1 .a.*
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.OBJLEN k1 .b
(integer) 1
127.0.0.1:6379> JSON.OBJLEN k1 .b.*
(error) WRONGTYPE JSON element is not an object
127.0.0.1:6379> JSON.OBJLEN k1 .c
(integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 .c.*
(error) WRONGTYPE JSON element is not an object
```

```
127.0.0.1:6379> JSON.OBJLEN k1 .d
(integer) 3
127.0.0.1:6379> JSON.OBJLEN k1 .d.*
(integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 .*
(integer) 0
```

JSON.OBJKEYS

Obtém nomes de chave nos valores de objeto no caminho.

Sintaxe

```
JSON.OBJKEYS <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de matriz de strings em massa. Cada elemento é uma matriz de chaves em um objeto correspondente.
- Se um valor não for um objeto, seu valor de retorno correspondente será vazio.
- Nulo se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Matriz de strings em massa. Cada elemento é um nome de chave no objeto.
- Se vários objetos forem selecionados, o comando retornará as chaves do primeiro objeto.
- Erro `WRONGTYPE` se o valor no caminho não for um objeto.
- `WRONGTYPE` erro se o caminho não existir.
- Nulo se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJKEYS k1 $.*
1) (empty array)
2) 1) "a"
3) 1) "a"
   2) "b"
4) 1) "a"
   2) "b"
   3) "c"
5) (empty array)
127.0.0.1:6379> JSON.OBJKEYS k1 $.d
1) 1) "a"
   2) "b"
   3) "c"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJKEYS k1 .*
1) "a"
127.0.0.1:6379> JSON.OBJKEYS k1 .d
1) "a"
2) "b"
3) "c"
```

JSON.RESP

Retorna o valor JSON em um determinado caminho fornecido no protocolo de Serialização do Redis (RESP). Se o valor for contêiner, a resposta será uma matriz RESP ou matriz aninhada.

- JSON nulo é mapeado para o RESP Null Bulk String.
- Valores booleanos JSON são associados às respectivas RESP Simple Strings.
- Os números inteiros são mapeados para números inteiros RESP.

- Os números de ponto flutuante duplo IEEE de 64 bits são mapeados para RESP Bulk Strings.
- As strings JSON são mapeadas para RESP Bulk Strings.
- As matrizes JSON são representados como matrizes RESP, onde o primeiro elemento é a string simples [, seguida pelos elementos da matriz.
- Os objetos JSON são representados como matrizes RESP, onde o primeiro elemento é a string simples {, seguida por pares de valores-chave, cada um dos quais é uma string RESP em massa.

Sintaxe

```
JSON.RESP <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de matrizes. Cada elemento da matriz representa a forma RESP do valor em um caminho.
- Matriz vazia se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Matriz que representa a forma RESP do valor em um caminho.
- Nulo se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
```

```
127.0.0.1:6379> JSON.RESP k1 $.address
1) 1) {
   2) 1) "street"
      2) "21 2nd Street"
   3) 1) "city"
      2) "New York"
   4) 1) "state"
      2) "NY"
   5) 1) "zipcode"
      2) "10021-3100"

127.0.0.1:6379> JSON.RESP k1 $.address.*
1) "21 2nd Street"
2) "New York"
3) "NY"
4) "10021-3100"

127.0.0.1:6379> JSON.RESP k1 $.phoneNumbers
1) 1) [
   2) 1) {
      2) 1) "type"
         2) "home"
      3) 1) "number"
         2) "555 555-1234"
   3) 1) {
      2) 1) "type"
         2) "office"
      3) 1) "number"
         2) "555 555-4567"

127.0.0.1:6379> JSON.RESP k1 $.phoneNumbers[*]
1) 1) {
   2) 1) "type"
      2) "home"
   3) 1) "number"
      2) "212 555-1234"
2) 1) {
   2) 1) "type"
      2) "office"
   3) 1) "number"
      2) "555 555-4567"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK

127.0.0.1:6379> JSON.RESP k1 .address
1) {
2) 1) "street"
   2) "21 2nd Street"
3) 1) "city"
   2) "New York"
4) 1) "state"
   2) "NY"
5) 1) "zipcode"
   2) "10021-3100"

127.0.0.1:6379> JSON.RESP k1
 1) {
 2) 1) "firstName"
    2) "John"
 3) 1) "lastName"
    2) "Smith"
 4) 1) "age"
    2) (integer) 27
 5) 1) "weight"
    2) "135.25"
 6) 1) "isAlive"
    2) true
 7) 1) "address"
    2) 1) {
       2) 1) "street"
          2) "21 2nd Street"
       3) 1) "city"
          2) "New York"
       4) 1) "state"
          2) "NY"
       5) 1) "zipcode"
          2) "10021-3100"
 8) 1) "phoneNumbers"
```

```
    2) 1) [
       2) 1) {
          2) 1) "type"
             2) "home"
          3) 1) "number"
             2) "212 555-1234"
       3) 1) {
          2) 1) "type"
             2) "office"
          3) 1) "number"
             2) "555 555-4567"
 9) 1) "children"
    2) 1) [
10) 1) "spouse"
    2) (nil)
```

JSON.SET

Define os valores JSON no caminho.

Se o caminho exigir um membro do objeto:

- Se o elemento pai não existir, o comando retornará o erro NONEXISTENT.
- Se o elemento pai existir, mas não for um objeto, o comando retornará ERROR.
- Se o elemento pai existir e for um objeto:
  - Se o membro não existir, um novo membro será anexado ao objeto pai se e somente se o objeto pai for o último filho no caminho. Caso contrário, o comando retornará o erro NONEXISTENT.
  - Se o membro existir, seu valor será substituído pelo valor JSON.

Se o caminho exigir um índice de matriz:

- Se o elemento pai não existir, o comando retornará o erro NONEXISTENT.
- Se o elemento pai existir, mas não for uma matriz o comando retornará ERROR.
- Se o elemento pai existir, mas o índice estiver fora dos limites, o comando retornará o erro OUTOFBOUNDARIES.
- Se o elemento pai existir e o índice for válido, o elemento será substituído pelo novo valor JSON.

Se o caminho solicitar um objeto ou matriz, o valor (objeto ou matriz) será substituído pelo novo valor JSON.

Sintaxe

```
JSON.SET <key> <path> <json> [NX | XX]
```

[NX | XX] Onde é possível ter 0 ou 1 de identificadores [NX | XX]

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (obrigatório) – um caminho JSON. Para uma nova chave do Redis, o caminho JSON deve ser a raiz “.”.
- NX (opcional) – Se o caminho for a raiz, defina o valor somente se a chave Redis não existir, ou seja, insira um novo documento. Se o caminho não for a raiz, defina o valor somente se o caminho não existir, ou seja, insira um valor no documento.
- XX (opcional) – Se o caminho for a raiz, defina o valor somente se a chave Redis existir, ou seja, substitua o documento existente. Se o caminho não for a raiz, defina o valor somente se o caminho existir, ou seja, atualize o valor existente.

Return

- String simples 'OK' em caso de sucesso.
- Nulo se a condição NX ou XX não for atendida.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{"a":1, "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.SET k1 $.a.* '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"a\":{\"a\":0,\"b\":0,\"c\":0}}"

127.0.0.1:6379> JSON.SET k2 . '{"a": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.SET k2 $.a[*] '0'
OK
```

```
127.0.0.1:6379> JSON.GET k2
"{\"a\":[0,0,0,0,0]}"
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"c":{"a":1, "b":2}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.SET k1 .c.a '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"c\":{\"a\":0,\"b\":2},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.SET k1 .e[-1] '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"c\":{\"a\":0,\"b\":2},\"e\":[1,2,3,4,0]}"
127.0.0.1:6379> JSON.SET k1 .e[5] '0'
(error) OUTOFBOUNDARIES Array index is out of bounds
```

JSON.STRAPPEND

Anexa uma string às strings JSON no caminho.

Sintaxe

```
JSON.STRAPPEND <key> [path] <json_string>
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido
- json_string (obrigatório) – A representação JSON de uma string. Observe que uma string JSON deve estar entre aspas, ou seja, “foo”.

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de inteiros,e representando o novo comprimento da matriz em cada caminho.
- Se um valor não for uma string, seu valor de retorno correspondente será nulo.
- Erro `SYNTAXERR` se o argumento de entrada json não for uma string JSON válida.

- Erro `NONEXISTENT` se o caminho não existir.

Se o caminho for uma sintaxe restrita:

- Inteiro, o novo comprimento da string.
- Se vários valores de string forem selecionados, o comando retornará o novo comprimento da última string atualizada.
- Erro `WRONGTYPE` se o valor no caminho não for uma string.
- Erro `WRONGTYPE` se o argumento da entrada json não for uma string JSON válida.
- `NONEXISTENT` erro se o caminho não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRAPPEND k1 $.a.a '"a"'
1) (integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 $.a.* '"a"'
1) (integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 $.b.* '"a"'
1) (integer) 2
2) (nil)
127.0.0.1:6379> JSON.STRAPPEND k1 $.c.* '"a"'
1) (integer) 2
2) (integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 $.c.b '"a"'
1) (integer) 4
127.0.0.1:6379> JSON.STRAPPEND k1 $.d.* '"a"'
1) (nil)
2) (integer) 2
3) (nil)
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
```

```
OK
127.0.0.1:6379> JSON.STRAPPEND k1 .a.a '"a"'
(integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 .a.* '"a"'
(integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 .b.* '"a"'
(integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 .c.* '"a"'
(integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 .c.b '"a"'
(integer) 4
127.0.0.1:6379> JSON.STRAPPEND k1 .d.* '"a"'
(integer) 2
```

JSON.STRLEN

Obtém o comprimento dos valores da string JSON no caminho.

Sintaxe

```
JSON.STRLEN <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de inteiros, que representa o comprimento do valor da string em cada caminho.
- Se um valor não for uma string, seu valor correspondente será nulo.
- Nulo se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- Inteiro, o comprimento da string.
- Se vários valores de string forem selecionados, o comando retornará o comprimento da primeira string.

- Erro `WRONGTYPE` se o valor no caminho não for uma string.
- `NONEXISTENT` erro se o caminho não existir.
- Nulo se a chave do documento não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRLEN k1 $.a.a
1) (integer) 1
127.0.0.1:6379> JSON.STRLEN k1 $.a.*
1) (integer) 1
127.0.0.1:6379> JSON.STRLEN k1 $.c.*
1) (integer) 1
2) (integer) 2
127.0.0.1:6379> JSON.STRLEN k1 $.c.b
1) (integer) 2
127.0.0.1:6379> JSON.STRLEN k1 $.d.*
1) (nil)
2) (integer) 1
3) (nil)
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRLEN k1 .a.a
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .a.*
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .c.*
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .c.b
(integer) 2
127.0.0.1:6379> JSON.STRLEN k1 .d.*
(integer) 1
```

JSON.TOGGLE

Alterna valores booleanos entre verdadeiro e falso no caminho.

Sintaxe

```
JSON.TOGGLE <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de inteiros (0 - falso, 1 - verdadeiro) que representam o valor booleano resultante em cada caminho.
- Se um valor for um não booleano, seu valor de retorno correspondente será null.
- `NONEXISTENT` se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- String (“verdadeiro”/“falso”) que representa o valor booleano resultante.
- `NONEXISTENT` se a chave do documento não existir.
- Erro `WRONGTYPE` se o valor no caminho não for um valor booleano.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":true, "b":false, "c":1, "d":null, "e":"foo", "f":
[], "g":{}}'
OK
127.0.0.1:6379> JSON.TOGGLE k1 $.*
1) (integer) 0
2) (integer) 1
```

```
3) (nil)
4) (nil)
5) (nil)
6) (nil)
7) (nil)
127.0.0.1:6379> JSON.TOGGLE k1 $.*
1) (integer) 1
2) (integer) 0
3) (nil)
4) (nil)
5) (nil)
6) (nil)
7) (nil)
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 . true
OK
127.0.0.1:6379> JSON.TOGGLE k1
"false"
127.0.0.1:6379> JSON.TOGGLE k1
"true"

127.0.0.1:6379> JSON.SET k2 . '{"isAvailable": false}'
OK
127.0.0.1:6379> JSON.TOGGLE k2 .isAvailable
"true"
127.0.0.1:6379> JSON.TOGGLE k2 .isAvailable
"false"
```

JSON.TYPE

Informa tipo de valores em um determinado caminho.

Sintaxe

```
JSON.TYPE <key> [path]
```

- chave (obrigatório) – uma chave Redis do tipo de documento JSON
- path (opcional) – um caminho JSON. Assumirá o padrão da raiz se não for fornecido

Return

Se o caminho for uma sintaxe aprimorada:

- Matriz de strings, que representa o tipo de valor em cada caminho. O tipo é um destes {"null", “boolean”, “string”, “number”, “integer”, “object” e “array"}.
- Se um caminho não existir, seu valor de retorno correspondente será nulo.
- Matriz vazia se a chave do documento não existir.

Se o caminho for uma sintaxe restrita:

- String, tipo do valor
- Nulo se a chave do documento não existir.
- Nulo se o caminho JSON for inválido ou não existir.

Exemplos

Sintaxe do caminho aprimorada:

```
127.0.0.1:6379> JSON.SET k1 . '[1, 2.3, "foo", true, null, {}, []]'
OK
127.0.0.1:6379> JSON.TYPE k1 $[*]
1) integer
2) number
3) string
4) boolean
5) null
6) object
7) array
```

Sintaxe do caminho restrita:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
```

```
127.0.0.1:6379> JSON.TYPE k1
object
127.0.0.1:6379> JSON.TYPE k1 .children
array
127.0.0.1:6379> JSON.TYPE k1 .firstName
string
127.0.0.1:6379> JSON.TYPE k1 .age
integer
127.0.0.1:6379> JSON.TYPE k1 .weight
number
127.0.0.1:6379> JSON.TYPE k1 .isAlive
boolean
127.0.0.1:6379> JSON.TYPE k1 .spouse
null
```

Uso de tags em seus recursos do MemoryDB

Para ajudar você a gerenciar seus clusters e outros recursos do MemoryDB, é possível atribuir seus próprios metadados a cada recurso na forma de tags. As tags permitem categorizar seus recursos da AWS de diferentes formas (como por finalidade, por proprietário ou por ambiente). Isso é útil quando você tem muitos recursos do mesmo tipo — é possível identificar rapidamente um recurso específico baseado nas tags que você atribuiu a ele. Este tópico descreve tags e mostra a você como criá-los.

> **Warning**
>
> Como uma prática recomendada, sugerimos que você não inclua dados confidenciais nas suas tags.

Conceitos básicos de tags

Uma etiqueta é um rótulo atribuído a um recurso da AWS. Cada tag consiste em uma chave e um valor opcional, ambos definidos por você. As tags permitem categorizar seus recursos da AWS de diferentes formas, por exemplo, por finalidade ou por proprietário. Por exemplo, você pode definir um conjunto de tags para os clusters do MemoryDB da sua conta que o ajudem a rastrear o proprietário e o grupo de usuários de cada cluster.

Recomendamos que você desenvolva um conjunto de chave de tags que atenda suas necessidades para cada tipo de recurso. Usar um conjunto consistente de chaves de tags facilita para você

gerenciar seus recursos. É possível pesquisar e filtrar os recursos de acordo com as tags que adicionar. Para obter mais informações sobre como implementar uma estratégia eficaz de marcação de recursos, consulte o whitepaper da AWS, Práticas recomendadas de marcação.

As tags não têm nenhum significado semântico para o MemoryDB e são interpretadas estritamente como uma sequência de caracteres. Além disso, as tags não são automaticamente atribuídas aos seus recursos. É possível editar chaves de tags e valores, e é possível remover as tags de um recurso a qualquer momento. É possível definir o valor de uma tag como `null`. Se você adicionar uma tag que tenha a mesma chave de uma tag existente nesse recurso, o novo valor substituirá o antigo. Se você excluir um recurso, todas as tags do recurso também serão excluídas.

Você pode trabalhar com tags usando o AWS Management Console, a AWS CLI e a API do MemoryDB.

Se você estiver usando o IAM, pode controlar quais usuários na sua conta da AWS têm permissão para criar, editar ou excluir tags. Para obter mais informações, consulte Permissões em nível de recurso.

Recursos que podem ser marcados

Você pode usar tags na maioria dos recursos do MemoryDB que já existem em sua conta. A tabela a seguir lista os recursos compatíveis com o uso de tags. Se você estiver usando o AWS Management Console, é possível aplicar tags aos recursos usando o Tag Editor. Algumas telas de recursos permitem que você especifique tags para um recurso ao criá-lo; por exemplo, uma tag com uma chave de nome e um valor que você especificar. Na maioria dos casos, o console aplicará as tags imediatamente depois de o recurso ser criado (em vez de durante a criação de recursos). O console pode organizar os recursos de acordo com a tag Nome, mas essa tag não tem nenhum significado semântico para o serviço do MemoryDB.

Além disso, algumas ações de criação de recursos permitem que você especifique tags para um recurso quando ele é criado. Se as tags não puderem ser aplicadas durante a criação dos recursos, nós reverteremos o processo de criação de recursos. Isso garante que os recursos sejam criados com tags ou, então, não criados, e que nenhum recurso seja deixado sem tags. Ao marcar com tags os recursos no momento da criação, você elimina a necessidade de executar scripts personalizados de uso de tags após a criação do recurso.

Se estiver usando a API do MemoryDB da Amazon, a AWS CLI ou um SDK da AWS, você poderá usar o parâmetro na ação relevante da API do MemoryDB para aplicar tags. Elas são:

- `CreateCluster`

- `CopySnapshot`
- `CreateParameterGroup`
- `CreateSubnetGroup`
- `CreateSnapshot`
- `CreateACL`
- `CreateUser`

A tabela a seguir descreve os recursos do MemoryDB que podem ser marcados e os recursos que podem ser marcados na criação usando a API do MemoryDB, a AWS CLI ou um SDKda AWS.

Suporte à marcação para recursos do MemoryDB

Compatível com tags	Oferece suporte à marcação na criação
Sim	Sim
Sim	Sim
Sim	Sim
Sim	Sim
Sim	Sim
Sim	Sim

É possível aplicar permissões em nível de recurso baseadas em tags em suas políticas do IAM às ações da API do MemoryDB que oferecem suporte à marcação na criação para implementar um controle granular sobre os usuários e grupos que podem marcar recursos na criação. Seus recursos estão devidamente protegidos a partir da criação. As tags são aplicadas imediatamente aos recursos. Portanto, todas as permissões em nível de recurso baseadas em tags que controlam o uso de recursos entram imediatamente em vigor. Seus recursos podem ser rastreados e relatados com mais precisão. É possível obrigar o uso de marcação com tags nos novos recursos e controlar quais chaves e valores de tag são definidos nos seus recursos.

Para obter mais informações, consulte Exemplo de marcação de recursos.

Para obter mais informações sobre como marcar os seus recursos para o faturamento, consulte Monitoramento de custos com tags de alocação de custos.

Tags em clusters e snapshots

As seguintes regras se aplicam à marcação como parte das operações de solicitação:

- CreateCluster:
  - Se o `--cluster-name` for fornecido:

    Se as tags forem incluídas na solicitação, o cluster será marcado.
  - Se o `--snapshot-name` for fornecido:

    Se as tags forem incluídas na solicitação, o cluster será marcado somente com essas tags. Se nenhuma tag for incluída na solicitação, as tags de snapshot serão adicionadas ao cluster.
- CreateSnapshot:
  - Se o `--cluster-name` for fornecido:

    Se as tags forem incluídas na solicitação, somente as tags de solicitação serão adicionadas ao snapshot. Se nenhuma tag for incluída na solicitação, as tags de cluster de cache serão adicionadas ao snapshot.
  - Para snapshots automáticos:

    As tags serão propagadas a partir das tags do cluster.
- CopySnapshot:

  Se as tags forem incluídas na solicitação, somente as tags de solicitação serão adicionadas ao snapshot. Se nenhuma tag for incluída na solicitação, as tags de snapshot da origem serão adicionadas ao snapshot copiado.
- TagResource e UntagResource:

  As tags serão adicionadas/removidas do recurso.

Restrições de tags

As restrições básicas a seguir se aplicam às tags:

- Número máximo de tags por recurso: 50
- Em todos os recursos, cada chave de etiqueta deve ser exclusiva e pode ter apenas um valor.
- Comprimento máximo da chave – 128 caracteres Unicode em UTF-8.
- Comprimento máximo do valor – 256 caracteres Unicode em UTF-8.
- Embora o MemoryDB permita qualquer caractere em suas tags, outros serviços podem ser restritivos. Os caracteres permitidos nos serviços são: letras, números e espaços representáveis em UTF-8 e os seguintes caracteres: + - = . _ : / @
- As chaves e os valores de tags diferenciam maiúsculas de minúsculas.
- O prefixo `aws:` é reservado para uso da AWS. Não é possível editar nem excluir a chave ou o valor de uma tag quando ela tem uma chave de tag com esse prefixo. As tags com o prefixo `aws:` não contam para as tags por limite de recurso.

Você não pode encerrar, parar ou excluir um recurso baseado unicamente em suas tags; será preciso especificar o identificador de recursos. Por exemplo, para excluir snapshots marcados com uma chave de tag chamada `DeleteMe`, use a ação `DeleteSnapshot` com os identificadores de recursos dos snapshots, como `snap-1234567890abcdef0`.

Para obter mais informações sobre os recursos do MemoryDB que você pode usar tags, consulte Recursos que podem ser marcados.

Exemplo de marcação de recursos

- Adicionar tags a um cluster.

```
aws memorydb tag-resource \
--resource-arn arn:aws:memorydb:us-east-1:111111222233:cluster/my-cluster \
--tags Key="project",Value="XYZ" Key="memorydb",Value="Service"
```

- Criação de um cluster usando tags.

```
aws memorydb create-cluster \
--cluster-name testing-tags \
--description cluster-test \
--subnet-group-name test \
--node-type db.r6g.large \
--acl-name open-access \
--tags Key="project",Value="XYZ" Key="memorydb",Value="Service"
```

- Criação de um snapshot com tags.

  Para esse caso, se você adicionar tags sob solicitação, mesmo que o cluster contenha tags, o snapshot receberá somente as tags da solicitação.

```
aws memorydb create-snapshot \
--cluster-name testing-tags \
--snapshot-name bkp-testing-tags-mycluster \
--tags Key="work",Value="foo"
```

Monitoramento de custos com tags de alocação de custos

Quando você adiciona tags de alocação de custos aos seus recursos no MemoryDB para Redis, pode acompanhar os custos agrupando as despesas nas suas faturas por valores de tag de recursos.

Uma tag de alocação de custos do MemoryDB é um par de valores-chave que você define e associa a um recurso do MemoryDB. A chave e o valor diferenciam maiúsculas de minúsculas. Você pode usar uma chave de tag para definir uma categoria, e o valor da tag pode ser um item nessa categoria. Por exemplo, você pode definir uma chave de tag de `CostCenter` e um valor de tag de `10010`, indicando que o recurso está atribuído ao centro de custo 10010. Você também pode usar tags para designar recursos como sendo usados para teste ou produção, usando uma chave como `Environment` e valores como `test` ou `production`. Recomendamos que você use um conjunto consistente de chaves de tag para facilitar o rastreamento dos custos associados aos seus recursos.

Use tags de alocação de custos para organizar sua fatura da AWS para refletir sua própria estrutura de custos. Para isso, inscreva-se para obter sua conta da AWS com os valores de chave de tag incluídos. Então, para ver o custo de recursos combinados, organize suas informações de faturamento de acordo com recursos com os mesmos valores de chave de tags. Por exemplo, você pode etiquetar vários recursos com um nome de aplicação específico, e depois organizar suas informações de faturamento para ver o custo total daquela aplicação em vários serviços.

Você também pode combinar tags para rastrear custos com um maior nível de detalhes. Por exemplo, para rastrear seus custos de serviços por região, você pode usar as chaves de tag `Service` e `Region`. Em um recurso, você pode ter os valores `MemoryDB` e `Asia Pacific (Singapore)` e, em outro recurso, os valores `MemoryDB` e `Europe (Frankfurt)`. você pode, então, ver seus custos totais do MemoryDB divididos por região. Para obter mais informações, consulte Usar tags de alocação de custos no Guia do usuário do AWS Billing.

Você pode adicionar tags de alocação de custo em clusters do MemoryDB. Ao adicionar, listar, modificar, copiar ou remover uma tag, a operação é aplicada somente ao cluster especificado.

Características das tags de alocação de custos do MemoryDB

- As etiquetas de alocação de custos são aplicadas aos recursos do MemoryDB que são especificados nas operações de CLI e API como um ARN. O tipo de recurso será um "cluster".

  Formato ARN: `arn:aws:memorydb:<region>:<customer-id>:<resource-type>/<resource-name>`

  ARN de exemplo: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`
- A chave de tags é o nome obrigatório da tag. O valor da string da chave pode ser de 1 a 128 caracteres Unicode e não pode ser prefixado com `aws:`. A string pode conter apenas o conjunto de letras Unicode, dígitos, espaços em branco, sublinhados (_), pontos finais (.), dois-pontos (:), barras invertidas (\), sinais de igualdade (=), sinais de adição (+), hífens (-) ou sinais de arroba (@).
- O valor da tag é o valor opcional da tag. O valor da string do valor pode ser de 1 a 256 caracteres Unicode e não pode ser prefixado com `aws:`. A string pode conter apenas o conjunto de letras Unicode, dígitos, espaços em branco, sublinhados (_), pontos finais (.), dois-pontos (:), barras invertidas (\), sinais de igualdade (=), sinais de adição (+), hífens (-) ou sinais de arroba (@).
- Um recurso do MemoryDB pode ter no máximo 50 tags.
- Os valores não têm que ser exclusivos em um conjunto de tags. Por exemplo, você pode ter um conjunto de tags no qual as chaves `Service` e `Application` têm ambas o valor `MemoryDB`.

A AWS não aplica nenhum significado semântico às suas tags. As tags são interpretadas estritamente como cadeias de caracteres. AWS a não define automaticamente nenhuma tag em nenhum recurso do MemoryDB.

Gerenciamento das suas tags de alocação de custos usando a AWS CLI

Você pode usar a AWS CLI para adicionar, modificar ou remover tags de alocação de custos.

Amostra de ARN: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`

Tópicos

- Modificação de tags usando a AWS CLI
- Remoção der tags usando a AWS CLI

Listagem de tags usando a AWS CLI

Você pode usar a opção AWS CLI para listar tags em um recurso existente do MemoryDB usando a operação list-tags.

O código a seguir usa a AWS CLI para listar as tags no cluster `my-cluster` do MemoryDB na região us-east-1.

Para Linux, macOS ou Unix:

```
aws memorydb list-tags \
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
```

Para Windows:

```
aws memorydb list-tags ^
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
```

A saída dessa operação será semelhante a uma lista de todas as tags no recurso.

```
{
   "TagList": [
      {
         "Value": "10110",
         "Key": "CostCenter"
      },
      {
         "Value": "EC2",
         "Key": "Service"
      }
   ]
}
```

Se não houver tags no recurso, a saída será uma TagList vazia.

```
{
```

```
   "TagList": []
}
```

Para obter mais informações, consulte a AWS CLI para a lista de tags do MemoryDB.

Adição de tags usando a AWS CLI

Você pode usar AWS CLI para adicionar tags a um recurso do MemoryDB existente usando a operação tag-resource da CLI. Se a chave de tag não existir no recurso, a chave e o valor serão adicionados ao recurso. Se a chave já existir no recurso, o valor associado a essa chave será atualizado para o novo valor.

O código a seguir usa AWS CLI para adicionar as chaves `Service` e `Region` com os valores `memorydb` e `us-east-1`, respectivamente, ao cluster `my-cluster` na região us-east-1.

Para Linux, macOS ou Unix:

```
aws memorydb tag-resource \
 --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster \
 --tags Key=Service,Value=memorydb \
        Key=Region,Value=us-east-1
```

Para Windows:

```
aws memorydb tag-resource ^
 --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster ^
 --tags Key=Service,Value=memorydb ^
        Key=Region,Value=us-east-1
```

A saída dessa operação será semelhante a uma lista de todas as tags no recurso após a operação, conforme mostrado a seguir.

```
{
   "TagList": [
      {
         "Value": "memorydb",
         "Key": "Service"
      },
      {
         "Value": "us-east-1",
```

```
        "Key": "Region"
      }
   ]
}
```

Para obter mais informações, consulte AWS CLI para MemoryDBtag-resource.

Você também pode usar AWS CLI para adicionar tags a um cluster ao criar um novo cluster usando a operação create-cluster.

Modificação de tags usando a AWS CLI

Você pode usar AWS CLI para modificar as tags em um cluster do MemoryDB.

Para modificar tags:

- Use tag-resource para adicionar uma nova tag e um valor ou para alterar o valor associado a uma tag existente.
- Use untag-resource para remover tags especificadas do recurso.

A saída de qualquer operação será uma lista de tags e seus valores no cluster especificado.

Remoção der tags usando a AWS CLI

Você pode usar a operação untag-resource da AWS CLI para remover tags de um cluster existente do MemoryDB.

O código a seguir usa a AWS CLI para remover as etiquetas com as chaves `Service` e `Region` do cluster `my-cluster` na região us-east-1.

Para Linux, macOS ou Unix:

```
aws memorydb untag-resource \
 --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster \
 --tag-keys Region Service
```

Para Windows:

```
aws memorydb untag-resource ^
```

```
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster ^
  --tag-keys Region Service
```

A saída dessa operação será semelhante a uma lista de todas as tags no recurso após a operação, conforme mostrado a seguir.

```
{
   "TagList": []
}
```

Para obter mais informações, consulte AWS CLI para untag-resource do MemoryDB.

Gerenciar suas tags de alocação de custos usando a API do MemoryDB

Você pode usar a API do MemoryDB para adicionar, modificar ou remover tags de alocação de custos.

Tags de alocação de custos são aplicadas para clusters do MemoryDB. O cluster que receberá tag é especificado usando um ARN (Nome de recurso da Amazon).

Amostra de ARN: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`

Tópicos

- Listagem de tags usando a API do MemoryDB
- Adicionar tags usando a API do MemoryDB
- Modificação de tags usando a API do MemoryDB
- Remover tags usando a API do MemoryDB

Listagem de tags usando a API do MemoryDB

Você pode usar a API do MemoryDB para listar tags em um recurso existente usando a operação ListTags.

O código a seguir usa a API do MemoryDB para listar as tags no recurso `my-cluster` na região us-east-1.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=ListTags
```

```
   &ResourceArn=arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Version=2021-01-01
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

Adicionar tags usando a API do MemoryDB

Você pode usar a API do MemoryDB para adicionar tags a um cluster do MemoryDB existente usando a operação TagResource. Se a chave de tag não existir no recurso, a chave e o valor serão adicionados ao recurso. Se a chave já existir no recurso, o valor associado a essa chave será atualizado para o novo valor.

O código a seguir usa a API do MemoryDB para adicionar as chaves `Service` e `Region` com os valores `memorydb` e `us-east-1`, respectivamente, ao recurso `my-cluster` na região us-east-1.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=TagResource
   &ResourceArn=arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Tags.member.1.Key=Service
   &Tags.member.1.Value=memorydb
   &Tags.member.2.Key=Region
   &Tags.member.2.Value=us-east-1
   &Version=2021-01-01
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte TagResource.

Modificação de tags usando a API do MemoryDB

Você pode usar a API do MemoryDB para modificar as tags em um cluster do MemoryDB.

Para modificar o valor de uma tag:

- Use a operação TagResource para adicionar uma nova tag e um valor ou para alterar o valor de uma tag existente.
- Para remover tags de um recurso, use a ação UntagResource.

A saída de qualquer operação será uma lista de tags e seus valores no recurso especificado.

Remover tags usando a API do MemoryDB

Você pode usar a API do MemoryDB para remover tags de um cluster do MemoryDB existente usando a operação UntagResource.

O código a seguir usa a API do MemoryDB para remover as tags com as chaves `Service` e `Region` do cluster `my-cluster` na região us-east-1.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=UntagResource
   &ResourceArn=arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &TagKeys.member.1=Service
   &TagKeys.member.2=Region
   &Version=2021-01-01
   &Timestamp=20210802T192317Z
   &X-Amz-Credential=<credential>
```

Gerenciamento da manutenção

Cada cluster tem uma janela de manutenção semanal durante a qual todas as alterações do sistema são aplicadas. Se você não especificar uma janela de manutenção preferencial ao criar ou modificar um cluster, o MemoryDB atribuirá uma janela de manutenção de 60 minutos dentro da janela de manutenção da sua região em um dia da semana escolhido aleatoriamente.

A janela de manutenção de 60 minutos é escolhida aleatoriamente entre um período de 8 horas por região. A tabela a seguir lista os blocos de tempo de cada região dos quais as janelas de manutenção padrão são atribuídas. Você pode escolher uma janela de manutenção preferida fora do bloco de janelas de manutenção da região.

Código da região	Nome da região	Janela de manutenção da região
`ap-northeast-1`	Região Ásia-Pacífico (Tóquio)	13h às 21h (UTC)
`ap-northeast-2`	Região Ásia-Pacífico (Seul)	12h às 20h (UTC)

Código da região	Nome da região	Janela de manutenção da região
ap-south-1	Região Ásia-Pacífico (Mumbai)	17h30 à 1h30 UTC
ap-southeast-1	Região Ásia-Pacífico (Singapura)	14h às 22h (UTC)
ap-east-1	Região Ásia-Pacífico (Hong Kong)	13h às 21h (UTC)
ap-southeast-2	Região Ásia-Pacífico (Sydney)	12h às 20h (UTC)
cn-north-1	Região da China (Pequim)	14h às 22h (UTC)
cn-northwest-1	Região da China (Ningxia)	14h às 22h (UTC)
eu-west-3	Região Europa (Paris)	De 23:59 a 07:29 UTC
eu-central-1	Região Europa (Frankfurt)	23h às 7h (UTC)
eu-west-1	Região Europa (Irlanda)	22h às 6h (UTC)
eu-west-2	Região Europa (Londres)	23h às 7h (UTC)
sa-east-1	Região América do Sul (São Paulo)	1h às 9h UTC
ca-central-1	Região do Canadá (Central)	3h às 7h (UTC)
us-east-1	Região Leste dos EUA (N. da Virgínia)	3h às 7h (UTC)
us-east-1	Região Leste dos EUA (Ohio)	5h às 12h UTC
us-west-1	Região Leste dos EUA (Norte da Califórnia)	6h às 14h (UTC)
us-west-2	Região Oeste dos EUA (Oregon)	6h às 14h (UTC)

Como alterar a janela de manutenção do cluster

A janela de manutenção deve ser definida no horário de menor utilização e, portanto, talvez precise ser modificada de vez em quando. Você pode modificar o cluster para especificar um intervalo de até

24 horas de duração durante o qual todas as atividades de manutenção solicitadas devem ocorrer. Todas as modificações de cluster diferidas ou pendentes que você tiver solicitado ocorrem durante esse período.

Mais informações

Para obter informações sobre sua janela de manutenção e substituição de nó, consulte:

- Substituição de nós: Gerenciamento de substituição de nó
- Modificar um cluster do MemoryDB: Alteração da janela de manutenção de um cluster

Práticas recomendadas

A seguir, você encontrará as práticas recomendadas para o MemoryDB para Redis. Seguir essas práticas melhora o desempenho e a confiabilidade do seu cluster.

Tópicos

- Comandos restritos do Redis
- Resiliência no MemoryDB para Redis
- Práticas recomendadas: pub/sub e multiplexação de E/S aprimorada
- Práticas recomendadas: redimensionamento online de clusters

Comandos restritos do Redis

Para oferecer uma experiência de serviço gerenciado, o MemoryDB restringe o acesso a determinados comandos que exigem privilégios avançados. Os seguintes comandos não estão disponíveis:

- `acl deluser`
- `acl load`
- `acl save`
- `acl setuser`
- `bgrewriteaof`
- `bgsave`
- `cluster addslot`
- `cluster delslot`
- `cluster setslot`
- `config`
- `debug`
- `migrate`
- `module`
- `psync`
- `replicaof`
- `save`
- `shutdown`
- `slaveof`
- `sync`

Resiliência no MemoryDB para Redis

A infraestrutura global da AWS é criada com base em regiões da AWS e zonas de disponibilidade. As regiões da AWS As regiões fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, as quais são conectadas com baixa latência, alto throughput e redes altamente redundantes. Com as zonas de disponibilidade, você pode 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.

Para obter mais informações sobre regiões e zonas de disponibilidade da AWS, consulte Infraestrutura global da AWS.

Além da infraestrutura global da AWS, o MemoryDB para Redis oferece vários atributos para ajudar a oferecer suporte às suas necessidades de resiliência de dados e snapshot.

Tópicos

Atenuar falhas

Ao planejar sua implementação do MemoryDB para Redis, você deve planejar para que as falhas tenham um impacto mínimo sobre seu aplicativo e seus dados. Os tópicos nesta seção discutem as abordagens que você pode tomar para proteger seu aplicativo e dados contra falhas.

Mitigando falhas: clusters do MemoryDB

Um cluster do MemoryDB é composto por um único nó primário no/do qual seu aplicativo pode ler e gravar e de 0 a 5 nós de réplica somente para leitura. No entanto, é altamente recomendável usar pelo menos uma réplica para alta disponibilidade. Sempre que os dados são gravados no nó primário, eles são mantidos no log de transações e atualizados de forma assíncrona nos nós de réplica.

Quando uma réplica de leitura falha

1. O MemoryDB detecta a réplica com falha.
2. O MemoryDB coloca o nó com falha offline.
3. O MemoryDB inicia e provisiona um nó de substituição na mesma zona de disponibilidade (Available Zone, AZ).

4. O novo nó é sincronizado com o log de transações.

Durante esse período, seu aplicativo pode continuar lendo e gravando usando os outros nós.

Multi-AZ do MemoryDB

Se o Multi-AZ for ativado em seus clusters do MemoryDB, uma falha primária será detectada e substituída automaticamente.

1. O MemoryDB detecta a falha do nó primário.
2. O MemoryDB faz o failover para uma réplica depois de garantir que ela seja consistente com o primário que falhou.
3. O MemoryDB gira uma réplica na AZ do primário com falha.
4. O novo nó é sincronizado com o log de transações.

O failover em um nó de réplica geralmente é mais rápido do que criar e provisionar um novo nó primário. Isso significa que seu aplicativo pode retomar a gravação no nó primário mais cedo.

Para obter mais informações, consulte Minimização do tempo de inatividade no MemoryDB com Multi-AZ.

Práticas recomendadas: pub/sub e multiplexação de E/S aprimorada

Ao usar o Redis versão 7 ou posterior, recomendamos usar Pub/Sub fragmentado. Também é possível melhorar o throughput e a latência usando a multiplexação de E/S aprimorada, que está disponível automaticamente ao usar o Redis versão 7 ou posterior e não requer nenhuma alteração no cliente. É ideal para workloads pub/sub, que geralmente são limitadas ao throughput com várias conexões de clientes.

Práticas recomendadas: redimensionamento online de clusters

A refragmentação consiste na adição e remoção de fragmentos ou nós de seu cluster bem como na redistribuição de espaços importantes. Como resultado, vários itens têm impacto na operação de refragmentação, como a carga no cluster, a utilização de memória e o tamanho geral dos dados. Para obter a melhor experiência, recomendamos que você siga as práticas gerais recomendadas de cluster para distribuição padrão uniforme de workload. Além disso, recomendamos as etapas a seguir.

Antes de iniciar a refragmentação, recomendamos o seguinte:

- Teste sua aplicação: teste o comportamento da sua aplicação durante a refragmentação em um ambiente de preparação, se possível.
- Receba uma notificação prévia de problemas de escalabilidade: a refragmentação é uma operação que demanda uso intensivo de computação. Por esse motivo, recomendamos manter a utilização da CPU abaixo de 80% em instâncias de vários núcleos e abaixo de 50% em instâncias de núcleo único durante a refragmentação. Monitore as métricas do MemoryDB e inicie a refragmentação antes que seu aplicativo comece a observar problemas de escalabilidade. As métricas úteis para acompanhar são `CPUUtilization`, `NetworkBytesIn`, `NetworkBytesOut`, `CurrConnections`, `NewConnections`, `FreeableMemory`, `SwapUsage` e `BytesUsedForMemoryDB`.
- Garanta memória livre suficiente disponível antes da redução de escala na horizontal: se você estiver reduzindo a escala na horizontal, garanta que a memória livre disponível nos fragmentos a serem retidos é, pelo menos, 1,5 vez maior do que a memória usada nos fragmentos que você planeja remover.
- Inicie a refragmentação em horários fora de pico: essa prática ajuda a reduzir a latência e o impacto de throughput no cliente durante a operação de refragmentação. Ela também ajuda a concluir a refragmentação com mais rapidez à medida que mais recursos podem ser usados na redistribuição de slots.

- Analise o comportamento de tempo limite do cliente: alguns clientes podem observar maior latência durante o redimensionamento de cluster online. Configurar sua biblioteca de cliente com um tempo limite maior pode ajudar dando tempo para o sistema se conectar mesmo em condições de carga maiores no servidor. Em algum casos, você pode abrir um grande número de conexões com o servidor. Nesses casos, considere adicionar o recuo exponencial para uma nova conexão lógica. Fazer isso pode ajudar a evitar uma intermitência de novas conexões acessando o servidor ao mesmo tempo.

Durante a refragmentação, recomendamos o seguinte:

- Evite comandos caros: evite executar operações com uso intensivo computacional e de E/S, como os comandos `KEYS` e `SMEMBERS`. Sugerimos essa abordagem porque essas operações aumentam a carga no cluster e geram impacto no desempenho do cluster. Em vez disso, use os comandos `SCAN` e `SSCAN`.
- Siga as melhores práticas do Lua: evite scripts Lua de longa execução e sempre declare antecipadamente as chaves usadas em scripts Lua. Recomendamos essa abordagem para determinar se o script Lua não está usando comandos entre slots. Certifique-se de que as chaves usadas em scripts Lua pertencem ao mesmo slot.

Após a refragmentação, observe o seguinte:

- A redução da escala horizontalmente pode ser parcialmente bem-sucedida se não houver memória suficiente disponível nos fragmentos de destino. Se isso ocorrer, analise a memória disponível e refaça a operação, se necessário.
- Slots com itens grandes não são migrados. Especificamente, slots com itens maiores do que 256 MB após a serialização não são migrados.
- Os comandos `FLUSHALL` e `FLUSHDB` não são compatíveis em scripts Lua durante uma operação de reestilhaçamento.

Noções básicas sobre a replicação do MemoryDB

O MemoryDB implementa a replicação com dados particionados em até 500 fragmentos.

Cada fragmento em um cluster tem um único nó primário de leitura/gravação e até cinco nós de réplica somente de leitura. Cada nó primário pode sustentar até 100 MB/s. É possível criar um cluster com alto número de fragmentos e baixo número de réplicas totalizando até 500 nós por cluster. Essa

configuração do cluster pode variar de 500 fragmentos e 0 réplicas para 100 fragmentos e 4 réplicas, que é o número máximo de réplicas permitidas.

Consistência

No MemoryDB, os nós primários são bastante consistentes. As operações de gravação bem-sucedidas são armazenadas de forma duradoura em um log transacional multi-AZ distribuído antes de retornar aos clientes. As operações de leitura nos primários sempre retornam os dados mais atualizados, refletindo os efeitos de todas as operações anteriores de gravação bem-sucedidas. Essa forte consistência é preservada em todos os failovers primários.

No MemoryDB, os nós de réplica acabam sendo consistentes. As operações de leitura de réplicas (usando o comando `READONLY`) nem sempre refletem os efeitos das operações de gravação bem-sucedidas mais recentes, com métricas de atraso publicadas no CloudWatch. No entanto, as operações de leitura de uma única réplica são sequencialmente consistentes. As operações de gravação bem-sucedidas entram em vigor em cada réplica na mesma ordem em que foram executadas na primária.

Replicação em um cluster

Cada réplica de leitura em um fragmento mantém uma cópia dos dados do nó primário do fragmento. Mecanismos de replicação assíncronos usando os logs de transação são usados para manter as réplicas de leitura sincronizadas com o primário. Os aplicativos podem ler a partir de qualquer nó no cluster. Os aplicativos podem apenas gravar nos nós primários. As réplicas de leitura aprimoram a escalabilidade da leitura. Como o MemoryDB armazena os dados em logs de transações duráveis, não há risco de perda de dados. Os dados são particionados em todos os fragmentos em um cluster do MemoryDB.

Os aplicativos usam o endpoint do cluster do MemoryDB para conectar com os nós do cluster. Para obter mais informações, consulte Encontrar endpoints de conexão.

Os clusters do MemoryDB são regionais e podem conter nós somente de uma região. Para melhorar a tolerância a falhas, você pode provisionar primários e réplicas de leitura em várias zonas de disponibilidade dentro dessa região.

O uso da replicação, que fornece o Multi-AZ, é altamente recomendado para todos os clusters do MemoryDB. Para obter mais informações, consulte Minimização do tempo de inatividade no MemoryDB com Multi-AZ.

Minimização do tempo de inatividade no MemoryDB com Multi-AZ

Há várias instâncias em que o MemoryDB pode precisar substituir um nó primário; elas incluem certos tipos de manutenção planejada e o evento improvável de uma falha no nó primário ou na zona de disponibilidade.

A resposta à falha do nó depende de qual nó apresentou a falha. No entanto, em todos os casos, o MemoryDB garante que nenhum dado seja perdido durante a substituição de nós ou failover. Por exemplo, se uma réplica falhar, o nó com falha será substituído e os dados serão sincronizados a partir do log de transações. Se o nó primário falhar, um failover é acionado para uma réplica consistente, o que garante que nenhum dado seja perdido durante o failover. As gravações agora são atendidas a partir do novo nó primário. O nó primário antigo é então substituído e sincronizado a partir do log de transações.

Se um nó primário falhar em um único fragmento de nó (sem réplicas), o MemoryDB deixará de aceitar gravações até que o nó primário seja substituído e sincronizado a partir do log de transações.

Essa substituição do nó pode resultar em algum tempo de inatividade do cluster, mas, se o multi-AZ estiver ativo, o tempo de inatividade será minimizado. A função do nó primário automaticamente fará um failover para uma das réplicas. Não há necessidade de criar e provisionar um novo nó primário, porque o MemoryDB lidará com isso de forma transparente. O failover e a promoção de réplica garantem que você possa continuar a gravar no novo primário assim que a promoção estiver concluída.

No caso de substituições de nó planejadas iniciadas devido a atualizações de manutenção ou de serviço, saiba que as substituições de nó planejadas agora são concluídas enquanto o cluster atende às solicitações de gravação recebidas.

O Multi-AZ em seus clusters do MemoryDB melhora sua tolerância a falhas. Isso é verdade principalmente nos casos em que os nós primários de seu cluster se tornam inacessíveis ou falham por qualquer motivo. O Multi-AZ em clusters do MemoryDB exige que cada fragmento tenha mais de um nó e seja ativado automaticamente.

Tópicos

Cenários de falha com respostas do multi-AZ

Se o Multi-AZ estiver ativo, um nó primário com falha fará o failover para uma réplica disponível. A réplica é sincronizada automaticamente com o log de transações e se torna primária, o que é muito mais rápido do que criar e reprovisionar um novo nó primário. Esse processo normalmente demora apenas alguns segundos até que você possa gravar novamente no cluster.

Quando o Multi-AZ está ativo, o MemoryDB monitora continuamente o estado do nó primário. Se o nó primário falhar, uma das seguintes ações será realizada, dependendo do tipo da falha.

Tópicos

Cenários de falha quando somente o nó primário falha

Se somente o nó primário falhar, uma réplica se tornará automaticamente primária. Depois disso, uma réplica de substituição é criada e provisionada na mesma zona de disponibilidade que o primário com falha.

Quando somente o nó primário falha, o recurso Multi-AZ do MemoryDB faz o seguinte:

1. O nó primário com falha é colocado offline.
2. Uma réplica atualizada se torna automaticamente primária.

   As gravações poderão ser retomadas assim que o processo de failover estiver concluído, normalmente depois de apenas alguns segundos.
3. Uma réplica de substituição é executada e provisionada.

   A réplica de substituição é executada na Zona de disponibilidade em que o nó primário com falha se encontrava, para que a distribuição de nós seja mantida.
4. A réplica é sincronizada com o log de transações.

Para obter informações sobre como encontrar os endpoints de um cluster, consulte os seguintes tópicos:

- Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Cenários de falha quando o nó primário e algumas réplicas falham

Se o primário e pelo menos uma réplica falharem, uma réplica atualizada será promovida a cluster primário. Novas réplicas de leitura também são criadas e provisionadas nas mesmas Zonas de disponibilidade que os nós com falha.

Quando o nó primário e algumas réplicas falham, o Multi-AZ do MemoryDB faz o seguinte:

1. O nó primário com falha e as réplicas com falha são colocadas offline.
2. Uma réplica disponível se tornará o nó primário.

   As gravações poderão ser retomadas assim que o processo de failover estiver concluído, normalmente depois de apenas alguns segundos.
3. Réplicas de substituição são criadas e provisionadas.

   As réplicas de substituição são criadas nas Zonas de disponibilidade dos nós com falha, de modo que a distribuição de nós seja mantida.
4. Todos os nós são sincronizados com o log de transações.

Para obter informações sobre como encontrar os endpoints de um cluster, consulte os seguintes tópicos:

- Localização do endpoint de um cluster do MemoryDB (AWS CLI)
- Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Cenários de falha quando cluster inteiro falha

Se tudo falhar, todos os nós serão recriados e provisionados nas mesmas Zonas de disponibilidade que os nós originais.

Não há perda de dados nesse cenário, pois os dados persistiram no log de transações.

Quando o cluster inteiro falha, o Multi-AZ do MemoryDB faz o seguinte:

1. O nó primário e as réplicas com falha são colocados offline.

2. Um nó primário substituto é criado e provisionado, sincronizado com o log de transações.
3. As réplicas de substituição são criadas e provisionadas, sincronizadas com o log de transações.

   As substituição são criadas nas Zonas de disponibilidade dos nós com falha, de modo que a distribuição de nós seja mantida.

Para obter informações sobre como encontrar os endpoints de um cluster, consulte os seguintes tópicos:

- Localização do endpoint de um cluster do MemoryDB (AWS CLI)
- Localização do endpoint para um cluster do MemoryDB (API do MemoryDB)

Teste do failover automático

Você pode testar o failover automático usando o console do MemoryDB, a AWS CLI e a API do MemoryDB.

Ao testar, observe o seguinte:

- Você pode usar essa operação até cinco vezes em qualquer período de 24 horas.
- Se você chamar essa operação em fragmentos em clusters diferentes, poderá fazer as chamadas simultaneamente.
- Em alguns casos, é possível chamar essa operação várias vezes em diferentes fragmentos no mesmo grupo de cluster do MemoryDB. Nesses casos, a substituição do primeiro nó deve ser concluída antes que uma chamada subsequente possa ser feita.
- Para determinar se a substituição do nó está completa, verifique os eventos usando o console do MemoryDB para Redis, a AWS CLI ou a API do MemoryDB. Procure pelos seguintes eventos relacionados a `FailoverShard`, listados aqui em ordem de ocorrência:

  1. mensagem do cluster: `FailoverShard API called for shard <shard-id>`
  2. mensagem do cluster: `Failover from primary node <primary-node-id> to replica node <node-id> completed`
  3. mensagem do cluster: `Recovering nodes <node-id>`
  4. mensagem do cluster: `Finished recovery for nodes <node-id>`

  Para obter mais informações, consulte as informações a seguir.

  - DescribeEvents na Referência da API do MemoryDB
- Essa API foi projetada para testar o comportamento do seu aplicativo em caso de failover do MemoryDB. Ela não foi projetada para ser uma ferramenta operacional para iniciar um failover a fim de resolver um problema com o cluster. Além disso, sob determinadas condições, como eventos operacionais de grande escala, a AWS pode bloquear essa API.

Tópicos

Teste do failover automático usando o AWS Management Console

Use o procedimento a seguir para testar o failover automático com o console.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Selecione o botão de opção à esquerda do cluster que você deseja testar. Esse cluster deve ter pelo menos um nó de réplica.
3. Na área Details, confirme se esse cluster está habilitado para Multi-AZ. Se o cluster não estiver habilitado para o Multi-AZ, escolha um cluster diferente ou modifique esse cluster para habilitar o Multi-AZ. Para obter mais informações, consulte Modificar um cluster do MemoryDB.
4. Escolha o nome do cluster.
5. Na página Fragmentos e nós, para o fragmento no qual você deseja testar o failover, escolha o nome do fragmento.
6. Para o nó, escolha Failover Primary.
7. Escolha Continue para fazer failover do primário ou Cancel para cancelar a operação e não fazer failover do nó primário.

   Durante o processo de failover, o console continua a mostrar o status do nó como disponível. Para acompanhar o progresso do seu teste de failover, escolha Events no painel de navegação do console. Na guia Eventos, observe os eventos que indicam que o failover foi iniciado (`FailoverShard API called`) e concluído (`Recovery completed`).

Teste do failover automático usando o AWS CLI

É possível testar o failover automático em qualquer cluster habilitado para Multi-AZ usando a AWS CLIoperação failover-shard.

Parâmetros

- `--cluster-name`: obrigatório. O cluster que será testado.
- `--shard-name`: obrigatório. O nome do fragmento no qual você deseja testar o failover automático. Você pode testar um máximo de cinco fragmentos em um período contínuo de 24 horas.

O exemplo a seguir usa o AWS CLI para chamar `failover-shard`do fragmento `0001` no cluster do MemoryDB`my-cluster`.

Para Linux, macOS ou Unix:

```
aws memorydb failover-shard \
   --cluster-name my-cluster \
   --shard-name 0001
```

Para Windows:

```
aws memorydb failover-shard ^
   --cluster-name my-cluster ^
   --shard-name 0001
```

Para acompanhar o progresso do failover, use a operação da AWS CLI `describe-events`.

Retorna a seguinte resposta em JSON:

```
{
    "Events": [
        {
            "SourceName": "my-cluster",
            "SourceType": "cluster",
            "Message": "Failover to replica node my-cluster-0001-002 completed",
            "Date": "2021-08-22T12:39:37.568000-07:00"
        },
        {
            "SourceName": "my-cluster",
            "SourceType": "cluster",
            "Message": "Starting failover for shard 0001",
            "Date": "2021-08-22T12:39:10.173000-07:00"
        }
    ]
}
```

Para obter mais informações, consulte as informações a seguir.

- fragmento de failover
- describe-events

Testar o failover automático usando a API do MemoryDB

O exemplo a seguir chama `FailoverShard` o fragmento `0003` no cluster`memorydb00`.

Example Teste do failover automático

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=FailoverShard
    &ShardName=0003
    &ClusterName=memorydb00
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T192317Z
    &X-Amz-Credential=<credential>
```

Para acompanhar o progresso do failover, use a operação de API do MemoryDB`DescribeEvents`.

Para obter mais informações, consulte as informações a seguir.

- FailoverShard
- DescribeEvents

Alteração do número de réplicas

Você pode aumentar ou diminuir dinamicamente o número de réplicas de leitura no cluster do MemoryDB usando o AWS Management Console, a AWS CLI ou a API do MemoryDB. Todos os fragmentos devem ter o mesmo número de réplicas.

Aumentar o número de réplicas em um cluster

Você pode aumentar o número de réplicas em um cluster do MemoryDB até um máximo de cinco fragmentos. Você pode fazer isso usando o AWS Management Console, a AWS CLI ou a API do MemoryDB.

Tópicos

Usar a AWS Management Console

Para aumentar o número de réplicas em um cluster do MemoryDB (console), consulte Adicição e Remoção de nós de um cluster.

Usar a AWS CLI

Para aumentar o número de réplicas em um cluster do MemoryDB, use o comando `update-cluster` com os seguintes parâmetros:

- `--cluster-name`: obrigatório. Identifica em qual cluster você deseja aumentar o número de réplicas.
- `--replica-configuration`: obrigatório. Permite que você defina o número de réplicas. Para aumentar a contagem de réplicas, defina a propriedade `ReplicaCount` para o número de réplicas que você deseja nesse fragmento ao final desta operação.

Example

O exemplo a seguir aumenta o número de réplicas no cluster `my-cluster` para dois.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --replica-configuration \
        ReplicaCount=2
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --replica-configuration ^
        ReplicaCount=2
```

Retorna a seguinte resposta em JSON:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "updating",
        "NumberOfShards": 1,
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxx.memorydb.us-east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
}
```

Para visualizar os detalhes do cluster atualizado quando seu status mudar de Atualizado para Disponível, use o seguinte comando:

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
    --cluster-name my-cluster
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster
    --show-shard-details
```

Retorna a seguinte resposta em JSON:

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 1,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-003",
```

```
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-22T12:59:31.844000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 3
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```

Para obter mais informações sobre como aumentar o número de réplicas usando a CLI, consulte update-cluster na referência de comandos da AWS CLI.

Usando a API do MemoryDB

Para aumentar o número de réplicas em um fragmento do MemoryDB, use a ação `UpdateCluster` com os seguintes parâmetros:

- `ClusterName`: obrigatório. Identifica em qual cluster você deseja aumentar o número de réplicas.
- `ReplicaConfiguration`: obrigatório. Permite que você defina o número de réplicas. Para aumentar a contagem de réplicas, defina a propriedade `ReplicaCount` para o número de réplicas que você deseja nesse fragmento ao final desta operação.

Example

O exemplo a seguir aumenta o número de réplicas no cluster `sample-cluster` para três. Quando o exemplo é concluído, existem três réplicas em cada fragmento. Esse número se aplica se for um cluster do MemoryDB com um único fragmento ou um cluster do MemoryDB com vários fragmentos.

```
https://memory-db.us-east-1.amazonaws.com/
      ?Action=UpdateCluster
      &ReplicaConfiguration.ReplicaCount=3
      &ClusterName=sample-cluster
      &Version=2021-01-01
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20210802T192317Z
      &X-Amz-Credential=<credential>
```

Para obter mais informações sobre como aumentar o número de réplicas usando a API, consulte UpdateCluster.

Diminuição do número de réplicas em um cluster

Você pode reduzir o número de réplicas em um cluster do MemoryDB. Você pode reduzir o número de réplicas para zero, mas não pode fazer o failover para uma réplica se seu nó primário falhar.

Você pode usar o AWS Management Console, a AWS CLI ou a API do MemoryDB para reduzir o número de réplicas em um cluster.

Tópicos

Usar a AWS Management Console

Para diminuir o número de réplicas em um cluster do MemoryDB (console), consulte Adicição e Remoção de nós de um cluster.

Usar a AWS CLI

Para diminuir o número de réplicas em um cluster do MemoryDB, use o comando `update-cluster` com os seguintes parâmetros:

- `--cluster-name`: obrigatório. Identifica em qual cluster você deseja diminuir o número de réplicas.
- `--replica-configuration`: obrigatório.

  `ReplicaCount` – defina essa propriedade para especificar o número de nós de réplica desejado.

Example

O exemplo a seguir usa `--replica-configuration` a fim de diminuir o número de réplicas no cluster `my-cluster` para o valor especificado.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --replica-configuration \
```

```
        ReplicaCount=1
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --replica-configuration ^
        ReplicaCount=1 ^
```

Retorna a seguinte resposta em JSON:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "updating",
        "NumberOfShards": 1,
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
}
```

Para visualizar os detalhes do cluster atualizado quando seu status mudar de Atualizado para Disponível, use o seguinte comando:

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
```

```
    --cluster-name my-cluster
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster
    --show-shard-details
```

Retorna a seguinte resposta em JSON:

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 1,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
```

```
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```

Para obter mais informações sobre como diminuir o número de réplicas usando a CLI, consulte update-cluster no referência de comandos da AWS CLI.

Usando a API do MemoryDB

Para diminuir o número de réplicas em um cluster do MemoryDB, use a ação `UpdateCluster` com os seguintes parâmetros:

- `ClusterName`: obrigatório. Identifica em qual cluster você deseja diminuir o número de réplicas.
- `ReplicaConfiguration`: obrigatório. Permite que você defina o número de réplicas.

  `ReplicaCount` – defina essa propriedade para especificar o número de nós de réplica desejado.

Example

O exemplo a seguir usa `ReplicaCount` para diminuir o número de réplicas no cluster `sample-cluster` para um. Quando o exemplo é concluído, existe uma réplica em cada fragmento. Esse número se aplica se for um cluster do MemoryDB com um único fragmento ou um cluster do MemoryDB com vários fragmentos.

```
https://memory-db.us-east-1.amazonaws.com/
      ?Action=UpdateCluster
      &ReplicaConfiguration.ReplicaCount=1
      &ClusterName=sample-cluster
      &Version=2021-01-01
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20210802T192317Z
      &X-Amz-Credential=<credential>
```

Para obter mais informações sobre como diminuir o número de réplicas usando a API, consulte UpdateCluster.

Snapshots e restauração

Os clusters do MemoryDB para Redis fazem backup automático dos dados em um log transacional Multi-AZ, mas você pode optar por criar point-in-time instantâneos de um cluster periodicamente ou sob demanda. Esses snapshots podem ser usados para recriar um cluster em um ponto anterior ou para criar um cluster totalmente novo. O snapshot consiste nos metadados do cluster, juntamente com todos os dados do cluster. Todos os snapshots são gravados no Amazon Simple Storage Service (Amazon S3), que fornece armazenamento durável. A qualquer momento, você pode restaurar seus dados criando um novo cluster do MemoryDB e preenchendo-o com dados de um snapshot. Com o MemoryDB, você pode gerenciar instantâneos usando a API AWS Management Console, the AWS Command Line Interface (AWS CLI) e MemoryDB.

Tópicos

- Restrições do snapshot
- Custo do snapshot
- Programação de snapshots automáticos
- Obtenção manual de snapshots
- Criar um snapshot final

Restrições do snapshot

Considere as seguintes restrições ao planejar ou fazer os snapshots:

- Para clusters do MemoryDB, o snapshot e a restauração estão disponíveis para todos os tipos de nós compatíveis.
- Durante qualquer período contíguo de 24 horas, você não pode criar mais de 20 backups manuais por cluster.
- O MemoryDB só suporta a captura de snapshots no nível do cluster. O MemoryDB não suporta a captura de snapshots no nível do fragmento ou do nó.
- Durante o processo de snapshot, não é possível executar outra operação da API ou da CLI no cluster.
- Se você excluir um cluster e solicitar um snapshot final, o MemoryDB sempre usará o snapshot dos nós primários. Isso garante que você capture os dados mais recentes antes que o cluster seja excluído.

Custo do snapshot

Usando o MemoryDB, é possível armazenar gratuitamente um snapshot para cada cluster do MemoryDB ativo. O espaço de armazenamento para snapshots adicionais é cobrado a uma taxa de USD 0,085/GB por mês para todas as regiões da AWS . Não há taxas de transferência de dados para criar um snapshot ou restaurar dados de um snapshot para um cluster do MemoryDB.

Programação de snapshots automáticos

Para qualquer cluster MemoryDB, você pode ativar snapshots automáticos. Quando snapshots automáticos estiverem habilitados, o MemoryDB criará um snapshot do cluster diariamente. Não há impacto no cluster e a alteração é imediata. Para ter mais informações, consulte Restauração a partir de um snapshot.

Ao agendar snapshots automáticos, você deve planejar as seguintes configurações:

- Janela de snapshot – Um período de cada dia em que o MemoryDB começa a criar um snapshot. A duração mínima da janela de backup é de 60 minutos. Você pode definir a janela de snapshot para qualquer momento que lhe seja mais conveniente ou o horário do dia que estiver fora dos períodos de utilização mais alta.

  Se uma janela de snapshot não for especificada, o MemoryDB atribuirá uma automaticamente.
- Snapshot retention limit – o número de dias em que o snapshot é mantido no Amazon S3. Por exemplo, se o limite de retenção for definido como 5, um Snapshot feito hoje será mantido por 5 dias. Quando o limite de retenção expirar, o snapshot será excluído automaticamente.

  O limite máximo de retenção de Snapshots é de 35 dias. Se o limite de retenção de snapshot estiver definido como 0, os snapshots automáticos serão desabilitados para o cluster. Os dados do MemoryDB ainda são totalmente duráveis, mesmo com a captura automática de Snapshots desativada.

Você pode ativar ou desativar instantâneos automáticos ao criar um cluster MemoryDB usando o console MemoryDB, o ou a AWS CLI API MemoryDB. Você pode ativar Snapshots automáticos ao criar um cluster do MemoryDB marcando a caixa Ativar backups automáticos na seção Snapshots. Para obter mais informações, Criação de um cluster do MemoryDB.

Obtenção manual de snapshots

Além dos snapshots automáticos, você pode criar um snapshot manual a qualquer momento. Ao contrário dos snapshots automáticos, que são excluídos automaticamente após um período de retenção especificado, os snapshots manuais não têm um período de retenção após o qual são excluídos automaticamente. Você deve excluir manualmente qualquer snapshot manual. Mesmo que você exclua um cluster ou nó, todos os snapshots manuais desse cluster ou nó serão mantidos. Caso não queira mais manter um snapshot manual, você deverá excluí-lo explicitamente por conta própria.

Snapshots manuais são úteis para testes e arquivamento. Por exemplo, suponha que você tenha desenvolvido um conjunto de dados de linha de base para fins de teste. Você poderá criar um snapshot manual dos dados e restaurá-lo sempre que desejar. Depois de testar um aplicativo que modifica os dados, você poderá redefinir esses dados criando um novo cluster e restaurando a partir do backup de linha de base. Quando o cluster estiver pronto, será possível testar seus aplicativos com base nos dados de linha de base novamente e repetir esse processo com a frequência necessária.

Além de criar diretamente um snapshot manual, você pode criar um snapshot manual de uma das seguintes maneiras:

- Copiar um snapshot: não importa se o backup de origem foi criado automaticamente ou manualmente.
- Criar um snapshot final: cria um snapshot imediatamente antes de excluir um cluster.

Outros tópicos de importância

- Restrições do snapshot
- Custo do snapshot

Você pode criar um instantâneo manual de um nó usando a API AWS Management Console MemoryDB ou a AWS CLI API MemoryDB.

Criação de um snapshot manual (Console)

Para criar um snapshot de um cluster (console)

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Clusters.

   A tela de clusters do MemoryDB é exibida.
3. escolha o botão de opção à esquerda do nome do cluster do MemoryDB do qual deseja fazer backup.
4. Escolha Ações e Tirar snapshot.
5. Na janela Snapshot, digite um nome para seu snapshot na caixa Nome do Snapshot. Recomendamos que o nome indique o cluster do backup e a data e hora de criação do snapshot.

   As restrições de nomenclatura de cluster são as seguintes:

   - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
   - Deve começar com uma letra.
   - Não podem conter dois hifens consecutivos.
   - Não podem terminar com um hífen.
6. Em Criptografia, escolha se deseja usar uma chave de criptografia padrão ou uma chave gerenciada pelo cliente. Para ter mais informações, consulte Criptografia em trânsito (TLS) do MemoryDB.
7. Em Tags, adicione opcionalmente tags para pesquisar e filtrar seus instantâneos ou monitorar seus AWS custos.
8. Selecione Take Snapshot (Fazer snapshot).

   O status do cluster muda para snapshotting. Quando o status retornar para disponível, o snapshot estará concluído.

Criação de um instantâneo manual (AWS CLI)

Para criar um instantâneo manual de um cluster usando o AWS CLI, use a `create-snapshot` AWS CLI operação com os seguintes parâmetros:

- `--cluster-name`: nome do cluster do MemoryDB a ser usado como fonte para o snapshot. Use esse parâmetro ao fazer backup de um cluster do MemoryDB.

  As restrições de nomenclatura de cluster são as seguintes:

  - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
  - Deve começar com uma letra.
  - Não podem conter dois hifens consecutivos.
  - Não podem terminar com um hífen.

- `--snapshot-name` - Nome do snapshot a ser criado.

Tópicos relacionados da

Para obter mais informações, consulte `create-snapshot` na Referência de comandos da AWS CLI .

Criação de um snapshot manual (API do MemoryDB)

Para criar um snapshot manual de um cluster usando a API do MemoryDB, use a operação `CreateSnapshot` da API do MemoryDB com os seguintes parâmetros:

- `ClusterName`: nome do cluster do MemoryDB a ser usado como fonte para o snapshot. Use esse parâmetro ao fazer backup de um cluster do MemoryDB.

  As restrições de nomenclatura de cluster são as seguintes:

  - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
  - Deve começar com uma letra.
  - Não podem conter dois hifens consecutivos.
  - Não podem terminar com um hífen.
- `SnapshotName` - Nome do snapshot a ser criado.

Tópicos relacionados da

Para obter mais informações, consulte CreateSnapshot.

Criar um snapshot final

Você pode criar um instantâneo final usando o console MemoryDB, o ou a AWS CLI API MemoryDB.

Criação de um snapshot final (console)

Você pode criar um snapshot final ao excluir um cluster do MemoryDB usando o console do MemoryDB.

Para criar um snapshot final ao excluir um cluster do MemoryDB, na página de exclusão, escolha Sim e dê um nome ao instantâneo em Etapa 4: excluir um cluster.

Criando um instantâneo final (AWS CLI)

Você pode criar um snapshot final ao excluir um cluster do MemoryDB usando a AWS CLI.

Ao excluir um cluster do MemoryDB

Para criar um instantâneo final ao excluir um cluster, use a `delete-cluster` AWS CLI operação, com os seguintes parâmetros:

- `--cluster-name`: nome do cluster que está sendo excluído.
- `--final-snapshot-name`: nome do snapshot final.

O código a seguir cria o snapshot final `bkup-20210515-final` ao excluir o cluster `myCluster`.

Para Linux, macOS ou Unix:

```
aws memorydb delete-cluster \
        --cluster-name myCluster \
        --final-snapshot-name bkup-20210515-final
```

Para Windows:

```
aws memorydb delete-cluster ^
        --cluster-name myCluster ^
        --final-snapshot-name bkup-20210515-final
```

Para obter mais informações, consulte delete-cluster na Referência de comando da AWS CLI .

Criação de um snapshot final (API do MemoryDB)

Você pode criar um snapshot final ao excluir um cluster do MemoryDB usando a API do MemoryDB.

Ao excluir um cluster do MemoryDB

Para criar um snapshot final, use a operação `DeleteCluster` da API do MemoryDB com os seguintes parâmetros.

- `ClusterName`: nome do cluster que está sendo excluído.
- `FinalSnapshotName`: nome do snapshot.

A seguinte operação da API do MemoryDB cria o snapshot `bkup-20210515-final` ao excluir o cluster `myCluster`.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DeleteCluster
    &ClusterName=myCluster
    &FinalSnapshotName=bkup-20210515-final
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210515T192317Z
    &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte DeleteCluster.

Descrição de snapshots

Os procedimentos a seguir mostram como exibir uma lista dos seus snapshots. Se desejar, você também pode visualizar os detalhes de um snapshot específico.

Descrição de snapshots (console)

Para exibir instantâneos usando o AWS Management Console

1. Faça login no console
2. No painel de navegação à esquerda, selecione Snapshots.
3. Use a pesquisa para filtrar por snapshots manuais, automáticos ou todos os snapshot.
4. Para ver os detalhes de um snapshot em particular, escolha o botão de opções à esquerda do nome do snapshot. Escolha Ações e, em seguida, Visualizar detalhes.
5. Opcionalmente, na página Visualizar detalhes, você pode realizar ações adicionais de snapshot, como copiar, restaurar ou excluir. Você também pode adicionar tags ao snapshot

Descrevendo instantâneos (AWS CLI)

Para exibir uma lista de snapshots e, opcionalmente, detalhes sobre um snapshot específico, use a operação `describe-snapshots` da CLI.

Exemplos

A seguinte operação usa o parâmetro `--max-results` para listar até 20 snapshots associados à sua conta. Omitir o parâmetro `--max-results` lista até 50 snapshots.

```
aws memorydb describe-snapshots --max-results 20
```

A operação a seguir usa o parâmetro `--cluster-name` para listar apenas os snapshots associados ao cluster `my-cluster`.

```
aws memorydb describe-snapshots --cluster-name my-cluster
```

A operação a seguir usa o parâmetro `--snapshot-name` para exibir os detalhes do snapshot `my-snapshot`.

```
aws memorydb describe-snapshots --snapshot-name my-snapshot
```

Para obter mais informações, consulte describe-snapshots.

Descrição de snapshots (API do MemoryDB)

Para exibir uma lista de snapshots, use a operação `DescribeSnapshots`.

Exemplos

A seguinte operação usa o parâmetro `MaxResults` para listar até 20 snapshots associados à sua conta. Omitir o parâmetro `MaxResults` lista até 50 snapshots.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeSnapshots
    &MaxResults=20
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

A operação a seguir usa o parâmetro `ClusterName` para listar todos os snapshots associados ao cluster `MyCluster`.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeSnapshots
    &ClusterName=MyCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

A operação a seguir usa o parâmetro `SnapshotName` para exibir os detalhes para o snapshot `MyBackup`.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeSnapshots
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &SnapshotName=MyBackup
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Para obter mais informações, consulte DescribeSnapshots.

Copiar um snapshot

Você pode fazer uma cópia de qualquer snapshot, seja ele criado automaticamente ou manualmente. Ao copiar um snapshot, a mesma chave de criptografia KMS da origem é usada para o destino, a menos que seja especificamente substituída. Você também pode exportar seu snapshot para poder acessá-lo de fora do MemoryDB. Para obter orientação sobre como exportar o snapshot, consulte Exportação de um snapshot.

Os procedimentos a seguir mostram como copiar um snapshot.

Copiar um snapshot (console)

Para copiar um snapshot (console)

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista dos seus snapshots, no painel de navegação esquerdo, escolha Snapshots.
3. Na lista de snapshots, escolha o botão de opções à esquerda do nome do snapshot que você deseja copiar.
4. Selecione Ações e Copiar.
5. Na página Copiar snapshot, faça o seguinte:

    a. Na caixa Novo nome do snapshot, insira um nome para o novo snapshot.

    b. Deixe a caixa Target S3 Bucket em branco. Esse campo só deve ser usado para exportar o snapshot e requer permissões especiais do S3. Para obter informações sobre como exportar um snapshot, consulte Exportação de um snapshot.

    c. Escolha se deseja usar a chave AWS KMS de criptografia padrão ou usar uma chave personalizada. Para ter mais informações, consulte Criptografia em trânsito (TLS) do MemoryDB.

    d. Opcionalmente, você também pode adicionar tags à cópia do snapshot.

    e. Escolha Copiar.

Copiando um instantâneo (AWS CLI)

Para compartilhar um snapshot, use a operação `copy-snapshot`.

Parâmetros

- `--source-snapshot-name`: nome do snapshot a ser copiado.
- `--target-snapshot-name`: nome da cópia do snapshot.
- `--target-bucket`: reservado para exportação de um snapshot. Não use esse parâmetro ao fazer uma cópia de um snapshot. Para ter mais informações, consulte Exportação de um snapshot.

O exemplo a seguir faz uma cópia de um snapshot automático.

Para Linux, macOS ou Unix:

```
aws memorydb copy-snapshot \
    --source-snapshot-name automatic.my-primary-2021-03-27-03-15 \
    --target-snapshot-name my-snapshot-copy
```

Para Windows:

```
aws memorydb copy-snapshot ^
    --source-snapshot-name automatic.my-primary-2021-03-27-03-15 ^
    --target-snapshot-name my-snapshot-copy
```

Para obter mais informações, consulte copy-snapshot.

Copiar um snapshot (API do MemoryDB)

Para copiar um snapshot, use a operação `copy-snapshot` com os seguintes parâmetros:

Parâmetros

- `SourceSnapshotName`: nome do snapshot a ser copiado.
- `TargetSnapshotName`: nome da cópia do snapshot.
- `TargetBucket`: reservado para exportação de um snapshot. Não use esse parâmetro ao fazer uma cópia de um snapshot. Para ter mais informações, consulte Exportação de um snapshot.

O exemplo a seguir faz uma cópia de um snapshot automático.

Example

```
https://memory-db.us-east-1.amazonaws.com/
```

```
    ?Action=CopySnapshot
    &SourceSnapshotName=automatic.my-primary-2021-03-27-03-15
    &TargetSnapshotName=my-snapshot-copy
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Para obter mais informações, consulte CopySnapshot.

Exportação de um snapshot

O MemoryDB para Redis oferece suporte para a exportação do snapshot do MemoryDB para um bucket do Amazon Simple Storage Service (Amazon S3), que lhe dá acesso de fora do MemoryDB. Os snapshots exportados do MemoryDB são totalmente compatíveis com o Redis de código aberto e podem ser carregados com a versão ou as ferramentas apropriadas do Redis. Você pode exportar um snapshot usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

A exportação de um snapshot pode ser útil se você precisar iniciar um cluster em outra AWS região. Você pode exportar seus dados em uma AWS região, copiar o arquivo.rdb para a nova AWS região e, em seguida, usar esse arquivo.rdb para semear o novo cluster em vez de esperar que o novo cluster seja preenchido por meio do uso. Para obter informações sobre como criar um novo cluster, consulte Propagação de um novo cluster com um snapshot criado externamente. Outra razão pela qual você pode querer exportar os dados do seu cluster é para usar o arquivo .rdb para processamento offline.

Important

- O snapshot do MemoryDB e o bucket do Amazon S3 para o qual você deseja copiá-lo devem estar na mesma região. AWS

  Embora os snapshots copiados para um bucket do Amazon S3 sejam criptografados, recomendamos que você não conceda acesso ao bucket do Amazon S3 no qual deseja armazená-los a outras pessoas.
- A exportação de um snapshot para o Amazon S3 não é compatível com clusters que usam classificação de dados em níveis. Para ter mais informações, consulte Classificação de dados em níveis.

Antes de exportar um snapshot para um bucket do Amazon S3, você deve ter um bucket do Amazon S3 na AWS mesma região do snapshot. Conceder acesso do MemoryDB ao bucket. As duas primeiras etapas mostram como fazer isso.

Warning

Os seguintes cenários expõem seus dados de maneiras que talvez você não queira:

- Quando outra pessoa tiver acesso ao bucket do Amazon S3 para o qual você exportou o snapshot.

  Para controlar o acesso aos seus snapshots, permita acesso ao bucket do Amazon S3 somente àqueles que você deseja que acessem seus dados. Para obter informações sobre como gerenciar o acesso a um bucket do Amazon S3, consulte Gerenciamento do acesso, no Guia do desenvolvedor do Amazon S3.
- Quando outra pessoa tem permissões para usar a operação CopySnapshot da API.

  Usuários ou grupos que possuem permissões para usar a operação `CopySnapshot` da API podem criar seus próprios buckets do Amazon S3 e copiar snapshots para eles. Para controlar o acesso aos seus snapshots, use uma política AWS Identity and Access Management (IAM) para controlar quem tem a capacidade de usar a `CopySnapshot` API. Para obter mais informações sobre como usar o IAM para controlar o uso de operações de API do MemoryDB, consulte Gerenciamento de identidade e acesso no MemoryDB para Redis no Guia do usuário do .

Tópicos

Etapa 1: Crie um bucket do Amazon S3

O procedimento a seguir usa o console do Amazon S3 para criar um bucket do Amazon S3 em que você exportará e armazenará seu snapshot do MemoryDB.

Como criar um bucket do Amazon S3

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Escolha Criar bucket.
3. Em Create a Bucket - Select a Bucket Name and Region, faça o seguinte:

   a. Em Nome do bucket, digite um nome para o bucket do Amazon S3.

b. Na lista de regiões, escolha uma AWS região para seu bucket do Amazon S3. Essa AWS região deve ser a mesma AWS região do snapshot do MemoryDB que você deseja exportar.

c. Escolha Criar.

Para obter mais informações sobre como criar um bucket do Amazon S3, consulte Criação de um bucket, no Guia do usuário do Amazon Simple Storage Service.

Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3

AWS As regiões introduzidas antes de 20 de março de 2019 estão habilitadas por padrão. Você pode começar a trabalhar nessas AWS regiões imediatamente. Regiões adicionadas após 20 de março de 2019 são desabilitadas por padrão. Você deve habilitar ou escolher essas regiões para poder usá-las, conforme descrito em Gerenciamento de regiões da AWS.

Conceda acesso ao MemoryDB ao seu bucket do S3 em uma região AWS

Para criar as permissões adequadas em um bucket do Amazon S3 em uma AWS região, siga as etapas a seguir.

Para conceder acesso ao MemoryDB a um bucket do S3

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Escolha o nome do bucket do Amazon S3 para o qual você deseja copiar o snapshot. Esse deve ser o bucket do S3 que você criou em Etapa 1: Crie um bucket do Amazon S3.
3. Escolha a guia Permissões e, em Permissões, escolha Política de bucket.
4. Atualize a política para conceder ao MemoryDB as permissões necessárias para realizar operações:

   - Adicione `[ "Service" : "`*`region-full-name`*`.memorydb-snapshot.amazonaws.com" ]` a `Principal`.
   - Adicione as seguintes permissões necessárias para exportar um snapshot para o bucket do Amazon S3.
     - `"s3:PutObject"`
     - `"s3:GetObject"`
     - `"s3:ListBucket"`

- "s3:GetBucketAcl"
- "s3:ListMultipartUploadParts"
- "s3:ListBucketMultipartUploads"

Veja a seguir um exemplo de como a política atualizada pode parecer.

```
{
    "Version": "2012-10-17",
    "Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "aws-region.memorydb-snapshot.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::example-bucket",
                "arn:aws:s3:::example-bucket/*"
            ]
        }
    ]
}
```

Etapa 3: exportar um snapshot do MemoryDB

Agora você criou seu bucket do S3 e concedeu permissões ao MemoryDB para acessá-lo. Altere a propriedade do objeto do S3 para ACLs ativadas – preferencialmente o proprietário do bucket. Em seguida, você pode usar o console MemoryDB, a AWS CLI ou a API MemoryDB para exportar seu snapshot para ele. O seguinte pressupõe que você tenha as seguintes permissões do IAM específicas adicionais do S3.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "s3:GetBucketLocation",
   "s3:ListAllMyBuckets",
   "s3:PutObject",
   "s3:GetObject",
   "s3:DeleteObject",
   "s3:ListBucket"
  ],
  "Resource": "arn:aws:s3:::*"
 }]
}
```

Exportação de um snapshot do MemoryDB (console)

O processo a seguir usa o console do MemoryDB para exportar um snapshot para um bucket do Amazon S3 para que você possa acessá-lo de fora do MemoryDB. O bucket do Amazon S3 deve estar na mesma AWS região do snapshot do MemoryDB.

Exportar um snapshot do MemoryDB para um bucket do Amazon S3

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista dos seus snapshots, no painel de navegação esquerdo, escolha Snapshots.
3. Na lista de snapshots, escolha o botão de opções à esquerda do nome snapshot que você deseja exportar.
4. Escolha Copiar.
5. Em Criar uma cópia do backup?, faça o seguinte:

    a. Na caixa Novo nome do snapshot, insira um nome para o snapshot.

        O nome deve ter entre 1 e 1.000 caracteres e pode ser codificado em UTF-8.

        O MemoryDB adiciona um fragmento identificador e `.rdb` ao valor que você inseriu aqui. Por exemplo, se você inserir `my-exported-snapshot`, o MemoryDB criará `my-exported-snapshot-0001.rdb`.

b. Na lista Local do S3 de destino, escolha o nome do bucket do Amazon S3 para o qual você deseja copiar seu snapshot (o bucket que você criou em Etapa 1: Crie um bucket do Amazon S3).

   O local de destino do S3 deve ser um bucket do Amazon S3 na região AWS do snapshot com as seguintes permissões para que o processo de exportação seja bem-sucedido.

   - Acesso ao objeto: Ler e Escrever.
   - Permissões de acesso: Ler.

   Para ter mais informações, consulte Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3.

c. Escolha Copiar.

> **Note**
>
> Se o seu bucket do S3 não tiver as permissões necessárias para que o MemoryDB exporte um snapshot para ele, você receberá uma das seguintes mensagens de erro. Retorne para Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3 a fim de adicionar as permissões especificadas e tente exportar o snapshot novamente.
>
> - O MemoryDB não recebeu permissão READ %s no bucket do S3.
>
>   Solução: adicione permissões de Leitura no bucket.
> - O MemoryDB não recebeu permissões WRITE %s no bucket do S3.
>
>   Solução: adicione permissões de Gravação no bucket.
> - O MemoryDB não recebeu permissões READ_ACP %s no bucket do S3.
>
>   Solução: adicione permissão de acesso de Leitura no bucket.

Se você quiser copiar seu snapshot para outra AWS região, use o Amazon S3 para copiá-lo. Para obter mais informações, consulte Cópia de objetos no Guia do usuário do Amazon Simple Storage Service.

Exportação de um instantâneo do MemoryDB (CLI)AWS

Exporte o snapshot para um bucket do Amazon S3 usando a operação `copy-snapshot` da CLI com os seguintes parâmetros:

Parâmetros

- `--source-snapshot-name`: nome do snapshot a ser copiado.
- `--target-snapshot-name`: nome da cópia do snapshot.

  O nome deve ter entre 1 e 1.000 caracteres e pode ser codificado em UTF-8.

  O MemoryDB adiciona um identificador de fragmento e `.rdb` ao valor que você inseriu aqui. Por exemplo, se você inserir `my-exported-snapshot`, o MemoryDB criará `my-exported-snapshot-0001.rdb`.
- `--target-bucket`: nome do bucket do Amazon S3 no qual você deseja exportar o snapshot. Uma cópia do snapshot é feita no bucket especificado.

  `--target-bucket`Deve ser um bucket do Amazon S3 na AWS região do snapshot com as seguintes permissões para que o processo de exportação seja bem-sucedido.
  - Acesso ao objeto: Ler e Escrever.
  - Permissões de acesso: Ler.

  Para ter mais informações, consulte Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3.

A operação a seguir copia um snapshot para o my-s3-bucket.

Para Linux, macOS ou Unix:

```
aws memorydb copy-snapshot \
    --source-snapshot-name automatic.my-primary-2021-06-27-03-15 \
    --target-snapshot-name my-exported-snapshot \
    --target-bucket my-s3-bucket
```

Para Windows:

```
aws memorydb copy-snapshot ^
    --source-snapshot-name automatic.my-primary-2021-06-27-03-15 ^
    --target-snapshot-name my-exported-snapshot ^
```

```
    --target-bucket my-s3-bucket
```

> **Note**
>
> Se o seu bucket do S3 não tiver as permissões necessárias para que o MemoryDB exporte um snapshot para ele, você receberá uma das seguintes mensagens de erro. Retorne para Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3 a fim de adicionar as permissões especificadas e tente exportar o snapshot novamente.
>
> - O MemoryDB não recebeu permissão READ %s no bucket do S3.
>
>   Solução: adicione permissões de Leitura no bucket.
> - O MemoryDB não recebeu permissões WRITE %s no bucket do S3.
>
>   Solução: adicione permissões de Gravação no bucket.
> - O MemoryDB não recebeu permissões READ_ACP %s no bucket do S3.
>
>   Solução: adicione permissão de acesso de Leitura no bucket.

Para obter mais informações, consulte `copy-snapshot` na Referência de comandos da AWS CLI .

Se você quiser copiar seu snapshot para outra AWS região, use a cópia do Amazon S3. Para obter mais informações, consulte Cópia de objetos no Guia do usuário do Amazon Simple Storage Service.

Exportação de um snapshot do MemoryDB (API do MemoryDB)

Exporte o snapshot para um bucket do Amazon S3 usando a operação `CopySnapshot` da API com os seguintes parâmetros.

Parâmetros

- `SourceSnapshotName`: nome do snapshot a ser copiado.
- `TargetSnapshotName`: nome da cópia do snapshot.

  O nome deve ter entre 1 e 1.000 caracteres e pode ser codificado em UTF-8.

  O MemoryDB adiciona um fragmento identificador e `.rdb` ao valor que você inseriu aqui. Por exemplo, se você inserir `my-exported-snapshot`, receberá `my-exported-snapshot-0001.rdb`.

- `TargetBucket`: nome do bucket do Amazon S3 no qual você deseja exportar o snapshot. Uma cópia do snapshot é feita no bucket especificado.

  `TargetBucket`Deve ser um bucket do Amazon S3 na AWS região do snapshot com as seguintes permissões para que o processo de exportação seja bem-sucedido.

  - Acesso ao objeto: Ler e Escrever.
  - Permissões de acesso: Ler.

  Para ter mais informações, consulte Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3.

O exemplo a seguir faz uma cópia de um snapshot automático para o `my-s3-bucket` do bucket do Amazon S3.

Example

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=CopySnapshot
    &SourceSnapshotName=automatic.my-primary-2021-06-27-03-15
    &TargetBucket=my-s3-bucket
    &TargetSnapshotName=my-snapshot-copy
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

> **Note**
>
> Se o seu bucket do S3 não tiver as permissões necessárias para que o MemoryDB exporte um snapshot para ele, você receberá uma das seguintes mensagens de erro. Retorne para Etapa 2: Conceder acesso do MemoryDB ao bucket do Amazon S3 a fim de adicionar as permissões especificadas e tente exportar o snapshot novamente.
>
> - O MemoryDB não recebeu permissão READ %s no bucket do S3.

Solução: adicione permissões de Leitura no bucket.

- O MemoryDB não recebeu permissões WRITE %s no bucket do S3.

  Solução: adicione permissões de Gravação no bucket.

- O MemoryDB não recebeu permissões READ_ACP %s no bucket do S3.

  Solução: adicione permissão de acesso de Leitura no bucket.

Para obter mais informações, consulte CopySnapshot.

Se você quiser copiar seu snapshot para outra AWS região, use a cópia do Amazon S3 para copiar o snapshot exportado para o bucket do Amazon S3 em outra região. AWS Para obter mais informações, consulte Cópia de objetos no Guia do usuário do Amazon Simple Storage Service.

Restauração a partir de um snapshot

Você pode restaurar os dados de um arquivo de snapshot MemoryDB ou ElastiCache for Redis .rdb em um novo cluster a qualquer momento.

O processo de restauração do MemoryDB para Redis oferece suporte para o seguinte:

- Migração de um ou mais arquivos de snapshot .rdb que você criou para o Redis ElastiCache para um cluster MemoryDB.

  Os arquivos .rdb devem ser colocados no S3 para realizar a restauração.
- Especificar um número de fragmentos no novo cluster que seja diferente do número de fragmentos no cluster que foi usado para criar o arquivo do snapshot.
- Especificar um tipo de nó diferente para o novo cluster, maior ou menor. Se estiver dimensionando para um tipo de nó menor, certifique-se de que o novo tipo de nó tenha memória suficiente para seus dados e a sobrecarga do Redis.
- Configurar os slots do novo cluster do MemoryDB de forma diferente do que no cluster que foi usado para criar o arquivo de snapshot.

> **Important**
>
> - Os clusters do MemoryDB não oferecem suporte a vários bancos de dados. Portanto, ao restaurar para o MemoryDB, a restauração falhará se o arquivo .rdb fizer referência a mais de um banco de dados.
> - Não é possível restaurar um snapshot de um cluster que usa a classificação de dados em níveis (p. ex., tipo de nó r6gd) para um cluster que não usa a classificação de dados em níveis (p. ex., tipo de nó r6g).

A realização de quaisquer alterações ao restaurar um cluster de um snapshot depende das escolhas feitas. Faça essas escolhas na caixa de diálogo Restaurar Cluster ao usar o console do MemoryDB para restaurar. Você faz essas escolhas definindo valores de parâmetros ao usar a API AWS CLI ou MemoryDB para restaurar.

Durante a operação de restauração, o MemoryDB cria o novo cluster e, em seguida, preenche-o com dados do arquivo de snapshot. Quando esse processo é concluído, o cluster está aquecido e pronto para aceitar solicitações.

> **Important**
>
> Antes de prosseguir, verifique se você criou um snapshot do cluster a partir do qual deseja restaurar. Para ter mais informações, consulte Obtenção manual de snapshots.
> Se quiser restaurar a partir de um snapshot criado externamente, consulte Propagação de um novo cluster com um snapshot criado externamente.

Os procedimentos a seguir mostram como restaurar um snapshot em um novo cluster usando o console MemoryDB, o ou a AWS CLI API MemoryDB.

Restauração a partir de um snapshot (Console)

Restaurar um snapshot para um novo cluster (console)

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, escolha Snapshots.
3. Na lista de snapshots, selecione o botão ao lado do nome do snapshot a partir do qual você deseja restaurar.
4. Escolha Ações e, em seguida, escolha Restaurar
5. Em Configuração do cluster, insira o seguinte:
   a. Nome do Cluster – obrigatório. O nome do novo cluster.
   b. Description – opcional. A descrição do novo cluster.
6. Preencha a seção Grupos de sub-redes:
   - Para Grupos de sub-rede, crie um novo grupo de sub-rede ou escolha um existente na lista disponível que você deseja aplicar a esse cluster. Se você estiver criando um novo:
     - Insira um Nome
     - Insira uma Descrição
     - Se você habilitou o Multi-AZ, o grupo de sub-redes deve conter pelo menos duas sub-redes que residem em zonas de disponibilidade diferentes. Para ter mais informações, consulte Sub-redes e grupos de sub-redes.

- Se você estiver criando um novo grupo de sub-redes e não tiver uma VPC existente, você deverá criar uma VPC. Para obter mais informações, consulte O que é a Amazon VPC? no Guia do usuário da Amazon VPC.

7. Complete a seção Configurações de Cluster:

   a. Para compatibilidade com a versão do Redis, aceite o padrão`6.0`.

   b. Em Porta, aceite a porta Redis padrão 6379 ou, se tiver um motivo para usar uma porta diferente, insira o número da porta.

   c. Para Grupo de parâmetros, aceite o `default.memorydb-redis6` do grupo de parâmetros.

      Os grupo de parâmetros controlam os parâmetros de runtime do seu cluster. Para ter mais informações sobre grupos de parâmetros, consulte Parâmetros específicos do Redis.

   d. Para Tipo de nó, escolha um valor para o tipo de nó (junto com o tamanho de memória associado) que você deseja.

      Se você escolher um tipo de nó da família r6gd, a classificação de dados em níveis será ativada automaticamente em seu cluster. Para ter mais informações, consulte Classificação de dados em níveis.

   e. Em Número de fragmentos, escolha o número de fragmentos desejado para este cluster.

      É possível alterar dinamicamente o número de fragmentos no cluster. Para ter mais informações, consulte Escalabilidade de clusters do MemoryDB.

   f. Em Réplicas por fragmento, escolha o número de nós de réplica de leitura desejados em cada fragmento.

      As seguintes restrições existem:

      - Se você tiver o Multi-AZ habilitado, verifique se tem pelo menos uma réplica por fragmento.
      - O número de réplicas é o mesmo para cada fragmento ao criar o cluster usando o console.

   g. Escolha Avançar.

   h. Conclua a seção Configurações avançadas:

      i. Em Grupos de segurança, escolha os grupos de segurança desejados para esse cluster. Um grupo de segurança atua como um firewall para controlar o acesso à rede

ao cluster. É possível usar o grupo de segurança padrão para sua VPC ou criar um novo.

Para obter mais informações sobre grupos de segurança, consulte Grupos de segurança para sua VPC no Guia do usuário da Amazon VPC.

ii. Os dados são criptografados das seguintes maneiras:

- Criptografia em repouso: permite a criptografia de dados armazenados em disco. Para obter mais informações, consulte Criptografia em repouso.

> **Note**
>
> Você tem a opção de fornecer uma chave de criptografia diferente escolhendo a chave AWS KMS gerenciada pelo cliente e escolhendo a chave.

- Criptografia em trânsito: permite a criptografia de dados na conexão. Esta opção está ativada por padrão. Para obter mais informações, consulte criptografia em trânsito.

Se você selecionar nenhuma criptografia, será criada uma lista de controle de acesso aberta chamada “acesso aberto” com um usuário padrão. Para ter mais informações, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

iii. Para Snapshot, especifique opcionalmente um período de retenção de snapshot e uma janela de snapshot. Por padrão, a opção Ativar snapshots automáticos está selecionada.

iv. Para Janela de manutenção, opcionalmente, especifique uma janela de manutenção. A Janela de manutenção é o tempo, geralmente de uma hora de duração, a cada semana quando o MemoryDB agenda a manutenção do sistema para seu cluster. É possível permitir que o MemoryDB escolha o dia e a hora da sua janela de manutenção (Sem preferência) ou é possível escolher o dia, a hora e a duração por conta própria (Especificar janela de manutenção). Se você escolher Especificar janela de manutenção, nas listas, escolha Dia de início, Hora de início e Duração (em horas) para sua janela de manutenção. Todos os horários são em UCT.

Para ter mais informações, consulte Gerenciamento da manutenção.

v. Em Notificações, escolha um tópico existente do Amazon Simple Notification Service (Amazon SNS) ou escolha a entrada de ARN manual e insira o nome de recurso da

Amazon (ARN) do tópico. O Amazon SNS permite que você envie notificações para dispositivos inteligentes conectados à Internet. O padrão é desabilitar notificações. Para obter mais informações, consulte https://aws.amazon.com/sns/.

i. Para Tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar seus clusters ou monitorar seus AWS custos.

j. Revise todas as suas entradas e opções e faça as correções necessárias. Quando estiver pronto, escolha Create cluster para executar seu cluster ou Cancel para cancelar a operação.

Assim que o status do seu cluster for available, você poderá conceder acesso ao EC2 a ele, conectar-se a ele e começar a usá-lo. Para obter mais informações, consulte Etapa 2: autorizar o acesso ao cluster e Etapa 3: Conectar-se ao cluster.

> **Important**
>
> Assim que seu cluster se tornar disponível, você será cobrado por cada hora ou hora parcial em que ele estiver ativo, mesmo que você não o esteja usando ativamente. Para interromper as cobranças aplicáveis para esse cluster, você deve excluí-lo. Consulte Etapa 4: excluir um cluster.

Restaurando a partir de um snapshot (CLI AWS)

Ao usar a operação `create-cluster`, verifique se incluiu o parâmetro `--snapshot-name` ou `--snapshot-arns` para propagar o novo cluster com os dados do snapshot.

Para mais informações, consulte:

- Criação de um cluster (AWS CLI) no Guia do usuário do MemoryDB.
- create-cluster na Referência de Comandos. AWS CLI

Restauração a partir de um snapshot (API do MemoryDB)

Você pode restaurar um snapshot do MemoryDB usando a operação `CreateCluster` da API do MemoryDB.

Ao usar a operação `CreateCluster`, verifique se incluiu o parâmetro `SnapshotName` ou `SnapshotArns` para propagar o novo cluster com os dados do snapshot.

Para mais informações, consulte:

- Criação de um cluster (API do MemoryDB) no Guia do usuário do MemoryDB.
- CreateClusterna referência da API MemoryDB.

Propagação de um novo cluster com um snapshot criado externamente

Quando você cria um novo cluster do MemoryDB, você pode propagá-lo com dados de um arquivo de snapshot .rdb do Redis.

Para semear um novo cluster MemoryDB a partir de um snapshot do MemoryDB ou ElastiCache do Redis, consulte. Restauração a partir de um snapshot

Quando você usa um arquivo .rdb do Redis para propagar um novo cluster, pode fazer o seguinte:

- Especifique o número de fragmentos no novo cluster. Esse número pode ser diferente do número de fragmentos no cluster que foi usado para criar o arquivo de snapshot.
- Especifique um tipo de nó diferente para o novo cluster, maior ou menor que o usado no cluster que fez o snapshot. Se dimensionar para um tipo de nó menor, certifique-se de que o novo tipo de nó tenha memória suficiente para seus dados e a sobrecarga do Redis.

Important

- É necessário garantir que seus dados de snapshot não excedam os recursos do nó.

  Se o snapshot for muito grande, o cluster resultante terá um status de `restore-failed`. Se isso acontecer, você deverá excluir o cluster e começar de novo.

  Para obter uma listagem completa dos tipos e especificações de nós, consulte Parâmetros específicos do tipo de nó do MemoryDB.
- Só é possível criptografar um arquivo .rdb do Redis com a criptografia no lado do servidor do Amazon S3 (SSE-S3). Para obter mais informações, consulte Proteger dados usando a criptografia no lado do servidor.

Etapa 1: Criar snapshot do Redis em cluster externo

Para criar o snapshot para propagar seu cluster do MemoryDB

1. Conecte-se à sua instância do Redis existente.
2. Execute a operação `BGSAVE` ou `SAVE` do Redis para criar um snapshot. Observe onde seu arquivo .rdb está localizado.

`BGSAVE` é assíncrono e não bloqueia outros clientes durante o processamento. Para obter mais informações, consulte BGSAVE no site do Redis.

`SAVE` é síncrono e bloqueia outros processos até terminar. Para obter mais informações, consulte SAVE no site do Redis.

Para obter informações adicionais sobre como criar um snapshot, consulte o tópico sobre Persistência do Redis no site do Redis.

Etapa 2: criar um bucket e uma pasta no Amazon S3

Quando você tiver criado o arquivo de snapshot, precisará carregá-lo em uma pasta dentro de um bucket do Amazon S3. Para fazer isso, primeiro você deve ter um bucket do Amazon S3 e uma pasta dentro desse bucket. Se você já possui um bucket do Amazon S3 e uma pasta com as permissões apropriadas, poderá pular para Etapa 3: carregar seu snapshot no Amazon S3.

Como criar um bucket do Amazon S3

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Para criar um bucket do Amazon S3, siga as instruções em Criação de um bucket no Guia do usuário do Amazon Simple Storage Service.

   O nome do bucket do Amazon S3 deve estar em conformidade com o DNS. Caso contrário, o MemoryDB não poderá acessar seu arquivo de backup. As regras para a conformidade de DNS são:

   - Os nomes devem ter no mínimo 3 e no máximo 63 caracteres de extensão.
   - Os nomes devem ser uma série de um ou mais rótulos separados por um ponto (.) em que cada rótulo:
     - Começa com uma letra minúscula ou um número.
     - Termina com uma letra minúscula ou um número.
     - Contém somente letras minúsculas, números e traços.
   - Os nomes não podem ser formatado como um endereço IP (por exemplo, 192.0.2.0).

É altamente recomendável que você crie seu bucket do Amazon S3 na mesma AWS região do seu novo cluster MemoryDB. Essa abordagem garante a maior velocidade de transferência de dados quando o MemoryDB lê seu arquivo .rdb do Amazon S3.

> **Note**
>
> Para manter seus dados da forma mais segura possível, restrinja ao máximo as permissões em seu bucket do Amazon S3. Ao mesmo tempo, as permissões ainda precisam permitir que o bucket e seu conteúdo seja usado para propagar o novo cluster do MemoryDB.

Para adicionar uma pasta a um bucket do Amazon S3

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Escolha o nome do bucket para o qual deseja fazer upload do arquivo .rdb.
3. Selecione Criar pasta.
4. Insira um nome para a nova pasta.
5. Escolha Salvar.

   Anote o nome do bucket e o nome da pasta.

Etapa 3: carregar seu snapshot no Amazon S3

Agora, faça upload do arquivo .rdb criado em Etapa 1: Criar snapshot do Redis em cluster externo. Carregue-o no bucket e na pasta do Amazon S3 que você criou em Etapa 2: criar um bucket e uma pasta no Amazon S3. Para obter mais informações sobre essa tarefa, consulte Upload de objetos. Entre as etapas 2 e 3, escolha o nome da pasta que você criou.

Para carregar seu arquivo .rdb em uma pasta do Amazon S3

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Escolha o nome do bucket do Amazon S3 criado na Etapa 2.
3. Escolha o nome da pasta que você criou na Etapa 2.

4. Escolha Carregar.
5. Escolha Adicionar arquivos.
6. Navegue para encontrar um ou mais arquivos que deseja carregar e depois escolha esses arquivos. Para escolher vários arquivos, mantenha pressionada a tecla Ctrl enquanto escolhe o nome de cada arquivo.
7. Escolha Open (Abrir).
8. Confirme se o arquivo ou arquivos corretos estão listados na página Upload e, em seguida, selecione Upload.

Anote o caminho para o arquivo .rdb. Por exemplo, se o nome do bucket for `myBucket` e o caminho for `myFolder/redis.rdb`, insira `myBucket/myFolder/redis.rdb`. Você precisa desse caminho para propagar o novo cluster com os dados neste snapshot.

Para obter mais informações, consulte as Regras para nomear buckets no Guia do usuário do Amazon Simple Storage Service.

Etapa 4: Conceder ao MemoryDB acesso de leitura ao arquivo .rdb

AWS As regiões introduzidas antes de 20 de março de 2019 estão habilitadas por padrão. Você pode começar a trabalhar nessas AWS regiões imediatamente. Regiões adicionadas após 20 de março de 2019 são desabilitadas por padrão. Você deve habilitar ou escolher essas regiões para poder usá-las, conforme descrito em Gerenciamento de regiões da AWS.

Conceda ao MemoryDB acesso de leitura ao arquivo .rdb

Conceder ao MemoryDB acesso de leitura ao arquivo de snapshot

1. Faça login no AWS Management Console e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Escolha o nome do bucket do S3 que contém seu arquivo .rdb.
3. Escolha o nome da pasta que contém seu arquivo .rdb.
4. Escolha o nome do seu arquivo de snapshot .rdb. O nome do arquivo selecionado aparece acima das guias na parte superior da página.
5. Escolha a guia Permissões.
6. Em Permissões, escolha Política de bucket e, em seguida, Editar.

7. Atualize a política para conceder ao MemoryDB as permissões necessárias para realizar operações:

   - Adicione `[ "Service" : "region-full-name.memorydb-snapshot.amazonaws.com" ]` a `Principal`.
   - Adicione as seguintes permissões necessárias para exportar um snapshot para o bucket do Amazon S3:
     - `"s3:GetObject"`
     - `"s3:ListBucket"`
     - `"s3:GetBucketAcl"`

   Veja a seguir um exemplo de como a política atualizada pode parecer.

```
{
    "Version": "2012-10-17",
    "Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "us-east-1.memorydb-snapshot.amazonaws.com"
            },
            "Action": [
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::example-bucket",
                "arn:aws:s3:::example-bucket/snapshot1.rdb",
                "arn:aws:s3:::example-bucket/snapshot2.rdb"
            ]
        }
    ]
}
```

8. Escolha Salvar.

Etapa 5: propagar o cluster do MemoryDB com os dados do arquivo .rdb

Agora, você está pronto para criar um cluster do MemoryDB e propagá-lo com dados do arquivo .rdb. Para criar o cluster, siga as instruções em Criação de um cluster do MemoryDB.

O método que você usa para dizer ao MemoryDB onde encontrar o snapshot do Redis que você fez o upload no Amazon S3 depende do método usado para criar o cluster:

Propagar o cluster do MemoryDB com os dados do arquivo .rdb

- Como usar o console do MemoryDB

  Depois de escolher o mecanismo Redis, expanda a seção Advanced Redis settings e localize Import data to cluster. Na caixa Propagar local S3 do arquivo RDB, digite o caminho do Amazon S3 para o(s) arquivo(s). Se você tiver vários arquivos .rdb, digite o caminho para cada um em uma lista separada por vírgulas. O caminho do Amazon S3 parece-se com *myBucket*/*myFolder*/*myBackupFilename*.rdb.
- Usando o AWS CLI

  Se você usar a operação `create-cluster` ou `create-cluster`, use o parâmetro `--snapshot-arns` para especificar um ARN totalmente qualificado para cada arquivo .rdb. Por exemplo, `arn:aws:s3:::`*myBucket*/*myFolder*/*myBackupFilename*`.rdb`. O ARN deve ser resolvido para os arquivos de snapshot que você armazenou no Amazon S3.
- Usando a API do MemoryDB

  Se você usar a operação `CreateCluster` ou `CreateCluster` da API do MemoryDB, use o parâmetro `SnapshotArns` para especificar um ARN totalmente qualificado para cada arquivo .rdb. Por exemplo, `arn:aws:s3:::`*myBucket*/*myFolder*/*myBackupFilename*`.rdb`. O ARN deve ser resolvido para os arquivos de snapshot que você armazenou no Amazon S3.

Durante o processo de criação do seu cluster, os dados no seu snapshot são gravados no cluster. Você pode monitorar o progresso visualizando as mensagens de eventos do MemoryDB. Para fazer isso, consulte o console do MemoryDB e escolha Eventos. Você também pode usar a interface de linha de comando do AWS MemoryDB ou a API do MemoryDB para obter mensagens de eventos.

Marcação de snapshots

Você pode atribuir os próprios metadados a cada snapshot na forma de tags. As tags permitem categorizar seus snapshots de diferentes formas, como por exemplo, por finalidade, por proprietário ou por ambiente. Isso é útil quando você tem muitos recursos do mesmo tipo. É possível identificar rapidamente um recurso específico baseado nas tags que você atribuiu a ele. Para ter mais informações, consulte Recursos que podem ser marcados.

As etiquetas de alocação de custos são um meio de rastrear seus custos em vários AWS serviços, agrupando suas despesas em faturas por valores de etiquetas. Para saber mais sobre alocação de custos, consulte Usar tags de alocação de custos.

Usando o console MemoryDB, a API ou MemoryDB AWS CLI, você pode adicionar, listar, modificar, remover ou copiar tags de alocação de custos em seus snapshots. Para ter mais informações, consulte Monitoramento de custos com tags de alocação de custos.

Excluir um snapshot

Um snapshot automático é excluído automaticamente quando o limite de retenção expira. Se você excluir um cluster, todos os seus snapshots automáticos também serão excluídos.

O MemoryDB fornece uma operação de API de exclusão que permite excluir um snapshot a qualquer momento, independentemente de o snapshot ter sido criado automaticamente ou manualmente. Como os snapshots manuais não possuem um limite de retenção, a exclusão manual é a única maneira de removê-los.

Você pode excluir um snapshot usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

Excluir um snapshot (console)

O procedimento a seguir exclui um snapshot usando o console do MemoryDB.

Para excluir um snapshot

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Snapshots.

   A tela de snapshots aparece com uma lista dos seus instantâneos.
3. Escolha o botão de opções à esquerda do nome do snapshot que você deseja excluir.
4. Escolha Ações e, em seguida, escolha Excluir.
5. Se você quiser excluir esse snapshot, insira `delete` na caixa de texto e escolha Excluir. Escolha Cancelar para cancelar a exclusão. O status muda para deleting.

Excluindo um instantâneo (CLI AWS)

Use a AWS CLI operação delete-snapshot com o parâmetro a seguir para excluir um snapshot.

- `--snapshot-name`: o nome do snapshot a ser excluído.

O código a seguir exclui o snapshot `myBackup`.

```
aws memorydb delete-snapshot --snapshot-name myBackup
```

Para obter mais informações, consulte delete-snapshot na Referência de comandos da AWS CLI .

Excluindo um instantâneo (API do MemoryDB)

Use a operação `DeleteSnapshot` da API com o seguinte parâmetro para excluir um snapshot.

- `SnapshotName`: o nome do snapshot a ser excluído.

O código a seguir exclui o snapshot `myBackup`.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DeleteSnapshot
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &SnapshotName=myBackup
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte DeleteSnapshot.

Escalabilidade

A quantidade de dados que o seu aplicativo precisa processar é raramente estática. Ela aumenta e diminui à medida que sua empresa cresce ou passa por flutuações normais na demanda. Se autogerenciar seus aplicativos, você precisará provisionar hardware suficiente para seus picos de demanda, o que pode ser caro. Usando o MemoryDB para Redis, você pode escalar para atender à demanda atual, pagando apenas pelo que você usa.

O conteúdo a seguir ajuda a encontrar o tópico correto para as ações de escalabilidade que você deseja executar.

Escalabilidade do MemoryDB

Ação	MemoryDB	
Aumento de escala	Refragmentação online e rebalanceamento de fragmentos do MemoryDB	

Ação	MemoryDB	
Alteração nos tipos de nó	Escalabilidade vertical online com modificação do tipo de nó	
Alteração no número de fragmentos	Escalabilidade de clusters do MemoryDB	

Escalabilidade de clusters do MemoryDB

À medida que a demanda dos clusters muda, convém melhorar a performance ou reduzir os custos alterando o número de fragmentos no cluster do MemoryDB. Recomendamos o uso da escalabilidade horizontal online para esse ajuste, pois permite que o seu cluster continue a atender às solicitações durante o processo de escalabilidade.

As condições sob as quais você pode decidir redimensionar seu cluster incluem o seguinte:

- Uso intenso de memória:

  Se os nós no cluster estão sob uso intenso da memória, você pode optar por aumentar a escala e ter mais recursos para melhor armazenar dados e atender a solicitações.

  Você pode determinar se os nós estão sob uso intenso da memória monitorando as seguintes métricas: FreeableMemory, SwapUsage e BytesUsedForMemoryDB.
- CPU ou gargalo de rede:

  Se os problemas de latência/throughput estão enfraquecendo seu cluster, pode ser necessário aumentar a escala para resolvê-los.

  Você pode monitorar seus níveis de latência e throughput monitorando as seguintes métricas: CPUUtilization, NetworkBytesIn, NetworkBytesOut, CurrConnectionse NewConnections.
- Seu cluster está acima da escala:

  A demanda atual no cluster permite que haja uma redução na escala sem afetar o desempenho e proporcionando corte de custos.

  Você pode monitorar o uso do seu cluster para determinar se pode reduzir a escala horizontalmente com segurança usando as seguintes métricas: FreeableMemory, SwapUseage, BytesUsedForMemoryDB, CPUUtilization, NetworkBytesIn, NetworkBytesOut, CurrConnections e NewConnections.

Impacto da escalabilidade no desempenho

Quando você altera a escala usando o processo offline, seu cluster fica offline para uma parte significativa do processo e, por conseguinte, não é capaz de atender a solicitações. Quando você altera a escala usando o método online, como a escalabilidade é uma operação com uso intensivo de computação, há queda no desempenho, mas ainda assim seu cluster continua atendendo a

solicitações durante a operação de escalabilidade. O quanto o desempenho é afetado depende do seu uso normal da CPU e dos seus dados.

Existem duas maneiras de escalar o cluster do MemoryDB: escalabilidade horizontal e vertical.

- A escalabilidade horizontal permite alterar o número de fragmentos no cluster adicionando ou removendo fragmentos. O processo de reestilhaçamento online permite expandir/reduzir enquanto o cluster continua veiculando solicitações de entrada.
- Escalabilidade vertical — altere o tipo de nó para redimensionar o cluster. O processo de escalabilidade vertical online permite expandir/reduzir enquanto o cluster continua veiculando solicitações de entrada.

Se estiver reduzindo o tamanho e a capacidade de memória do cluster, reduzindo ou expandindo, garanta que a nova configuração tenha memória suficiente para seus dados e a sobrecarga do Redis.

Refragmentação offline e rebalanceamento de fragmentos do MemoryDB

A principal vantagem de obter a reconfiguração de fragmentos offline é que você pode fazer mais do que simplesmente adicionar ou remover fragmentos de seu cluster. Quando você faz o reestilhaçamento offline, além de alterar o número de fragmentos no seu cluster, é possível fazer o seguinte:

- Alterar o tipo de nó do seu cluster.
- Fazer o upgrade para uma versão mais recente do mecanismo.

> **Note**
>
> A refragmentação offline não é compatível com clusters que tenham a classificação de dados em níveis ativada. Para obter mais informações, consulte Classificação de dados em níveis..

A principal desvantagem da reconfiguração de fragmentos offline é que o cluster fica offline começando com a parte de restauração do processo e continua até você atualizar os endpoints no aplicativo. O tempo em que o cluster fica offline depende da quantidade de dados no seu cluster.

Para reconfigurar o cluster de fragmentos do MemoryDB offline

1. Crie um snapshot manual do cluster do MemoryDB existente. Para obter mais informações, consulte Obtenção manual de snapshots.
2. Crie um novo cluster fazendo a restauração a partir do snapshot. Para obter mais informações, consulte Restauração a partir de um snapshot.
3. Atualize os endpoints no seu aplicativo para os endpoints do novo cluster. Para obter mais informações, consulte Encontrar endpoints de conexão.

Refragmentação online e rebalanceamento de fragmentos do MemoryDB

Ao optar pela refragmentação e rebalanceamento de fragmentos online com o MemoryDB, você pode escalar seu MemoryDB de forma dinâmica sem tempo de inatividade. Essa abordagem significa que seu cluster pode continuar atendendo a solicitações mesmo durante a escalabilidade ou o rebalanceamento.

Você pode fazer o seguinte:

- Aumentar a escala horizontalmente – aumente a capacidade de leitura e gravação adicionando fragmentos ao seu cluster do MemoryDB.

  Se você adicionar um ou mais fragmentos ao cluster, o número de nós em cada novo fragmento será o mesmo que o número de nós no menor dos fragmentos existentes.
- Reduzir a escala horizontalmente – reduza a capacidade de leitura e gravação e, por conseguinte, os custos, removendo fragmentos do cluster do MemoryDB.

Atualmente, as seguintes limitações se aplicam à refragmentação online do MemoryDB:

- Há limitações em relação a slots ou espaços de chave e itens grandes:

  Se qualquer uma das chaves em um fragmento contiver um item grande, essa chave não será migrada para um novo fragmento durante o aumento da escala ou o rebalanceamento. Essa funcionalidade pode resultar em fragmentos desbalanceados.

  Se qualquer uma das chaves em um fragmento contiver um item grande (itens maiores do que 256 MB após a serialização), o fragmento não será excluído na redução da escala. Essa funcionalidade pode resultar na não exclusão de alguns fragmentos.

- Ao aumentar a escala horizontalmente, o número de nós em novos fragmentos fica igual ao número de nós nos fragmentos existentes.

Para obter mais informações, consulte Práticas recomendadas: redimensionamento online de clusters.

Você pode escalar horizontalmente ou rebalancear seu cluster do MemoryDB usando o AWS Management Console, a AWS CLI e a API do MemoryDB.

Adição de fragmentos com refragmentação online

Você pode adicionar fragmentos ao seu cluster do MemoryDB usando o AWS Management Console, a AWS CLI, ou a API do MemoryDB.

Adição de fragmentos (console)

Você pode usar o AWS Management Console para adicionar um ou mais fragmentos ao seu cluster do MemoryDB. O procedimento a seguir descreve o processo.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista de clusters, escolha o nome do cluster a partir do qual você deseja adicionar um fragmento.
3. Na guia Fragmentos e nós, escolha Adicionar/Excluir fragmentos
4. Em Novo número de fragmentos, insira o número de fragmentos que você deseja.
5. Escolha Confirmar para manter as alterações ou Cancelar para descartá-las.

Adição de fragmentos (AWS CLI)

O processo a seguir descreve como reconfigurar os fragmentos no seu cluster do MemoryDB adicionando fragmentos com a AWS CLI.

Use os parâmetros a seguir com `update-cluster`.

Parâmetros

- `--cluster-name`: obrigatório. Especifica em qual cluster a operação de reconfiguração de fragmento será executada.
- `--shard-configuration`: obrigatório. Permite que você defina o número de fragmentos.

- `ShardCount` – defina essa propriedade para especificar o número de fragmentos que você deseja.

Example

O exemplo a seguir modifica o número de fragmentos no cluster `my-cluster` para 2.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --shard-configuration \
        ShardCount=2
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --shard-configuration ^
        ShardCount=2
```

Retorna a seguinte resposta em JSON:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "updating",
        "NumberOfShards": 2,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
```

```
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
}
```

Para visualizar os detalhes do cluster atualizado quando seu status mudar de Atualizado para Disponível, use o seguinte comando:

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
    --cluster-name my-cluster
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster
    --show-shard-details
```

Retorna a seguinte resposta em JSON:

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 2,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-8191",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
```

```
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                },
                {
                    "Name": "0002",
                    "Status": "available",
                    "Slots": "8192-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0002-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-22T14:26:18.693000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0002-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-22T14:26:18.765000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
```

```
                            "Port": 6379
                        }
                    }
                ],
                "NumberOfNodes": 2
            }
        ],
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "ACLName": "my-acl",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
  ]
}
```

Para obter mais informações, consulte update-cluster na referência de comandos da AWS CLI.

Adição de fragmentos (API do MemoryDB)

Você pode usar a API do MemoryDB para reconfigurar os fragmentos no cluster do MemoryDB online usando a operação `UpdateCluster`.

Use os parâmetros a seguir com `UpdateCluster`.

Parâmetros

- `ClusterName`: obrigatório. Especifica em qual cluster a operação de reconfiguração de fragmento será executada.

- `ShardConfiguration`: obrigatório. Permite que você defina o número de fragmentos.
  - `ShardCount` – defina essa propriedade para especificar o número de fragmentos que você deseja.

Para obter mais informações, consulte UpdateCluster.

Remoção de fragmentos com refragmentação online

Você pode remover fragmentos do seu cluster do MemoryDB usando o AWS Management Console, a AWS CLI, ou a API do MemoryDB.

Remoção de fragmentos (console)

O processo a seguir descreve como reconfigurar os fragmentos em seu cluster do MemoryDB removendo os fragmentos usando o AWS Management Console.

> **Important**
>
> Antes de remover os fragmentos do seu cluster, o MemoryDB verifica se todos os seus dados cabem nos demais fragmentos. Se os dados couberem, os fragmentos serão excluídos do cluster como solicitado. Se os dados não couberem nos fragmentos restantes, o processo será encerrado e o cluster será deixado com a mesma configuração do fragmento anterior à solicitação.

Você pode usar a AWS Management Console para remover um ou mais fragmentos de seu cluster do MemoryDB. Não é possível remover todos os fragmentos de um cluster. Em vez disso, você deve excluir o cluster. Para obter mais informações, consulte Etapa 4: excluir um cluster. O procedimento a seguir descreve o processo para remover um ou mais fragmentos.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista de clusters, escolha o nome do cluster do qual você deseja remover um fragmento.
3. Na guia Fragmentos e nós, escolha Adicionar/Excluir fragmentos
4. Em Novo número de fragmentos, insira o número de fragmentos que você deseja (com um mínimo de 1).
5. Escolha Confirmar para manter as alterações ou Cancelar para descartá-las.

Remoção de fragmentos (AWS CLI)

O processo a seguir descreve como reconfigurar os fragmentos em seu cluster do MemoryDB removendo os fragmentos usando o AWS CLI.

> **Important**
>
> Antes de remover os fragmentos do seu cluster, o MemoryDB verifica se todos os seus dados cabem nos demais fragmentos. Se os dados couberem, os fragmentos serão excluídos do cluster como solicitado e seus espaços de chave serão mapeados para os fragmentos restantes. Se os dados não couberem nos fragmentos restantes, o processo será encerrado e o cluster permanecerá com a mesma configuração de fragmentos anterior à solicitação.

Você pode usar a AWS CLI para remover um ou mais fragmentos de seu cluster do MemoryDB. Não é possível remover todos os fragmentos de um cluster. Em vez disso, você deve excluir o cluster. Para obter mais informações, consulte Etapa 4: excluir um cluster.

Use os parâmetros a seguir com `update-cluster`.

Parâmetros

- `--cluster-name`: obrigatório. Especifica em qual cluster a operação de reconfiguração de fragmento será executada.
- `--shard-configuration`: obrigatório. Permite que você defina o número de fragmentos usando a propriedade `ShardCount`:

  `ShardCount` – defina essa propriedade para especificar o número de fragmentos que você deseja.

Example

O exemplo a seguir modifica o número de fragmentos no cluster `my-cluster` para 2.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
```

```
    --shard-configuration \
        ShardCount=2
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --shard-configuration ^
        ShardCount=2
```

Retorna a seguinte resposta em JSON:

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "updating",
        "NumberOfShards": 2,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "DataTiering": "false",
        "AutoMinorVersionUpgrade": true
    }
}
```

Para visualizar os detalhes do cluster atualizado quando seu status mudar de Atualizado para Disponível, use o seguinte comando:

Para Linux, macOS ou Unix:

```
aws memorydb describe-clusters \
    --cluster-name my-cluster
    --show-shard-details
```

Para Windows:

```
aws memorydb describe-clusters ^
    --cluster-name my-cluster
    --show-shard-details
```

Retorna a seguinte resposta em JSON:

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 2,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-8191",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
```

```
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                },
                {
                    "Name": "0002",
                    "Status": "available",
                    "Slots": "8192-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0002-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-22T14:26:18.693000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0002-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-22T14:26:18.765000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
```

```
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```

Para obter mais informações, consulte update-cluster na referência de comandos da AWS CLI.

Remoção de fragmentos (API do MemoryDB)

Você pode usar a API do MemoryDB para reconfigurar os fragmentos no cluster do MemoryDB online usando a operação `UpdateCluster`.

O processo a seguir descreve como reconfigurar os fragmentos em seu cluster do MemoryDB removendo fragmentos usando a API do MemoryDB.

> **Important**
>
> Antes de remover fragmentos do seu cluster, o MemoryDB verifica se todos os seus dados cabem nos demais fragmentos. Se os dados couberem, os fragmentos serão excluídos do cluster como solicitado e seus espaços de chave serão mapeados para os fragmentos restantes. Se os dados não couberem nos fragmentos restantes, o processo será encerrado e o cluster permanecerá com a mesma configuração de fragmentos anterior à solicitação.

Você pode usar a API do MemoryDB para remover um ou mais fragmentos de seu cluster do MemoryDB. Não é possível remover todos os fragmentos de um cluster. Em vez disso, você deve excluir o cluster. Para obter mais informações, consulte Etapa 4: excluir um cluster.

Use os parâmetros a seguir com `UpdateCluster`.

Parâmetros

- `ClusterName`: obrigatório. Especifica em qual cluster a operação de reconfiguração de fragmento será executada.
- `ShardConfiguration`: obrigatório. Permite que você defina o número de fragmentos usando a propriedade `ShardCount`:

  `ShardCount` – defina essa propriedade para especificar o número de fragmentos que você deseja.

Escalabilidade vertical online com modificação do tipo de nó

Usando a escalabilidade vertical online com o MemoryDB, você poderá escalar dinamicamente os clusters com tempo de inatividade mínimo. Isso permite que o cluster veicule solicitações mesmo ao ser escalado.

> **Note**
>
> Não há compatibilidade com escalabilidade entre um cluster de classificação de dados em níveis (p. ex., um cluster que use um tipo de nó r6gd) e um cluster sem classificação de dados em níveis (p. ex., um cluster que use um tipo de nó r6g). Para obter mais informações, consulte Classificação de dados em níveis.

Você pode fazer o seguinte:

- Aumentar a escala verticalmente – aumente a capacidade de leitura e gravação ajustando o tipo de nó do cluster MemoryDB para usar um tipo de nó maior.

  O MemoryDB redimensiona dinamicamente o cluster ao permanecer online e veicular solicitações.
- Redução de escala vertical: reduza a capacidade de leitura e gravação ajustando o tipo de nó para usar um nó menor. Novamente, o MemoryDB redimensiona dinamicamente o cluster ao permanecer online e veicular solicitações. Nesse caso, você reduz os custos diminuindo o nó.

> **Note**
>
> Os processos de expansão e redução dependem da criação de clusters com tipos de nó recém-selecionados e da sincronização dos novos nós com os anteriores. Para garantir um fluxo suave de expansão/redução, faça o seguinte:
>
> - Embora o processo de escalabilidade vertical seja desenvolvido para permanecer totalmente online, ele depende da sincronização dos dados entre o nó antigo e o novo nó. Recomendamos iniciar a expansão/redução no horário em que você acredita que o tráfego de dados seja mínimo.
> - Teste o comportamento de seu aplicativo durante a escalabilidade em um ambiente de preparação, se possível.

Aumento de escala vertical online

Tópicos

- Aumento de escala vertical de clusters do MemoryDB (console)
- Aumento de escala vertical de clusters do MemoryDB (AWS CLI)
- Aumento de escala vertical de clusters do MemoryDB (API do MemoryDB)

Aumento de escala vertical de clusters do MemoryDB (console)

O procedimento a seguir descreve como aumentar a escala verticalmente de um cluster do MemoryDB usando o AWS Management Console. Durante esse processo, o cluster do MemoryDB continuará a atender solicitações com tempo de inatividade mínimo.

Para aumentar a escala verticalmente de um cluster (console)

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista de clusters, escolha o cluster.
3. Escolha Ações e Modificar.
4. Na caixa de diálogo Modificar cluster:
    - Na lista Node type, escolha o tipo de nó a partir do qual você deseja escalar. Para expandir, selecione um tipo de nó maior do que o nó existente.

5. Escolha Salvar alterações.

   O status do cluster muda para Modificação. Quando o status mudar para available, a modificação estará completa, e você poderá começar a usar o novo cluster.

Aumento de escala vertical de clusters do MemoryDB (AWS CLI)

O procedimento a seguir descreve como aumentar a escala verticalmente de um cluster do MemoryDB usando o AWS CLI. Durante esse processo, o cluster do MemoryDB continuará a atender solicitações com tempo de inatividade mínimo.

Como aumentar a escala verticalmente de um cluster MemoryDB (AWS CLI)

1. Determine os tipos de nó para os quais você pode expandir executando o comando AWS CLI da `list-allowed-node-type-updates` com o seguinte parâmetro.

   Para Linux, macOS ou Unix:

   ```
   aws memorydb list-allowed-node-type-updates \
        --cluster-name my-cluster-name
   ```

   Para Windows:

   ```
   aws memorydb list-allowed-node-type-updates ^
        --cluster-name my-cluster-name
   ```

   A saída do comando acima é semelhante a esta (formato JSON).

   ```
   {
        "ScaleUpNodeTypes": [
            "db.r6g.2xlarge",
            "db.r6g.large"
        ],
        "ScaleDownNodeTypes": [
            "db.r6g.large"
        ],
   }
   ```

   Para obter mais informações, consulte list-allowed-node-type-updates na Referência da AWS CLI.

2. Modifique seu cluster para aumentar a escala verticalmente para o novo tipo de nó maior usando o comando `update-cluster` da AWS CLI com os seguintes parâmetros.

   - `--cluster-name` – o nome do cluster de cache que você está aumentando.
   - `--node-type` – o novo tipo de nó para o qual você deseja escalar o cluster. Esse valor deve ser um dos tipos de nós retornados pelo comando `list-allowed-node-type-updates` na etapa 1.

   Para Linux, macOS ou Unix:

   ```
   aws memorydb update-cluster  \
    --cluster-name my-cluster \
    --node-type db.r6g.2xlarge
   ```

   Para Windows:

   ```
   aws memorydb update-cluster ^
        --cluster-name my-cluster ^
        --node-type db.r6g.2xlarge ^
   ```

   Para obter mais informações, consulte update-cluster.

Aumento de escala vertical de clusters do MemoryDB (API do MemoryDB)

O processo a seguir escala seu cluster do tipo de nó atual para um novo tipo de nó maior usando a API do MemoryDB. Durante esse processo, o MemoryDB atualiza as entradas do DNS para que elas apontem para os novos nós. Você pode escalar clusters habilitados para failover automático enquanto o cluster permanece online e atende às solicitações recebidas.

O tempo necessário para aumentar a escala verticalmente até um tipo de nó maior varia, dependendo do tipo de nó e da quantidade de dados no seu cluster atual.

Como aumentar a escala verticalmente de um cluster do MemoryDB (API do MemoryDB)

1. Determine quais tipos de nós você pode aumentar a escala verticalmente usando a ação `ListAllowedNodeTypeUpdates` da API do MemoryDB com o seguinte parâmetro.

- `ClusterName` – o nome do cluster. Use esse parâmetro para descrever um cluster específico em vez de todos os clusters.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=ListAllowedNodeTypeUpdates
    &ClusterName=MyCluster
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210802T192317Z
    &X-Amz-Credential=<credential>
```

Para obter mais informações, consulte ListAllowedNodeTypeUpdates na referência da API do MemoryDB para Redis.

2. Escale seu cluster atual para o novo tipo de nó usando a ação `UpdateCluster` da API do MemoryDB e com os seguintes parâmetros.

- `ClusterName` – o nome do cluster.
- `NodeType` – o novo tipo de nó maior dos clusters nesse cluster. Esse valor deve ser um dos tipos de instância retornados pela ação `ListAllowedNodeTypeUpdates` na etapa 1.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateCluster
    &NodeType=db.r6g.2xlarge
    &ClusterName=myCluster
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Para obter mais informações, consulte UpdateCluster.

Redução de escala vertical online

Tópicos

Redução de escala vertical de clusters do MemoryDB (console)

O procedimento a seguir descreve como reduzir a escala verticalmente de um cluster do MemoryDB usando o AWS Management Console. Durante esse processo, o cluster do MemoryDB continuará a atender solicitações com tempo de inatividade mínimo.

Para reduzir a escala verticalmente de um cluster do MemoryDB (console)

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. Na lista de clusters, escolha seu cluster preferido.
3. Escolha Ações e Modificar.
4. Na caixa de diálogo Modificar cluster:
    - Na lista Node type, escolha o tipo de nó a partir do qual você deseja escalar. Para reduzir, selecione um tipo de nó menor do que o nó existente. Observe que nem todos os tipos de nó estão disponíveis para redução.
5. Escolha Salvar alterações.

    O status do cluster muda para Modificação. Quando o status mudar para available, a modificação estará completa, e você poderá começar a usar o novo cluster.

Redução de escala vertical dos clusters do MemoryDB (AWS CLI)

O procedimento a seguir descreve como reduzir a escala verticalmente de um cluster do MemoryDB usando o AWS CLI. Durante esse processo, o cluster do MemoryDB continuará a atender solicitações com tempo de inatividade mínimo.

Para reduzir a escala verticalmente de um cluster do MemoryDB (AWS CLI)

1. Determine os tipos de nó que você pode reduzir executando o comando `list-allowed-node-type-updates` da AWS CLI com o seguinte parâmetro.

   Para Linux, macOS ou Unix:

   ```
   aws memorydb list-allowed-node-type-updates \
        --cluster-name my-cluster-name
   ```

   Para Windows:

   ```
   aws memorydb list-allowed-node-type-updates ^
        --cluster-name my-cluster-name
   ```

   A saída do comando acima é semelhante a esta (formato JSON).

   ```
   {
        "ScaleUpNodeTypes": [
            "db.r6g.2xlarge",
            "db.r6g.large"
        ],
        "ScaleDownNodeTypes": [
            "db.r6g.large"
        ],
   }
   ```

   Para obter mais informações, consulte list-allowed-node-type-updates.

2. Modifique seu cluster para reduzir a escala verticalmente para o novo tipo de nó menor usando o comando `update-cluster` com os seguintes parâmetros.

   - `--cluster-name` – o nome do cluster para o qual você está reduzindo a escala verticalmente.
   - `--node-type` – o novo tipo de nó para o qual você deseja escalar o cluster. Esse valor deve ser um dos tipos de nós retornados pelo comando `list-allowed-node-type-updates` na etapa 1.

   Para Linux, macOS ou Unix:

```
aws memorydb update-cluster  \
     --cluster-name my-cluster \
     --node-type db.r6g.large
```

Para Windows:

```
aws memorydb update-cluster ^
     --cluster-name my-cluster ^
     --node-type db.r6g.large
```

Para obter mais informações, consulte update-cluster.

Redução de escala vertical de clusters do MemoryDB (API do MemoryDB)

O processo a seguir escala seu cluster do tipo de nó atual para um novo tipo de nó menor usando a API do MemoryDB. Durante esse processo, o cluster do MemoryDB continuará a atender solicitações com tempo de inatividade mínimo.

O tempo necessário para reduzir a escala verticalmente até um tipo de nó menor varia, dependendo do tipo de nó e da quantidade de dados no seu cluster atual.

Redução de escala vertical (API do MemoryDB)

1. Determine quais tipos de nó para os quais você pode reduzir a escala verticalmente usando a ação ListAllowedNodeTypeUpdates da API com o seguinte parâmetro:

   - `ClusterName` – o nome do cluster. Use esse parâmetro para descrever um cluster específico em vez de todos os clusters.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=ListAllowedNodeTypeUpdates
    &ClusterName=MyCluster
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210802T192317Z
    &X-Amz-Credential=<credential>
```

2. Escale seu cluster atual para o novo tipo de nó usando a ação UpdateCluster da API com os seguintes parâmetros.

   - `ClusterName` – o nome do cluster.
   - `NodeType` – o novo tipo de nó menor dos clusters nesse cluster. Esse valor deve ser um dos tipos de instância retornados pela ação `ListAllowedNodeTypeUpdates` na etapa 1.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateCluster
    &NodeType=db.r6g.2xlarge
    &ClusterName=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Configuração de parâmetros do mecanismo usando grupos de parâmetros

O MemoryDB para Redis usa parâmetros para controlar as propriedades de runtime de seus nós e clusters. Geralmente, as versões mais recentes do mecanismo incluem parâmetros adicionais para dar suporte à funcionalidade mais recente. Para tabelas de parâmetros, consulte Parâmetros específicos do Redis.

Como seria de se esperar, alguns valores de parâmetros, como `maxmemory`, são determinados pelo mecanismo e tipo de nó. Para uma tabela desses valores de parâmetro por tipo de nó, consulte Parâmetros específicos do tipo de nó do MemoryDB.

Tópicos

- Gerenciamento de parâmetros
- Camadas de grupos de parâmetros

- Criar um parameter group
- Listagem de grupos de parâmetros por nome
- Listagem dos valores de um grupo de parâmetros
- Modificar um parameter group
- Exclusão de um grupo de parâmetros
- Parâmetros específicos do Redis

Gerenciamento de parâmetros

Os parâmetros são agrupados em parameter groups nomeados para facilitar o gerenciamento de parâmetros. Um parameter group representa uma combinação de valores específicos para os parâmetros que são transmitidos ao software do mecanismo durante a inicialização. Esses valores determinam como o processo do mecanismo em cada nó se comportará em runtime. Os valores dos parâmetros em um parameter group específico aplicam-se a todos os nós associados ao grupo, independentemente do cluster ao qual eles pertencem.

Para ajustar o desempenho do cluster, você pode modificar alguns valores de parâmetros ou alterar o parameter group do cluster.

- Não é possível modificar ou excluir os parameter groups padrão. Se você precisar de valores de parâmetros personalizados, deverá criar um parameter group personalizado.
- A família do parameter groups e o cluster que você está atribuindo a ela devem ser compatíveis. Por exemplo, se o seu cluster estiver executando o Redis versão 6, você só poderá usar grupos de parâmetros, padrão ou personalizados, da família memorydb_redis6.
- Quando você altera os parâmetros de um cluster, a alteração é aplicada ao cluster imediatamente. Isso é verdadeiro se você alterar o próprio grupo de parâmetro do cluster ou um valor do parâmetro dentro do grupo do parâmetro do cluster.

Camadas de grupos de parâmetros

Níveis de grupos de parâmetros do MemoryDB para Redis

Padrão global

O grupo de parâmetros raiz de nível superior para todos os clientes do MemoryDB para Redis na região.

O grupo de parâmetros padrão global:

- É reservado para o MemoryDB e não está disponível para o cliente.

Padrão do cliente

Uma cópia do grupo de parâmetros padrão global que é criada para uso do cliente.

O grupo de parâmetros padrão do cliente:

- É criado e de propriedade do MemoryDB.
- Está disponível para o cliente para uso como um grupo de parâmetros para qualquer cluster que esteja executando uma versão de mecanismo compatível com esse grupo de parâmetros.
- Não pode ser editado pelo cliente.

Propriedade do cliente

Uma cópia do grupo de parâmetros padrão do cliente. Um grupo de parâmetros de propriedade do cliente é criado sempre que o cliente cria um grupo de parâmetros.

O grupo de parâmetros de propriedade do cliente:

- É criado e de propriedade do cliente.
- Pode ser atribuído a qualquer um dos clusters compatíveis com o cliente.
- Pode ser modificado pelo cliente para criar um grupo de parâmetros personalizado.

  Nem todos os valores dos parâmetros podem ser modificados. Para ter mais informações, consulte Parâmetros específicos do Redis.

Criar um parameter group

Você precisará criar um novo parameter group se houver um ou mais valores de parâmetros que você deseja alterar a partir dos valores padrão. Você pode criar um grupo de parâmetros usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

Criação de um grupo de parâmetros (console)

O procedimento a seguir mostra como criar um grupo de parâmetros usando o console do MemoryDB.

Para criar um grupo de parâmetros usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista de todos os parameter groups disponíveis, no painel de navegação à esquerda, escolha Parameter Groups.
3. Para criar um grupo de parâmetros, selecione Criar grupo de parâmetros.

   A página Criar grupo de parâmetros é exibida.
4. Na caixa Name, digite um nome exclusivo para esse parameter group.

   Ao criar um cluster ou modificar o parameter group de um cluster, você escolherá o parameter group pelo seu nome. Portanto, recomendamos que o nome seja informativo e de alguma forma identifique a família do parameter group.

   As limitações de nomenclatura de grupo de parâmetros são as seguintes:

   - Deve começar com uma letra ASCII.
   - Pode conter apenas letras ASCII, dígitos e hífens.
   - Deve ter entre 1 e 255 caracteres.
   - Não podem conter dois hifens consecutivos.
   - Não podem terminar com um hífen.
5. Na caixa Description, digite uma descrição para o parameter group.
6. Na caixa de compatibilidade da versão do Redis, escolha uma versão do mecanismo à qual esse grupo de parâmetros corresponde.

7. Nas Tags, adicione opcionalmente tags para pesquisar e filtrar seus grupos de parâmetros ou monitorar seus AWS custos.
8. Para criar o parameter group, escolha Create.

   Para encerrar o processo sem criar o parameter group, escolha Cancel.
9. Quando o parameter group for criado, ele terá os valores padrão da família. Para alterar os valores padrão, você deve modificar o parameter group. Para ter mais informações, consulte Modificar um parameter group.

Criação de um grupo de parâmetros (AWS CLI)

Para criar um grupo de parâmetros usando o AWS CLI, use o comando `create-parameter-group` com esses parâmetros.

- `--parameter-group-name`: O nome do grupo de parâmetros.

  As limitações de nomenclatura de grupo de parâmetros são as seguintes:
  - Deve começar com uma letra ASCII.
  - Pode conter apenas letras ASCII, dígitos e hífens.
  - Deve ter entre 1 e 255 caracteres.
  - Não podem conter dois hifens consecutivos.
  - Não podem terminar com um hífen.
- `--family`: o mecanismo e a família de versões para o grupo de parâmetros.
- `--description`: uma descrição fornecida pelo usuário para o grupo de parâmetros.

Example

O exemplo a seguir cria um grupo de parâmetros chamado myRedis6x usando a família memorydb_redis6 como modelo.

Para Linux, macOS ou Unix:

```
aws memorydb create-parameter-group \
    --parameter-group-name myRedis6x  \
    --family memorydb_redis6 \
    --description "My first parameter group"
```

Para Windows:

```
aws memorydb create-parameter-group ^
    --parameter-group-name myRedis6x  ^
    --family memorydb_redis6 ^
    --description "My first parameter group"
```

A saída desse comando deve ser semelhante a esta.

```
{
    "ParameterGroup": {
        "Name": "myRedis6x",
        "Family": "memorydb_redis6",
        "Description": "My first parameter group",
        "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x"
    }
}
```

Quando o parameter group for criado, ele terá os valores padrão da família. Para alterar os valores padrão, você deve modificar o parameter group. Para ter mais informações, consulte Modificar um parameter group.

Para ter mais informações, consulte `create-parameter-group`.

Criação de um grupo de parâmetros (API do MemoryDB)

Para criar um grupo de parâmetros usando a API do MemoryDB, use a ação `CreateParameterGroup` com esses parâmetros.

- `ParameterGroupName`: O nome do grupo de parâmetros.

  As limitações de nomenclatura de grupo de parâmetros são as seguintes:
  - Deve começar com uma letra ASCII.
  - Pode conter apenas letras ASCII, dígitos e hífens.
  - Deve ter entre 1 e 255 caracteres.
  - Não podem conter dois hifens consecutivos.
  - Não podem terminar com um hífen.
- `Family`: o mecanismo e a família de versões para o grupo de parâmetros. Por exemplo, `memorydb_redis6`.

- `Description`: uma descrição fornecida pelo usuário para o grupo de parâmetros.

Example

O exemplo a seguir cria um grupo de parâmetros chamado myRedis6x usando a família memorydb_redis6 como modelo.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=CreateParameterGroup
   &Family=memorydb_redis6
   &ParameterGroupName=myRedis6x
   &Description=My%20first%20parameter%20group
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

A resposta dessa ação deve ser algo semelhante ao seguinte.

```
<CreateParameterGroupResponse xmlns="http://memory-db.us-east-1.amazonaws.com/
doc/2021-01-01/">
  <CreateParameterGroupResult>
    <ParameterGroup>
      <Name>myRedis6x</Name>
      <Family>memorydb_redis6</Family>
      <Description>My first parameter group</Description>
      <ARN>arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x</ARN>
    </ParameterGroup>
  </CreateParameterGroupResult>
  <ResponseMetadata>
    <RequestId>d8465952-af48-11e0-8d36-859edca6f4b8</RequestId>
  </ResponseMetadata>
</CreateParameterGroupResponse>
```

Quando o parameter group for criado, ele terá os valores padrão da família. Para alterar os valores padrão, você deve modificar o parameter group. Para ter mais informações, consulte Modificar um parameter group.

Para ter mais informações, consulte `CreateParameterGroup`.

Listagem de grupos de parâmetros por nome

Você pode listar os grupos de parâmetros usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

Listagem de grupos de parâmetros por nome (console)

O procedimento a seguir mostra como visualizar uma lista dos grupos de parâmetros usando o console do MemoryDB.

Para listar grupos de parâmetros usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista de todos os parameter groups disponíveis, no painel de navegação à esquerda, escolha Parameter Groups.

Listando grupos de parâmetros por nome (AWS CLI)

Para gerar uma lista de grupos de parâmetros usando o AWS CLI, use o comando`describe-parameter-groups`. Se você fornecer um nome de parameter group, somente esse parameter group será listado. Se você não fornecer o nome de um parameter group, até `--max-results` parameter groups serão listados. Em ambos os casos, o nome, a família e a descrição do parameter group estão listados.

Example

O código de exemplo a seguir lista o grupo de parâmetros myRedis6x.

Para Linux, macOS ou Unix:

```
aws memorydb describe-parameter-groups \
    --parameter-group-name myRedis6x
```

Para Windows:

```
aws memorydb describe-parameter-groups ^
    --parameter-group-name myRedis6x
```

A saída desse comando será algo assim, listando o nome, a família e a descrição do parameter group.

```
{
    "ParameterGroups": [
     {
         "Name": "myRedis6x",
         "Family": "memorydb_redis6",
         "Description": "My first parameter group",
                "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
     }
    ]
}
```

Example

O código de exemplo a seguir lista o grupo de parâmetros myRedis6x para grupos de parâmetros em execução no mecanismo Redis versão 5.0.6 e posteriores.

Para Linux, macOS ou Unix:

```
aws memorydb describe-parameter-groups \
    --parameter-group-name myRedis6x
```

Para Windows:

```
aws memorydb describe-parameter-groups ^
    --parameter-group-name myRedis6x
```

A saída desse comando será semelhante a esta, listando o nome, a família e a descrição do grupo de parâmetros.

```
{
    "ParameterGroups": [
     {
         "Name": "myRedis6x",
         "Family": "memorydb_redis6",
         "Description": "My first parameter group",
                "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
```

```
     }
    ]
}
```

Example

O código de exemplo a seguir lista até 20 grupos de parâmetros.

```
aws memorydb describe-parameter-groups --max-results 20
```

A saída JSON desse comando será semelhante a esta, listando o nome, a família e a descrição de cada grupo de parâmetros.

```
{
    "ParameterGroups": [
        {
            "ParameterGroupName": "default.memorydb-redis6",
            "Family": "memorydb_redis6",
            "Description": "Default parameter group for memorydb_redis6",
            "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
default.memorydb-redis6"
        },
        ...
    ]
}
```

Para ter mais informações, consulte `describe-parameter-groups`.

Listagem de grupos de parâmetros por nome (MemoryDB API)

Para gerar uma lista de grupos de parâmetros usando API do MemoryDB, use a ação `DescribeParameterGroups`. Se você fornecer um nome de parameter group, somente esse parameter group será listado. Se você não fornecer o nome de um parameter group, até `MaxResults` parameter groups serão listados. Em ambos os casos, o nome, a família e a descrição do parameter group estão listados.

Example

O código de exemplo a seguir lista até 20 grupos de parâmetros.

```
https://memory-db.us-east-1.amazonaws.com/
```

```
   ?Action=DescribeParameterGroups
   &MaxResults=20
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

A resposta dessa ação será semelhante a esta, listando o nome, a família e a descrição, no caso de memorydb_redis6, para cada grupo de parâmetros.

```
<DescribeParameterGroupsResponse xmlns="http://memory-db.us-east-1.amazonaws.com/
doc/2021-01-01/">
  <DescribeParameterGroupsResult>
    <ParameterGroups>
      <ParameterGroup>
        <Name>myRedis6x</Name>
        <Family>memorydb_redis6</Family>
        <Description>My custom Redis 6 parameter group</Description>
        <ARN>arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x</ARN>
      </ParameterGroup>
       <ParameterGroup>
        <Name>default.memorydb-redis6</Name>
        <Family>memorydb_redis6</Family>
        <Description>Default parameter group for memorydb_redis6</Description>
        <ARN>arn:aws:memorydb:us-east-1:012345678912:parametergroup/default.memorydb-
redis6</ARN>
      </ParameterGroup>
    </ParameterGroups>
  </DescribeParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeParameterGroupsResponse>
```

Example

O código de exemplo a seguir lista o grupo de parâmetros myRedis6x.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeParameterGroups
   &ParameterGroupName=myRedis6x
   &SignatureVersion=4
```

```
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

A resposta dessa ação será semelhante ao seguinte: listagem do nome, família e descrição.

```
<DescribeParameterGroupsResponse xmlns="http://memory-db.us-east-1.amazonaws.com/
doc/2021-01-01/">
  <DescribeParameterGroupsResult>
    <ParameterGroups>
      <ParameterGroup>
        <Name>myRedis6x</Name>
        <Family>memorydb_redis6</Family>
        <Description>My custom Redis 6 parameter group</Description>
        <ARN>arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x</ARN>
      </ParameterGroup>
    </ParameterGroups>
  </DescribeParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeParameterGroupsResponse>
```

Para ter mais informações, consulte DescribeParameterGroups.

Listagem dos valores de um grupo de parâmetros

Você pode listar os parâmetros e seus valores para um grupo de parâmetros usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

Listagem dos valores de um grupo de parâmetros (console)

O procedimento a seguir mostra como listar os parâmetros e seus valores para um grupo de parâmetros usando o console do MemoryDB.

Listar os parâmetros de um grupo de parâmetros e seus valores usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista de todos os parameter groups disponíveis, no painel de navegação à esquerda, escolha Parameter Groups.
3. Escolha o grupo de parâmetros para o qual você deseja listar os parâmetros e valores selecionando o nome (não a caixa ao lado) do nome do grupo de parâmetros.

   Os parâmetros e seus valores serão listados na parte inferior da tela. Devido ao número de parâmetros, talvez seja necessário rolar para cima e para baixo para encontrar o parâmetro de interesse.

Listando os valores de um grupo de parâmetros (AWS CLI)

Para listar os parâmetros de um grupo de parâmetros e seus valores usando o AWS CLI, use o comando`describe-parameters`.

Example

O código de exemplo a seguir lista todos os parâmetros e seus valores para o grupo de parâmetros myRedis6x.

Para Linux, macOS ou Unix:

```
aws memorydb describe-parameters \
    --parameter-group-name myRedis6x
```

Para Windows:

```
aws memorydb describe-parameters ^
    --parameter-group-name myRedis6x
```

Para ter mais informações, consulte `describe-parameters`.

Listagem dos valores de um grupo de parâmetros (API do MemoryDB)

Para listar os parâmetros de um grupo de parâmetros e seus valores usando a API do MemoryDB, use a ação `DescribeParameters`.

Para ter mais informações, consulte `DescribeParameters`.

Modificar um parameter group

> **Important**
>
> Não é possível modificar um parameter group padrão.

Você pode modificar alguns valores de parâmetros em um parameter group. Esses valores de parâmetros são aplicados a clusters associados ao parameter group. Para obter mais informações sobre quando uma alteração no valor de um parâmetro é aplicada a um parameter group, consulte Parâmetros específicos do Redis.

Modificação de um grupo de parâmetros (console)

O procedimento a seguir mostra como alterar o valor do parâmetro usando o console do MemoryDB. Você usaria o mesmo procedimento para alterar o valor de qualquer parâmetro.

Para alterar o valor de um parâmetro usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista de todos os parameter groups disponíveis, no painel de navegação à esquerda, escolha Parameter Groups.
3. Escolha o grupo de parâmetros que deseja modificar selecionando o botão de rádio à esquerda do nome do grupo de parâmetros.

   Escolha Ações e, em seguida, Visualizar detalhes. Como alternativa, você também pode selecionar o nome do grupo de parâmetros para ir para a página de detalhes.

4. Para modificar o parâmetro, selecione Editar. Todos os parâmetros editáveis estarão habilitados para edição. Talvez seja necessário percorrer as páginas para encontrar o parâmetro que você deseja alterar. Como alternativa, você pode pesquisar o parâmetro por nome, valor ou tipo na caixa de pesquisa.
5. Faça as modificações necessárias nos parâmetros.
6. Para salvar suas alterações, escolha Salvar alterações.
7. Se você modificou os valores dos parâmetros em várias páginas, poderá revisar todas as alterações selecionando Visualizar alterações. Para confirmar as alterações, selecione Salvar alterações. Para fazer mais modificações, selecione Voltar.
8. A página de detalhes do parâmetro também oferece a opção de redefinir para os valores padrão. Para redefinir os valores padrão, selecione Redefinir para o padrão. As caixas de seleção aparecerão no lado esquerdo de todos os parâmetros. Você pode selecionar os que deseja redefinir e selecionar Prossiga para redefinir para confirmar.

   Selecione Confirmar para confirmar a ação de redefinição na caixa de diálogo.
9. A página de detalhes do parâmetro permite que você defina o número de parâmetros que deseja ver em cada página. Use o ícone de engrenagem no lado direito para fazer essas alterações. Você também pode ativar/desativar as colunas desejadas na página de detalhes. Essas alterações permanecem durante a sessão do console.

   Para localizar o parâmetro que você alterou, consulte Parâmetros específicos do Redis.

Modificando um grupo de parâmetros (AWS CLI)

Para alterar o valor de um parâmetro usando o AWS CLI, use o comando`update-parameter-group`.

Para encontrar o nome e os valores permitidos do parâmetro que você deseja alterar, consulte Parâmetros específicos do Redis

Para obter mais informações, consulte update-parameter-group.

Modificação de um grupo de parâmetros (MemoryDB API)

Para alterar os valores dos parâmetros de um grupo de parâmetros usando a API MemoryDB, use a ação `UpdateParameterGroup`.

Para encontrar o nome e os valores permitidos do parâmetro que você deseja alterar, consulte Parâmetros específicos do Redis

Para ter mais informações, consulte `UpdateParameterGroup`.

Exclusão de um grupo de parâmetros

Você pode excluir um grupo de parâmetros personalizado usando o console MemoryDB AWS CLI, o ou a API MemoryDB.

Não será possível excluir um parameter group se ele estiver associado a qualquer cluster. Você também não pode excluir nenhum dos parameter groups padrão.

Exclusão de um grupo de parâmetros (console)

O procedimento a seguir mostra como excluir um grupo de parâmetros usando o console do MemoryDB.

Para excluir um grupo de parâmetros usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Para ver uma lista de todos os parameter groups disponíveis, no painel de navegação à esquerda, escolha Parameter Groups.
3. Escolha os grupos de parâmetros que deseja excluir selecionando o botão de opções à esquerda do nome do grupo de parâmetros.

   Escolha Ações e, em seguida, escolha Excluir.
4. A tela de confirmação Delete Parameter Groups será exibida.
5. Para excluir os grupos de parâmetros, digite Delete na caixa de texto de confirmação.

   Para manter os parameter groups, escolha Cancel.

Excluindo um grupo de parâmetros (AWS CLI)

Para excluir um grupo de parâmetros usando o AWS CLI, use o comando`delete-parameter-group`. Para o parameter group a ser excluído, o parameter group especificado por `--parameter-group-name` não pode ter nenhum cluster associado a ele, nem pode ser um parameter group padrão.

O código de exemplo a seguir exclui o grupo de parâmetros myRedis6x.

Example

Para Linux, macOS ou Unix:

```
aws memorydb delete-parameter-group \
    --parameter-group-name myRedis6x
```

Para Windows:

```
aws memorydb delete-parameter-group ^
    --parameter-group-name myRedis6x
```

Para obter mais informações, consulte delete-parameter-group.

Exclusão de um grupo de parâmetros (MemoryDB API)

Para excluir um grupo de parâmetros usando a API do MemoryDB, use a ação `DeleteParameterGroup`. Para o parameter group a ser excluído, o parameter group especificado por `ParameterGroupName` não pode ter nenhum cluster associado a ele, nem pode ser um parameter group padrão.

Example

O código de exemplo a seguir exclui o grupo de parâmetros myRedis6x.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DeleteParameterGroup
   &ParameterGroupName=myRedis6x
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

Para ter mais informações, consulte `DeleteParameterGroup`.

Parâmetros específicos do Redis

Se você não especificar um grupo de parâmetros para seu cluster do Redis, será usado um grupo de parâmetros padrão apropriado à versão de seu mecanismo. Não é possível alterar os valores de nenhum parâmetro em um grupo de parâmetros padrão. No entanto, é possível criar um grupo de parâmetros personalizado e atribui-lo ao seu cluster a qualquer momento, desde que os valores de parâmetros condicionalmente modificáveis sejam os mesmos nos dois grupos de parâmetros. Para ter mais informações, consulte Criar um parameter group.

Tópicos

Alterações de parâmetros do Redis 7

> **Note**
>
> O MemoryDB introduziu uma versão prévia da Pesquisa vetorial que inclui um novo grupo de parâmetros imutáveis `default.memorydb-redis7.search.preview`. Esse grupo de parâmetros está disponível no console do MemoryDB e ao criar um novo vector-search-enabled cluster usando o comando da CLI create-cluster. A versão prévia está disponível nas seguintes AWS regiões: Leste dos EUA (Norte da Virgínia), Leste dos EUA (Ohio), Oeste dos EUA (Oregon), Ásia-Pacífico (Tóquio) e Europa (Irlanda).

Família de grupos de parâmetros: memorydb_redis7

Os parâmetros adicionados no Redis 7 são os seguintes.

Nome	Detalhes	Descrição
`latency-tracking`	Valores permitidos: `yes`, `no` Padrão: `no` Tipo: string	Quando definido como yes (sim), rastreia as latências por comando e permite exportar a distribuição de percentil por meio do comando de estatísticas de latência do `INFO` e as

Nome	Detalhes	Descrição
	Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	distribuições de latência cumulativa (histogra mas) por meio do comando `LATENCY`.
`hash-max-listpack-entries`	Valores permitidos: `0+` Padrão: `512` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	O número máximo de entradas de hash para que o conjunto de dados seja compactado.
`hash-max-listpack-value`	Valores permitidos: `0+` Padrão: `64` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	O limite das maiores entradas de hash para que o conjunto de dados seja compactado.

Nome	Detalhes	Descrição
`zset-max-listpack-entries`	Valores permitidos: `0+` Padrão: `128` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	O número máximo de entradas do conjunto classificado para que o conjunto de dados seja compactado.
`zset-max-listpack-value`	Valores permitidos: `0+` Padrão: `64` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	O limite das maiores entradas do conjunto classificado para que o conjunto de dados seja compactado.

Os parâmetros alterados no Redis 7 são os seguintes.

Nome	Detalhes	Descrição
`activerehashing`	Permite modificação: no. No Redis 7, esse parâmetro está oculto e ativado por padrão. Para desativá-lo, você precisa criar um caso de suporte.	Permite modificação era Sim.

Os parâmetros removidos no Redis 7 são os seguintes.

Nome	Detalhes	Descrição
`hash-max-ziplist-entries`	Valores permitidos: `0+` Padrão: `512` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	Use `listpack` em vez de `ziplist` para representar uma pequena codificação de hash
`hash-max-ziplist-value`	Valores permitidos: `0+` Padrão: `64` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	Use `listpack` em vez de `ziplist` para representar uma pequena codificação de hash
`zset-max-ziplist-entries`	Valores permitidos: `0+` Padrão: `128` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	Use `listpack` em vez de `ziplist` para representar uma pequena codificação de hash.

Nome	Detalhes	Descrição
`zset-max-ziplist-value`	Valores permitidos: `0+` Padrão: `64` Tipo: inteiro Modificável: sim As alterações entrarão em vigor: imediatamente em todos os nós no cluster.	Use `listpack` em vez de `ziplist` para representar uma pequena codificação de hash.

Parâmetros do Redis 6

> **Note**
>
> Na versão 6.2 do mecanismo Redis, quando a família de nós r6gd foi introduzida para uso com Classificação de dados em níveis, somente as políticas de memória máxima `noeviction`, `volatile-lru` e `allkeys-lru` são compatíveis com os tipos de nós r6gd.

Família do grupo de parâmetros: memorydb_redis6

Os parâmetros adicionados no Redis 6 são os seguintes.

Nome	Detalhes	Descrição
`maxmemory-policy`	Tipo: STRING Valores permitidos: volatile-lru, allkeys-lru, volatile-lfu, allkeys-lfu, volatile-random, allkeys-random, volatile-ttl, noeviction Padrão: noeviction	A política de remoção de chaves quando o uso máximo da memória é atingido. Para obter mais informações, consulte Usando o Redis como um cache LRU.

Nome	Detalhes	Descrição
`list-compress-depth`	Tipo: INTEGER Valores permitidos: 0- Padrão: 0	A profundidade de compactação é o número de nós ziplist de lista rápida de cada lado da lista a serem excluídos da compactação. O início e o final cauda da lista são sempre descompactados para operações Push e Pop rápidas. As configurações são: • 0: desabilitar toda a compactação. • 1: começar a compactar com o 1º nó de início e final. [início] -> nó-> nó -> ...-> nó -> [final] Todos os nós, exceto [início] e [final] são compactados. • 2: começar a compactar com o 2º nó de início e final. [início] -> [próximo] -> nó-> nó -> ...-> nó -> [anterior] -> [final] [início], [próximo], [anterior], [final] não são compactados. Todos os outros nós são compactados. • Etc.

Nome	Detalhes	Descrição
`hll-sparse-max-bytes`	Tipo: INTEGER Valores permitidos: 1-16000 Padrão: 3000	HyperLogLog limite de bytes de represent ação esparsa. O limite inclui o cabeçalho de 16 bytes. Quando o HyperLogLog uso da representação esparsa ultrapassa esse limite, ele é convertido na representação densa. Não é recomendado um valor superior a 16000, porque, nesse ponto, a representação densa é mais eficiente em termos de memória. Recomendamos um valor de 3000 para ter os benefícios da codificação eficiente do espaço sem diminuir demais o `PFADD`, que é O(N) com a codificação esparsa. O valor pode ser aumentado para ~ 10000 quando a CPU não é uma preocupação, mas o espaço é, e o conjunto de dados é composto por muitos HyperLogLogs com cardinalidade na faixa de 0 a 15000.
`lfu-log-factor`	Tipo: INTEGER Valores permitidos: 1- Padrão: 10	O fator do log para incrementar o contador de chaves da política de remoção da LFU.
`lfu-decay-time`	Tipo: INTEGER Valores permitidos: 0- Padrão: 1	A quantidade de tempo, em minutos, para diminuir o contador de chaves da política de remoção da LFU.

Nome	Detalhes	Descrição
`active-defrag-max-scan-fields`	Tipo: INTEGER Valores permitidos: 1-1000000 Padrão: 1000	Número máximo de campos set/hash/zset/list que serão processados a partir da varredura do dicionário principal durante a desfragmentação ativa.
`active-defrag-threshold-upper`	Tipo: INTEGER Valores permitidos: 1-100 Padrão: 100	Porcentagem máxima de fragmentação em que usamos o esforço máximo.
`client-output-buffer-limit-pubsub-hard-limit`	Tipo: INTEGER Valores permitidos: 0- Padrão: 33554432	Para clientes de publicação/assinatura do Redis: se o buffer de saída de um cliente atingir o número especificado de bytes, o cliente será desconectado.
`client-output-buffer-limit-pubsub-soft-limit`	Tipo: INTEGER Valores permitidos: 0- Padrão: 8388608	Para clientes de publicação/assinatura do Redis: se o buffer de saída de um cliente atingir o número especificado de bytes, o cliente será desconectado, mas somente se essa condição persistir por `client-output-buffer-limit-pubsub-soft-seconds.`
`client-output-buffer-limit-pubsub-soft-seconds`	Tipo: INTEGER Valores permitidos: 0- Padrão: 60	Para clientes de publicação/assinatura do Redis: se o buffer de saída de um cliente permanecer em `client-output-buffer-limit-pubsub-soft-limit` bytes por mais tempo que esse número de segundos, o cliente será desconectado.

Nome	Detalhes	Descrição
`timeout`	Tipo: INTEGER Valores permitidos: 0,20- Padrão: 0	O número de segundos que um nó espera antes do tempo limite. Os valores são: • 0 – nunca desconectar um cliente ocioso. • 1-19 – valores inválidos. • >=20 – o número de segundos que um nó espera antes de desconectar um cliente ocioso.
`notify-keyspace-events`	Tipo: STRING Valores permitidos: NULL Padrão: NULL	Os eventos do espaço de chave para o Redis notificar os clientes do Pub/Sub (publicação/assinatura). Por padrão, todas as notificações estão desabilitadas.
`maxmemory-samples`	Tipo: INTEGER Valores permitidos: 1- Padrão: 3	Para least-recently-used `(LRU)` `time-to-live (TTL)` cálculos, esse parâmetro representa o tamanho amostral das chaves a serem verificadas. Por padrão, o Redis escolhe 3 chaves e usa a que foi usada menos recentemente.
`slowlog-max-len`	Tipo: INTEGER Valores permitidos: 0- Padrão: 128	O comprimento máximo do Redis Slow Log. Não há limite para esse comprimento. Esteja ciente de que isso consumirá memória. Você pode recuperar a memória usada pelo log lento com `SLOWLOG RESET.`

Nome	Detalhes	Descrição
activerehashing	Tipo: STRING Valores permitidos: sim,não Padrão: sim	A tabela de hash principal é sofre rehashing dez vezes por segundo. Cada operação de rehash consome 1 milissegundo de tempo da CPU. Esse valor é definido quando você cria o grupo de parâmetros. Ao atribuir um novo grupo de parâmetros a um cluster, esse valor deve ser o mesmo nos grupo de parâmetros antigo e novo.
client-output-buffer-limit-normal-hard-limit	Tipo: INTEGER Valores permitidos: 0- Padrão: 0	Se o buffer de saída de um cliente atingir o número especificado de bytes, o cliente será desconectado. O padrão é zero (sem limite fixo).
client-output-buffer-limit-normal-soft-limit	Tipo: INTEGER Valores permitidos: 0- Padrão: 0	Se o buffer de saída de um cliente atingir o número especificado de bytes, o cliente será desconectado, mas somente se essa condição persistir por `client-output-buffer-limit-normal-soft-seconds` . O padrão é zero (sem limite flexível).
client-output-buffer-limit-normal-soft-seconds	Tipo: INTEGER Valores permitidos: 0- Padrão: 0	Se o buffer de saída de um cliente permanecer em `client-output-buffer-limit-normal-soft-limit` bytes por mais tempo que esse número de segundos, o cliente será desconectado. O padrão é zero (sem limite de tempo).

Nome	Detalhes	Descrição
`tcp-keepalive`	Tipo: INTEGER Valores permitidos: 0- Padrão: 300	Se estiver definido como um valor diferente de zero (N), os clientes do nó são sondados a cada N segundos para garantir que ainda estejam conectados. Com a configuração padrão de 0, essa sondagem não ocorre.
`active-defrag-cycle-min`	Tipo: INTEGER Valores permitidos: 1-75 Padrão: 5	Esforço mínimo para desfragmentação em porcentagem de CPU.
`stream-node-max-bytes`	Tipo: INTEGER Valores permitidos: 0- Padrão: 4096	A estrutura do fluxo de dados é uma árvore radix de nós que codifica vários itens dentro. Use esta configuração para especificar o tamanho máximo de um nó único em uma árvore radix em bytes. Se definido como 0, o tamanho do nó da árvore é ilimitado.
`stream-node-max-entries`	Tipo: INTEGER Valores permitidos: 0- Padrão: 100	A estrutura do fluxo de dados é uma árvore radix de nós que codifica vários itens dentro. Use essa configuração para especificar o número máximo de itens que um único nó pode conter antes de alternar para um novo nó ao anexar novas entradas de fluxo. Se definido como 0, o número de itens no nó da árvore é ilimitado.
`lazyfree-lazy-eviction`	Tipo: STRING Valores permitidos: sim,não Padrão: não	Realiza uma exclusão assíncrona em remoções.

Nome	Detalhes	Descrição
`active-defrag-ignore-bytes`	Tipo: INTEGER Valores permitidos: 1048576- Padrão: 104857600	Quantidade mínima de desperdício de fragmentação para iniciar a desfragmentação ativa.
`lazyfree-lazy-expire`	Tipo: STRING Valores permitidos: sim,não Padrão: não	Realiza uma exclusão assíncrona em chaves expiradas.
`active-defrag-threshold-lower`	Tipo: INTEGER Valores permitidos: 1-100 Padrão: 10	Porcentagem mínima de fragmentação para iniciar a desfragmentação ativa.
`active-defrag-cycle-max`	Tipo: INTEGER Valores permitidos: 1-75 Padrão: 75	Esforço máximo para desfragmentação em porcentagem de CPU.
`lazyfree-lazy-server-del`	Tipo: STRING Valores permitidos: sim,não Padrão: não	Realiza uma exclusão assíncrona para comandos que atualizam valores.

Nome	Detalhes	Descrição
`slowlog-log-slower-than`	Tipo: INTEGER Valores permitidos: 0- Padrão: 10000	O tempo máximo de execução, em microsseg undos, a ser excedido para que o seja feito o log do comando pelo atributo `Slow Log` do Redis. Observe que um número negativo desativa o log lento, enquanto um valor de zero força o log de todos os comandos.
`hash-max-ziplist-entries`	Tipo: INTEGER Valores permitidos: 0- Padrão: 512	Determina a quantidade de memória usada para hashes. Hashes com menos que o número especificado de entradas são armazenados usando uma codificação especial que economiza espaço.
`hash-max-ziplist-value`	Tipo: INTEGER Valores permitidos: 0- Padrão: 64	Determina a quantidade de memória usada para hashes. Hashes com entradas menores que o número especificado de bytes são armazenados usando uma codificação especial que economiza espaço.
`set-max-intset-entries`	Tipo: INTEGER Valores permitidos: 0- Padrão: 512	Determina a quantidade de memória utilizada para certos tipos de conjuntos (strings que são inteiros em radix 10 no intervalo de inteiros de 64 bits com sinal). Esses conjuntos com menos que o número especificado de entradas são armazenados usando uma codificação especial que economiza espaço.

Nome	Detalhes	Descrição
`zset-max-ziplist-entries`	Tipo: INTEGER Valores permitidos: 0- Padrão: 128	Determina a quantidade de memória utilizada para conjuntos classificados. Os conjuntos classificados com menos que o número especificado de elementos são armazenados usando uma codificação especial que economiza espaço.
`zset-max-ziplist-value`	Tipo: INTEGER Valores permitidos: 0- Padrão: 64	Determina a quantidade de memória utilizada para conjuntos classificados. Os conjuntos classificados com entradas menores que o número especificado de bytes são armazenados usando uma codificação especial que economiza espaço.
`tracking-table-max-keys`	Tipo: INTEGER Valores permitidos: 1-100000000 Padrão: 1000000	Para ajudar o cache do lado do cliente, o Redis oferece suporte ao monitoramento de quais clientes acessaram quais chaves. Quando a chave monitorada é modificada, mensagens de invalidação são enviadas a todos os clientes para notificá-los que seus valores armazenados em cache não são mais válidos. Esse valor permite que você especifique o limite superior desta tabela.
`acllog-max-len`	Tipo: INTEGER Valores permitidos: 1-10000 Padrão: 128	O número máximo de entradas no log da ACL.

Nome	Detalhes	Descrição
active-expire-effort	Tipo: INTEGER Valores permitidos: 1-10 Padrão: 1	O Redis exclui chaves que excederam seu tempo de vida por dois mecanismos. Em um, uma chave é acessada e se descobre que ela está expirada. No outro, um trabalho periódico amostra as chaves e faz com que aquelas que excederam seu tempo de vida expirem. Esse parâmetro define a quantidade de esforço que o Redis usa para expirar itens no trabalho periódico. O valor padrão de 1 tenta evitar ter mais de 10 por cento das chaves expiradas ainda na memória. Ele também tenta evitar consumir mais de 25% da memória total e adicionar latência ao sistema. Você pode aumentar esse valor até 10 para aumentar a quantidade de esforço gasto em chaves expirando. A compensação é mais CPU e latência potencialmente maior. Recomendamos um valor de 1, a menos que você esteja vendo alto uso de memória e possa tolerar um aumento na utilização da CPU.
lazyfree-lazy-user-del	Tipo: STRING Valores permitidos: sim,não Padrão: não	Especifica se o comportamento padrão do comando `DEL` age da mesma forma que `UNLINK`.
activedefrag	Tipo: STRING Valores permitidos: sim,não Padrão: não	Desfragmentação ativa da memória habilitada.

Nome	Detalhes	Descrição
`maxclients`	Tipo: INTEGER Valores permitidos: 65000 Padrão: 65000	O número máximo de clientes que podem ser conectados ao mesmo tempo. Não modificável.
`client-query-buffer-limit`	Tipo: INTEGER Valores permitidos: 1048576-1073741824 Padrão: 1073741824	Tamanho máximo de um único buffer de consulta do cliente. As alterações ocorrem imediatamente
`proto-max-bulk-len`	Tipo: INTEGER Valores permitidos: 1048576-536870912 Padrão: 536870912	Tamanho máximo de uma única solicitação de elemento. As alterações ocorrem imediatamente

Parâmetros específicos do tipo de nó do MemoryDB

Embora a maioria dos parâmetros tenha um valor único, alguns parâmetros têm valores diferentes dependendo do tipo de nó usado. A tabela a seguir mostra o valor padrão para o `maxmemory` para cada tipo de nó. O valor de `maxmemory` é o número máximo de bytes disponíveis para uso, dados e outros usos no nó.

Tipo de nó	Maxmemory
db.r7g.large	14037181030
db.r7g.xlarge	28261849702
db.r7g.2xlarge	56711183565

Tipo de nó	Maxmemory
db.r7g.4xlarge	113609865216
db.r7g.8xlarge	225000375228
db.r7g.12xlarge	341206346547
db.r7g.16xlarge	450000750456
db.r6gd.xlarge	28261849702
db.r6gd.2xlarge	56711183565
db.r6gd.4xlarge	113609865216
db.r6gd.8xlarge	225000375228
db.r6g.large	14037181030
db.r6g.xlarge	28261849702
db.r6g.2xlarge	56711183565
db.r6g.4xlarge	113609865216
db.r6g.8xlarge	225000375228
db.r6g.12xlarge	341206346547
db.r6g.16xlarge	450000750456
db.t4g.small	1471026299
db.t4g.medium	3317862236

Note

Todos os tipos de instâncias do MemoryDB devem ser criados em uma nuvem privada virtual (VPC) da Amazon.

Tutorial: Configurando uma função Lambda para acessar o MemoryDB em uma Amazon VPC

Neste tutorial, você pode aprender como:

- Crie um cluster MemoryDB em sua Amazon Virtual Private Cloud padrão (Amazon VPC) na região us-east-1.
- Crie uma função Lambda para acessar o cluster. Ao criar a função do Lambda, você fornece os IDs de sub-rede da sua Amazon VPC e grupo de segurança de VPC para permitir que a função do Lambda acesse os recursos na sua VPC. Para ilustração neste tutorial, a função Lambda gera um UUID, o grava no cluster e o recupera do cluster.
- Invoque a função Lambda manualmente e verifique se ela acessou o cluster em sua VPC.
- Limpe a função, o cluster e a função do IAM do Lambda que foram configurados para este tutorial.

Tópicos

Etapa 1: criar um cluster

Para criar um cluster, siga estas etapas.

Tópicos

Etapa 1.1: criar um cluster

Nesta etapa, você cria um cluster na Amazon VPC padrão na região us-east-1 em sua conta usando a (CLI). AWS Command Line Interface Para obter informações sobre a criação de clusters usando o console ou a API do MemoryDB, consulte. Etapa 1: criar um cluster

```
aws memorydb create-cluster --cluster-name cluster-01 --engine-version 7.0 --acl-name
 open-access \
--description "MemoryDB IAM auth application" \
--node-type db.r6g.large
```

O valor do campo Status está definido como `CREATING`. Pode levar alguns minutos para que o MemoryDB conclua a criação do seu cluster.

Etapa 1.2: Copiar o endpoint do cluster

Verifique se o MemoryDB concluiu a criação do cluster com o `describe-clusters` comando.

```
aws memorydb describe-clusters \
--cluster-name cluster-01
```

Copie o endereço do endpoint do cluster mostrado na saída. Você precisará desse endereço ao criar o pacote de implantação da função do Lambda.

Etapa 1.3: Criar função do IAM

1. Crie um documento de política de confiança do IAM, conforme mostrado abaixo, para o perfil que permita que sua conta assuma o novo perfil. Salve a política em um arquivo chamado trust-policy.json. Certifique-se de substituir account_id 123456789012 nesta política pelo seu account_id.

   ```
   {
   "Version": "2012-10-17",
       "Statement": [{
           "Effect": "Allow",
           "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
           "Action": "sts:AssumeRole"
       },
       {
   ```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }]
}
```

2. Crie um documento de política do IAM, conforme mostrado abaixo. Salve a política em um arquivo chamado policy.json. Certifique-se de substituir account_id 123456789012 nesta política pelo seu account_id.

```
{
"Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
      "Action" : [
        "memorydb:Connect"
      ],
      "Resource" : [
        "arn:aws:memorydb:us-east-1:123456789012:cluster/cluster-01",
        "arn:aws:memorydb:us-east-1:123456789012:user/iam-user-01"
      ]
    }
  ]
}
```

3. Criar um perfil do IAM.

```
aws iam create-role \
--role-name "memorydb-iam-auth-app" \
--assume-role-policy-document file://trust-policy.json
```

4. Crie a política do IAM.

```
aws iam create-policy \
  --policy-name "memorydb-allow-all" \
  --policy-document file://policy.json
```

5. Anexe a política do IAM à função. Certifique-se de substituir account_id 123456789012 neste manual de política pelo seu account_id.

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

Etapa 1.4: Criar uma lista de controle de acesso (ACL)

1. Crie um novo usuário habilitado para o IAM.

```
aws memorydb create-user \
  --user-name iam-user-01 \
--authentication-mode Type=iam \
--access-string "on ~* +@all"
```

2. Crie uma ACL e anexe-a ao cluster.

```
aws memorydb create-acl \
  --acl-name iam-acl-01 \
  --user-names iam-user-01

aws memorydb update-cluster \
  --cluster-name cluster-01 \
  --acl-name iam-acl-01
```

Etapa 2: Criar uma função do Lambda

Para criar uma função Lambda, siga estas etapas.

Tópicos

Etapa 2.1: criar o pacote de implantação

Neste tutorial, fornecemos um exemplo de código em Python para sua função Lambda.

Python

O exemplo a seguir, o código Python lê e grava um item em seu cluster MemoryDB. Copie o código e o salve em um arquivo chamado `app.py`. Certifique-se de substituir o `cluster_endpoint` valor no código pelo endereço do endpoint que você copiou na etapa 1.2.

```
from typing import Tuple, Union
from urllib.parse import ParseResult, urlencode, urlunparse

import botocore.session
import redis
from botocore.model import ServiceId
from botocore.signers import RequestSigner
from cachetools import TTLCache, cached
import uuid

class MemoryDBIAMProvider(redis.CredentialProvider):
    def __init__(self, user, cluster_name, region="us-east-1"):
        self.user = user
        self.cluster_name = cluster_name
        self.region = region

        session = botocore.session.get_session()
        self.request_signer = RequestSigner(
            ServiceId("memorydb"),
            self.region,
            "memorydb",
            "v4",
            session.get_credentials(),
            session.get_component("event_emitter"),
        )

    # Generated IAM tokens are valid for 15 minutes
    @cached(cache=TTLCache(maxsize=128, ttl=900))
    def get_credentials(self) -> Union[Tuple[str], Tuple[str, str]]:
        query_params = {"Action": "connect", "User": self.user}

        url = urlunparse(
            ParseResult(
                scheme="https",
                netloc=self.cluster_name,
                path="/",
                query=urlencode(query_params),
                params="",
                fragment="",
```

```
            )
        )
        signed_url = self.request_signer.generate_presigned_url(
            {"method": "GET", "url": url, "body": {}, "headers": {}, "context": {}},
            operation_name="connect",
            expires_in=900,
            region_name=self.region,
        )
        # RequestSigner only seems to work if the URL has a protocol, but
        # MemoryDB only accepts the URL without a protocol
        # So strip it off the signed URL before returning
        return (self.user, signed_url.removeprefix("https://"))

def lambda_handler(event, context):
    username = "iam-user-01" # replace with your user id
    cluster_name = "cluster-01" # replace with your cache name
    cluster_endpoint = "clustercfg.cluster-01.xxxxxx.memorydb.us-east-1.amazonaws.com"
 # replace with your cluster endpoint
    creds_provider = MemoryDBIAMProvider(user=username, cluster_name=cluster_name)
    redis_client = redis.Redis(host=cluster_endpoint, port=6379,
 credential_provider=creds_provider, ssl=True, ssl_cert_reqs="none")

    key='uuid'
    # create a random UUID - this will be the sample element we add to the cluster
    uuid_in = uuid.uuid4().hex
    redis_client.set(key, uuid_in)
    result = redis_client.get(key)
    decoded_result = result.decode("utf-8")
    # check the retrieved item matches the item added to the cluster and print
    # the results
    if decoded_result == uuid_in:
        print(f"Success: Inserted {uuid_in}. Fetched {decoded_result} from MemoryDB.")
    else:
        raise Exception(f"Bad value retrieved. Expected {uuid_in}, got
 {decoded_result}")

    return "Fetched value from MemoryDB"
```

Esse código usa a `redis-py` biblioteca Python para colocar itens em seu cluster e recuperá-los. Esse código é usado `cachetools` para armazenar em cache os tokens IAM Auth gerados por 15 minutos. Para criar um pacote de implantação contendo `redis-py` `ecachetools`, execute as etapas a seguir.

No diretório do projeto que contém o arquivo do `app.py` código-fonte, crie um pacote de pastas para instalar as `cachetools` bibliotecas `redis-py` e.

```
mkdir package
```

Instale `redis-py` e `cachetools` use o pip.

```
pip install --target ./package redis
pip install --target ./package cachetools
```

Crie um arquivo.zip contendo as `cachetools` bibliotecas `redis-py` e. No Linux e no macOS, execute o comando a seguir. No Windows, use seu utilitário zip preferido para criar um arquivo.zip com as `cachetools` bibliotecas `redis-py` e na raiz.

```
cd package
zip -r ../my_deployment_package.zip
```

Adicione o código de função ao arquivo .zip. No Linux e no MacOS, execute o comando a seguir. No Windows, use seu utilitário zip preferido para adicionar app.py à raiz do seu arquivo.zip.

```
cd ..
zip my_deployment_package.zip app.py
```

Etapa 2.2: Criar o perfil do IAM (perfil de execução)

Anexe a política AWS gerenciada nomeada `AWSLambdaVPCAccessExecutionRole` à função.

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole"
```

Etapa 2.3: fazer upload do pacote de implantação (criar função do Lambda)

Nesta etapa, você cria a função Lambda (AccessMemoryDB) usando o comando AWS CLI create-function.

No diretório do projeto que contém o arquivo.zip do pacote de implantação, execute o seguinte comando do Lambda `create-function` CLI.

Para a opção de função, use o ARN da função de execução que você criou na etapa 2.2. Para o vpc-config, insira listas separadas por vírgulas das sub-redes da VPC padrão e o ID do grupo de segurança da VPC padrão. É possível encontrar esses valores no console do Amazon VPC. Para encontrar as sub-redes da sua VPC padrão, escolha Suas VPCs e, em seguida, escolha a VPC padrão da sua conta AWS . Para encontrar o grupo de segurança dessa VPC, acesse Segurança e escolha Grupos de segurança. Não se esqueça de selecionar a região us-east-1.

```
aws lambda create-function \
--function-name AccessMemoryDB  \
--region us-east-1 \
--zip-file fileb://my_deployment_package.zip \
--role arn:aws:iam::123456789012:role/memorydb-iam-auth-app \
--handler app.lambda_handler \
--runtime python3.12  \
--timeout 30 \
--vpc-config SubnetIds=comma-separated-vpc-subnet-ids,SecurityGroupIds=default-
security-group-id
```

Etapa 3: testar a função do Lambda

Nesta etapa, você invoca a função Lambda manualmente usando o comando invoke. Quando a função Lambda é executada, ela gera um UUID e o grava no ElastiCache cache que você especificou no seu código Lambda. Depois, a função do Lambda recupera o item do cache.

1. Invoque a função Lambda AccessMemory (DB) usando AWS Lambda o comando invoke.

   ```
   aws lambda invoke \
   --function-name AccessMemoryDB  \
   --region us-east-1 \
   output.txt
   ```

2. Verifique se a função do Lambda foi executada com êxito, da seguinte forma:

   - Analise o arquivo output.txt.
   - Verifique os resultados em CloudWatch Logs abrindo o CloudWatch console e escolhendo o grupo de registros para sua função (AccessRedis/aws/lambda/). O fluxo de logs deve conter uma saída semelhante à mostrada a seguir:

     ```
     Success: Inserted 826e70c5f4d2478c8c18027125a3e01e. Fetched
      826e70c5f4d2478c8c18027125a3e01e from MemoryDB.
     ```

- Analise os resultados no AWS Lambda console.

Etapa 4: limpar (opcional)

Para limpar, siga estas etapas.

Tópicos

Etapa 4.1: Excluir a função Lambda

```
aws lambda delete-function \
 --function-name AccessMemoryDB
```

Etapa 4.2: Excluir o cluster MemoryDB

Excluir o cluster.

```
aws memorydb delete-cluster \
 --cluster-name cluster-01
```

Remova o usuário e a ACL.

```
aws memorydb delete-user \
 --user-id iam-user-01

aws memorydb delete-acl \
 --acl-name iam-acl-01
```

Etapa 4.3: Remover a função e as políticas do IAM

```
aws iam detach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

```
aws iam detach-role-policy \
--role-name "memorydb-iam-auth-app" \
--policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole"

aws iam delete-role \
 --role-name "memorydb-iam-auth-app"

 aws iam delete-policy \
  --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

Pesquisa vetorial

Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

A pesquisa vetorial do MemoryDB amplia a funcionalidade do MemoryDB. Ela pode ser usada em conjunto com a funcionalidade existente do MemoryDB. As aplicações que não usam a pesquisa vetorial não são afetadas por sua presença. A pré-visualização da pesquisa vetorial está disponível no MemoryDB 7.1 versão 7.1 em diante nas seguintes regiões: Leste dos EUA (Norte da Virgínia e Ohio), Oeste dos EUA (Oregon), UE (Irlanda) e Ásia-Pacífico (Tóquio).

A pesquisa vetorial do Amazon MemoryDB para Redis simplifica a arquitetura da sua aplicação e, ao mesmo tempo, oferece uma pesquisa vetorial de alta velocidade. A pesquisa vetorial para MemoryDB é ideal para casos de uso em que performance e escala máximas são os critérios de seleção mais importantes. Você pode usar seus dados existentes do MemoryDB, ou a API do Redis, para criar casos de uso de machine learning e IA generativa, como geração aumentada de recuperação, detecção de anomalias, recuperação de documentos e recomendações em tempo real.

Tópicos

Visão geral sobre a pesquisa vetorial

Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

A pesquisa vetorial baseia-se na criação, na manutenção e no uso de índices. Cada operação de pesquisa vetorial especifica um único índice e está confinada a esse índice, ou seja, as operações em um índice não são afetadas pelas operações em nenhum outro índice. Com exceção das

operações para criar e destruir índices, qualquer número de operações pode ser emitido em qualquer índice a qualquer momento, o que significa que, no nível do cluster, várias operações em vários índices podem estar em andamento simultaneamente.

Índices individuais são objetos nomeados existentes em um namespace exclusivo, que é separado dos outros namespaces do Redis: chaves, funções etc. Cada índice é conceitualmente semelhante a uma tabela de banco de dados convencional, pois está estruturado em duas dimensões: coluna e linhas. Cada linha da tabela corresponde a uma chave do Redis. Cada coluna no índice corresponde a um membro ou a uma parte dessa chave. Neste documento, os termos chave, linha e registro são idênticos e usados de maneira intercambiável. Da mesma forma, os termos coluna, campo, caminho e membro são essencialmente idênticos e também são usados de maneira intercambiável.

Não há comandos especiais para adicionar, excluir ou modificar dados indexados. Em vez disso, os comandos HASH ou JSON existentes que modificam uma chave em um índice também atualizam automaticamente esse índice.

Tópicos

Índices e o espaço de chaves do Redis

Índices são construídos e mantidos em um subconjunto do espaço de chaves do Redis. Vários índices podem escolher subconjuntos separados ou sobrepostos do espaço de chaves do Redis, sem limitação. O espaço de chaves para cada índice é definido por uma lista de prefixos de chave que são fornecidos quando esse índice é criado. A lista de prefixos é opcional e, se omitida, todo o espaço de chaves do Redis fará parte desse índice. Os índices também são digitados, pois cobrem apenas as chaves que têm um tipo correspondente. Atualmente, apenas há suporte para índices JSON e HASH. Um índice HASH indexa somente as chaves HASH cobertas por sua lista de prefixos e, da mesma maneira, um índice JSON indexa somente as chaves JSON cobertas por sua lista de prefixos. As chaves na lista de prefixos do espaço de chaves de um índice que não têm o tipo designado são ignoradas e não afetam as operações de pesquisa.

Quando um comando HASH ou JSON modifica uma chave que está dentro de um espaço de chaves de um índice, esse índice é atualizado. Esse processo envolve a extração dos campos declarados para cada índice e a atualização do índice com o novo valor. O processo de atualização é feito em um thread em segundo plano, o que significa que os índices são consistentes apenas eventualmente com o conteúdo do espaço de chaves. Assim, a inserção ou atualização de uma chave não ficará visível nos resultados da pesquisa por um curto período. Durante períodos de grande carga do sistema e/ou forte mutação de dados, o atraso na visibilidade pode se tornar maior.

A criação de um índice é um processo em várias etapas. A primeira etapa é executar o comando FT.CREATE que define o índice. A execução bem-sucedida de uma criação inicia automaticamente a segunda etapa: o preenchimento. O processo de preenchimento é executado em um thread em segundo plano e verifica o espaço de chaves do Redis em busca de chaves que estejam na lista de prefixos do novo índice. Cada chave encontrada é adicionada ao índice. Por fim, todo o espaço de chaves é verificado, concluindo o processo de criação do índice. Enquanto o processo de preenchimento está em execução, são permitidas mutações das chaves indexadas, não há restrições, e o processo de preenchimento do índice não será concluído até que todas as chaves estejam devidamente indexadas. Tentativas de operações de consulta feitas enquanto um índice está sendo preenchido não são permitidas e serão encerradas com um erro. A conclusão do processo de preenchimento pode ser determinada pela saída do comando `FT.INFO` desse índice ('backfill_status').

Tipos de campos de índice

Cada campo (coluna) de um índice tem um tipo específico que é declarado quando esse índice é criado e um local dentro de uma chave. Para chaves HASH, a localização é o nome do campo dentro do HASH. Para chaves JSON, a localização é uma descrição do caminho do JSON. Quando uma chave é modificada, os dados associados aos campos declarados são extraídos, convertidos no tipo declarado e armazenados no índice. Se os dados estiverem ausentes ou não puderem ser convertidos com êxito no tipo declarado, esse campo será omitido do índice. Há quatro tipos de campos, conforme explicado a seguir:

- Campos numéricos contêm um único número. Para campos JSON, devem ser seguidas as regras numéricas de números JSON. Para HASH, espera-se que o campo contenha o texto ASCII de um número escrito no formato padrão para números fixos ou de pontos flutuantes. Independentemente da representação na chave, esse campo é convertido em um número de pontos flutuantes de 64 bits para armazenamento no índice. Campos numéricos podem ser usados com o operador de pesquisa de intervalo. Como os números subjacentes são armazenados em ponto flutuante com

suas limitações de precisão, as regras comuns sobre comparações numéricas para números de pontos flutuantes são aplicáveis.

- Campos de etiqueta contêm zero ou mais valores de etiqueta codificados como uma única string UTF-8. A string é analisada em valores de etiquetas usando um caractere separador (o padrão é uma vírgula, mas pode ser substituído) com espaços em branco à esquerda e à direita removidos. Qualquer número de valores de etiquetas pode estar contido em um único campo de etiqueta. Campos de etiquetas podem ser usados para filtrar consultas de equivalência de valores de etiquetas com comparação com ou sem distinção entre maiúsculas e minúsculas.
- Campos de texto contêm um blob de bytes que não precisa ser compatível com UTF-8. Esses campos podem ser usados para decorar resultados de consultas com valores significativos para a aplicação. Por exemplo, um URL ou o conteúdo de um documento etc.
- Campos vetoriais contêm um vetor de números, também conhecido como incorporação. Esses campos oferecem suporte à pesquisa do vizinho mais próximo (KNN) de vetores com tamanho fixo usando um algoritmo e uma métrica de distância especificados. No caso de índices HASH, o campo deve conter todo o vetor codificado em formato binário (little-endian IEEE 754). No caso de chaves JSON, o caminho deve fazer referência a uma matriz do tamanho correto preenchida com números. Quando uma matriz JSON é usada como um campo vetorial, a representação interna dessa matriz na chave JSON é convertida no formato exigido pelo algoritmo selecionado, reduzindo o consumo e a precisão da memória. Operações de leitura subsequentes usando os comandos JSON produzirão o valor de precisão reduzido.

Algoritmos de índice vetorial

São fornecidos dois algoritmos de índice vetorial:

- Flat: o algoritmo Flat é um processamento linear por força bruta de cada vetor no índice, produzindo respostas exatas dentro dos limites da precisão dos cálculos de distância. Devido ao processamento linear do índice, os tempos de execução desse algoritmo podem ser muito altos para índices grandes.
- HNSW (Hierarchical Navigable Small Worlds) — O algoritmo HNSW é uma alternativa que fornece uma aproximação da resposta correta em troca de tempos de execução substancialmente menores. O algoritmo é controlado por três parâmetros: `M`, `EF_CONSTRUCTION` e `EF_RUNTIME`. Os dois primeiros parâmetros são especificados no momento da criação do índice e não podem ser alterados. O parâmetro `EF_RUNTIME` tem um valor padrão que é especificado no momento da criação do índice, mas pode ser substituído mais tarde em qualquer operação de consulta individual. Esses três parâmetros interagem para equilibrar o consumo de memória e CPU durante

operações de ingestão e consulta, bem como para controlar a qualidade da aproximação de uma pesquisa KNN exata (conhecida como taxa de recall).

Ambos os algoritmos de pesquisa vetorial (Flat e HNSW) aceitam um parâmetro opcional `INITIAL_CAP`. Quando especificado, esse parâmetro pré-aloca memória para os índices, resultando na redução da sobrecarga de gerenciamento de memória e no aumento das taxas de ingestão de vetores.

Algoritmos de pesquisa vetorial, como o HNSW, podem não lidar de maneira eficiente com a exclusão ou substituição de vetores inseridos anteriormente. O uso dessas operações pode resultar no consumo excessivo de memória de índice e/ou degradação na qualidade do recall. A reindexação é um dos métodos para restaurar o uso e/ou a recuperação ideais da memória.

Expressão de consulta de pesquisa vetorial

Os comandos FT.SEARCH e FT.AGGREGATE exigem uma expressão de consulta. Essa expressão é um único parâmetro de cadeia de caracteres que é composto por um ou mais operadores. Cada operador usa um campo no índice para identificar um subconjunto das chaves no índice. Vários operadores podem ser combinados usando combinadores boolianos e parênteses para aprimorar ou restringir ainda mais o conjunto coletado de chaves (ou o conjunto de resultados).

Curinga

O operador curinga, o asterisco ('*'), corresponde a todas as chaves no índice.

Intervalo numérico

O operador de intervalo numérico tem a sintaxe a seguir.

```
<range-search> ::= '@' <numeric-field-name> ':' '[' <bound> <bound> ']'
<bound>  ::= <number> | '(' <number>
<number> ::= <integer> | <fixed-point> | <floating-point> | 'Inf' | '-Inf' | '+Inf'
```

O < numeric-field-name > deve ser um campo do tipo declarado`NUMERIC`. O limite é inclusivo por padrão, mas um parêntese de abertura inicial ['('] pode ser usado para tornar um limite exclusivo. A pesquisa de intervalo pode ser convertida em uma única comparação relacional (<, <=, >, >=) usando `Inf`, `+Inf` ou `-Inf` como um dos limites. Independentemente do formato numérico especificado (inteiro, ponto fixo, ponto flutuante, infinito), o número é convertido em ponto flutuante de 64 bits para realizar comparações, reduzindo a precisão de acordo.

Example Exemplos

```
@numeric-field:[0 10]                       // 0   <= <value> <= 10
@numeric-field:[(0 10]                      // 0   <  <value> <= 10
@numeric-field:[0 (10]                      // 0   <= <value> <  10
@numeric-field:[(0 (10]                     // 0   <  <value> <  10
@numeric-field:[1.5 (Inf]                   // 1.5 <= value
```

Comparação de etiquetas

O operador de comparação de etiquetas tem a seguinte sintaxe.

```
<tag-search> ::= '@' <tag-field-name> ':' '{' <tag> [ '|' <tag> ]* '}'
```

Se alguma das etiquetas no operador corresponder a qualquer uma das etiquetas no campo de etiqueta do registro, este último será incluído no conjunto de resultados. O campo criado por `<tag-field-name>` deve ser um campo do índice declarado com o tipo `TAG`. Exemplos de comparação de etiquetas são:

```
@tag-field:{ atag }
@tag-field: { tag1 | tag2 }
```

Combinações boolianas

Os conjuntos de resultados de um operador numérico ou de etiqueta podem ser combinados usando a lógica booliana: e/ou. Parênteses podem ser usados para agrupar operadores e/ou alterar a ordem de avaliação. A sintaxe dos operadores lógicos boolianos é:

```
<expression> ::= <phrase> | <phrase> '|' <expression> | '(' <expression> ')'
<phrase> ::= <term> | <term> <phrase>
<term> ::= <range-search> | <tag-search> | '*'
```

Vários termos combinados em uma frase são indicados com “e”. Várias frases combinadas com a barra vertical ('|') são indicadas com “ou”.

Pesquisa vetorial

O operador de pesquisa vetorial realiza uma pesquisa de K vizinho mais próximo de um índice de campo vetorial. A sintaxe de uma pesquisa vetorial é:

```
<vector-search> ::= <expression> '=>[KNN' <k> '@' <vector-field-name> '$' <parameter-
name> <modifiers> ']'
<modifiers> ::= [ 'EF_RUNTIME' <integer> ] [ 'AS' <distance-field-name>]
```

A pesquisa vetorial é aplicada somente aos vetores que atendem a `<expression>`, que pode ser qualquer combinação dos operadores definidos acima: curinga, pesquisa por intervalo, pesquisa por etiqueta e/ou combinações boolianas dos mesmos.

- <k> é um número inteiro que especifica o número de vetores vizinhos mais próximos a serem retornados.
- `<vector-field-name>` deve especificar um campo de tipo declarado `VECTOR`.
- O campo `<parameter-name>` especifica uma das entradas para a tabela `PARAM` do comando `FT.SEARCH` ou `FT.AGGREGATE`. Esse parâmetro é o valor vetorial de referência para cálculos de distância. O valor do vetor é codificado no valor `PARAM` no formato binário little-endian IEEE 754 (a mesma codificação de um campo vetorial HASH)
- Para índices vetoriais do tipo HNSW, a cláusula `EF_RUNTIME` opcional pode ser usada para substituir o valor padrão do parâmetro `EF_RUNTIME` que foi estabelecido quando o índice foi criado.
- O `<distance-field-name>` opcional fornece um nome de campo para o conjunto de resultados a fim de conter a distância calculada entre o vetor de referência e a chave localizada.

Comando INFO

A pesquisa vetorial aumenta o comando INFO do Redis com várias seções adicionais de estatísticas e contadores. Uma solicitação para recuperar a seção `SEARCH` recuperará todas as seções a seguir:

Seção **`search_memory`**

Nome	Descrição
search_used_memory_bytes	Número de bytes de memória consumidos em todas as estruturas de dados de pesquisa
search_used_memory_human	Versão legível por humanos do valor acima

Seção `search_index_stats`

Nome	Descrição
search_number_of_indexes	Número de índices criados
search_num_fulltext_indexes	Número de campos não vetoriais em todos os índices
search_num_vector_indexes	Número de campos vetoriais em todos os índices
search_num_hash_indexes	Número de índices em chaves de tipo HASH
search_num_json_indexes	Número de índices em chaves de tipo JSON
search_total_indexed_keys	Número total de chaves em todos os índices
vetores_indexados totais de pesquisa	Número total de vetores em todos os índices
search_total_indexed_hash_keys	Número total de chaves de tipo HASH em todos os índices
search_total_indexed_json_keys	Número total de chaves de tipo JSON em todos os índices
search_total_index_size	Bytes usados por todos os índices
search_total_fulltext_index_size	Bytes usados por estruturas de índices não vetoriais
search_total_vector_index_size	Bytes usados por estruturas de índices vetoriais
search_max_index_lag_ms	Atraso na ingestão durante a última atualização do lote de ingestão

Seção `search_ingestion`

Nome	Descrição
search_background_indexing_status	Status da ingestão. `NO_ACTIVITY` significa ocioso. Outros valores indicam que há chaves em processo de ingestão.
search_ingestion_paused	Exceto durante a reinicialização, sempre deve ser “no”.

Seção `search_backfill`

> **Note**
>
> Alguns dos campos documentados nesta seção apenas são visíveis quando um preenchimento está em andamento.

Nome	Descrição
search_num_active_backfills	Número de atividades atuais de preenchimento
search_backfills_paused	Exceto quando estiver sem memória, sempre deve ser “no”.
search_highest_backfill_progress_percentage	% de conclusão (0 a 100) do preenchimento mais concluído
search_lowest_backfill_progress_percentage	% de conclusão (0 a 100) do preenchimento menos concluído

Seção **`search_query`**

Nome	Descrição
search_num_active_queries	Número de comandos `FT.SEARCH` e `FT.AGGREGATE` em andamento

Segurança da pesquisa vetorial

Os mecanismos de segurança de ACL (Listas de controle de acesso) do Redis para acesso a comandos e dados são estendidos para controlar o recurso de pesquisa. Há suporte total para o controle de ACL de comandos de pesquisa individuais. É fornecida uma nova categoria de ACL, `@search`, e muitas das categorias existentes (`@fast`, `@read`, `@write` etc.) são atualizadas para incluir os novos comandos. Os comandos de pesquisa não modificam os dados de chaves, o que significa que o mecanismo de ACL existente para acesso de gravação é preservado. As regras de acesso para operações HASH e JSON não são modificadas pela presença de um índice. O controle normal de acesso em nível de chave ainda é aplicado a esses comandos.

Comandos de pesquisa com um índice também têm o acesso controlado por meio da ACL do Redis. Verificações de acesso são realizadas no nível do índice inteiro, e não no nível por chave. Isso significa que o acesso a um índice será concedido a um usuário somente se este tiver permissão para acessar todas as chaves possíveis na lista de prefixos do espaço de chaves desse índice. Em outras palavras, o conteúdo real de um índice não controla o acesso. Pelo contrário, é o conteúdo teórico de um índice, conforme definido pela lista de prefixos, que é usado para a verificação de segurança. Pode ser fácil criar uma situação em que um usuário tem acesso de leitura e/ou gravação a uma chave, mas não consegue acessar um índice contendo essa chave. Observe que somente o acesso para leitura ao espaço de chaves é necessário para criar ou usar um índice: a presença ou ausência do acesso para gravação não é levada em consideração.

Para obter mais informações sobre o uso de ACLs com o MemoryDB, consulte Autenticação de usuários com listas de controle de acesso (ACLs).

Atributos e limites da pesquisa vetorial

Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

Disponibilidade da pesquisa vetorial

A configuração MemoryDB habilitada para pesquisa vetorial é compatível com os tipos de nós R6g, R7g e T4g e está disponível nas seguintes AWS regiões: Leste dos EUA (Norte da Virgínia), Leste dos EUA (Ohio), Oeste dos EUA (Oregon), Ásia-Pacífico (Tóquio) e Europa (Irlanda).

Restrições paramétricas

A tabela a seguir mostra os limites de vários itens de pesquisa vetorial na prévia:

Item	Valor máximo
Número de dimensões em um vetor	32768
Número de índices que podem ser criados	10
Número de campos em um índice	50
Número de operações simultâneas de preenchimento do índice FT.CREATE	1
Cláusula FT.SEARCH e FT.AGGREGATE TIMEOUT (milissegundos)	60000
Número de estágios do pipeline no comando FT.AGGREGATE	32
Número de campos na cláusula FT.AGGREGATE LOAD	1024
Número de campos na cláusula FT.AGGREGATE GROUPBY	16
Número de campos na cláusula FT.AGGREGATE SORTBY	16
Número de parâmetros na cláusula FT.AGGREGATE PARAM	32
Parâmetro HNSW M	512

Item	Valor máximo
Parâmetro HNSW EF_CONSTRUCTION	4096
Parâmetro HNSW EF_RUNTIME	4096

Limites de escala

Atualmente, a pesquisa vetorial para MemoryDB está limitada a um único fragmento, e não há suporte para escala horizontal. A pesquisa vetorial oferece suporte para escala vertical e de réplica.

Restrições operacionais

Persistência e preenchimento de índices

A prévia da pesquisa vetorial mantém a definição de índices, mas não seu conteúdo. Portanto, qualquer solicitação ou evento operacional que faça com que um nó seja iniciado ou reiniciado exige que todos os índices sejam reconstruídos a partir de sua definição e dos principais dados de origem. O processo de reconstrução é iniciado automaticamente quando todos os dados são restaurados: nenhuma ação do usuário é necessária para iniciar isso. A reconstrução é executada como uma operação de preenchimento assim que os dados são restaurados. Isso é funcionalmente equivalente ao sistema executar automaticamente um comando FT.CREATE para cada índice definido. Observe que o nó fica disponível para operações do aplicativo assim que os dados são restaurados, mas provavelmente antes da conclusão do preenchimento do índice, o que significa que os preenchimentos voltarão a ficar visíveis para as aplicações. Por exemplo, comandos de pesquisa usando índices de preenchimento podem ser rejeitados. Para obter mais informações sobre preenchimento, consulte Visão geral sobre a pesquisa vetorial.

A conclusão do preenchimento do índice não é sincronizada entre um primário e uma réplica. Essa falta de sincronização pode se tornar inesperadamente visível para as aplicações e, portanto, é recomendável que estas verifiquem a conclusão do preenchimento nos primários e em todas as réplicas antes de iniciar as operações de pesquisa.

Importação/exportação de snapshots e migração em tempo real

A presença de índices de pesquisa em um arquivo RDB limita a transportabilidade compatível desses dados. O formato dos índices definidos pela edição da prévia somente é compreendido

por outro cluster da edição da prévia. Assim, arquivos RDB com índices de pesquisa só podem ser transferidos entre ou utilizados por clusters do MemoryDB habilitados para a prévia.

No entanto, arquivos RDB que não contêm índices não são restritos dessa maneira. Assim, os dados em um cluster de prévia podem ser exportados para clusters sem prévia, excluindo os índices antes da exportação.

Consumo de memória

A implementação atual de índices vetoriais consome aproximadamente o dobro da quantidade de memória que será consumida pela implementação da Disponibilidade geral.

Sem memória durante o preenchimento

Semelhante às operações de gravação do Redis, um preenchimento de índice está sujeito a limitações. out-of-memory Se a memória do Redis ficar cheia enquanto um preenchimento estiver em andamento, todos os preenchimentos serão pausados. Se a memória ficar disponível, o processo de preenchimento será retomado. Também é possível excluir e indexar quando o preenchimento é pausado devido à falta de memória.

Transações

Os comandos `FT.CREATE`, `FT.DROPINDEX`, `FT.ALIASADD`, `FT.ALIASDEL` e `FT.ALIASUPDATE` não podem ser executados em um contexto transacional, ou seja, não dentro de um bloco MULTI/EXEC ou dentro de um script LUA ou FUNCTION.

Casos de uso

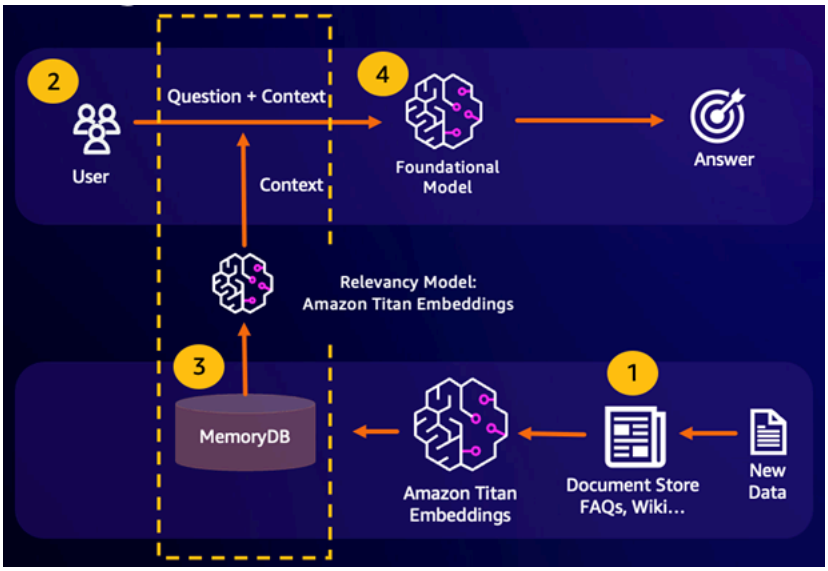
> Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

Veja a seguir estão os casos de uso da pesquisa vetorial.

Geração Aumentada de Recuperação (RAG)

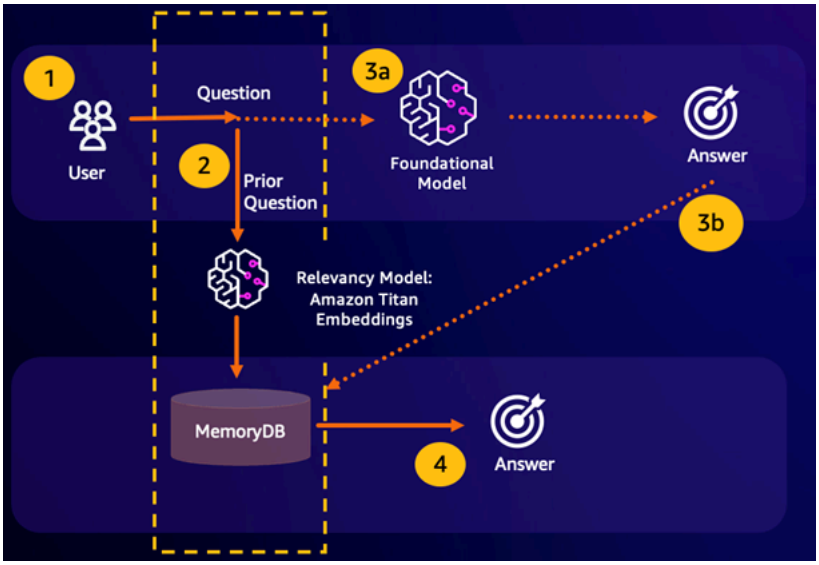
A Retrieval Augmented Generation (RAG) aproveita a pesquisa vetorial para recuperar passagens relevantes de um grande corpus de dados para ampliar um grande modelo de linguagem (LLM). Especificamente, um codificador incorpora o contexto de entrada e a consulta de pesquisa em

vetores e, em seguida, usa a pesquisa aproximada do vizinho mais próximo para encontrar passagens semanticamente semelhantes. Essas passagens recuperadas são concatenadas com o contexto original para fornecer informações adicionais relevantes ao LLM a fim de retornar uma resposta mais precisa ao usuário.



Memória de buffer do modelo de base (FM)

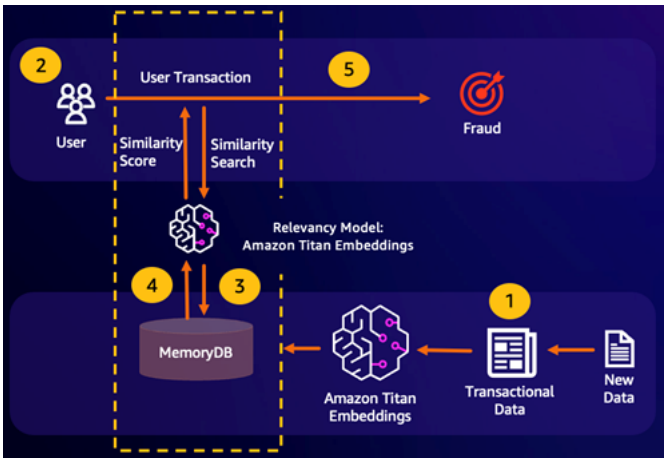
A memória de buffer do modelo de base (FM) é um processo para reduzir os custos computacionais armazenando resultados anteriores do FM. Ao reutilizar resultados anteriores de inferências anteriores em vez de recalculá-los, a memória de buffer do FM reduz a quantidade de computação necessária durante a inferência por meio dos FMs. Essa memória de buffer de FM permite que modelos de linguagens grandes respondam mais rapidamente com custos mais baixos devido às taxas de serviço do FM.



- Acerto da pesquisa semântica: se a consulta de um cliente for semanticamente semelhante com base em uma pontuação de semelhança definida com uma pergunta anterior, a memória de buffer do FM (MemoryDB) retornará a resposta à pergunta anterior na etapa 4 e não chamará o FM na etapa 3. Isso evitará a latência do modelo de base (FM) e os custos incorridos, proporcionando uma experiência mais rápida para o cliente.
- Erro na pesquisa semântica: se a consulta de um cliente não for semanticamente semelhante com base em uma pontuação de semelhança definida com uma consulta anterior, o cliente chamará o FM para fornecer uma resposta ao cliente na etapa 3a. A resposta gerada do FM será então armazenada como um vetor no MemoryDB para futuras consultas (etapa 3b), para minimizar os custos de FM em questões semanticamente semelhantes. Nesse fluxo, a etapa 4 não seria invocada, pois não havia uma pergunta semanticamente semelhante para a consulta original.

Detecção de fraudes

A detecção de fraudes, uma forma de detecção de anomalias, representa transações válidas como vetores enquanto compara as representações vetoriais de transações inéditas. A fraude é detectada quando essas transações inéditas têm baixa semelhança com os vetores que representam os dados transacionais válidos. Isso permite que a fraude seja detectada por meio da modelagem do comportamento normal, em vez de tentar prever todas as ocorrências possíveis de uma fraude. O MemoryDB permite que as organizações façam isso em períodos de alta throughput, com o mínimo de falsos positivos e latência de menos de 10 milissegundos.



Outros casos de uso

- Os mecanismos de recomendação podem encontrar produtos ou conteúdos semelhantes aos usuários representando itens como vetores. Os vetores são criados pela análise de atributos e padrões. Com base nos padrões e atributos do usuário, novos itens não vistos podem

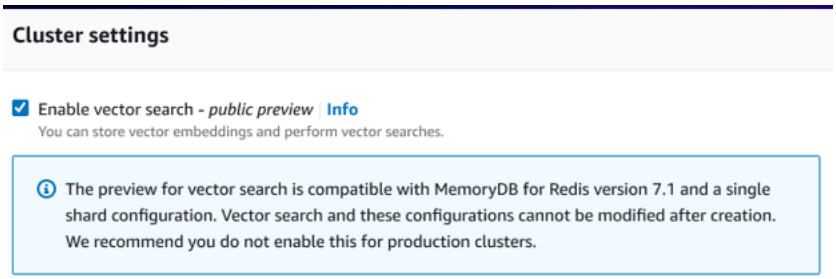
ser recomendados aos usuários, encontrando os vetores mais semelhantes já classificados positivamente alinhados ao usuário.

- Mecanismos de pesquisa de documentos representam documentos de texto como vetores densos de números, capturando o significado semântico. No momento da pesquisa, o mecanismo converte uma consulta de pesquisa em um vetor e encontra documentos com os vetores mais semelhantes a essa consulta usando a pesquisa aproximada do vizinho mais próximo. Essa abordagem de semelhança vetorial permite combinar documentos com base no significado, em vez de apenas combinar palavras-chave.

Usando o AWS Management Console

Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

Para criar um cluster habilitado para pesquisa vetorial no console, você precisa habilitar a pesquisa vetorial nas configurações do cluster. A pesquisa vetorial está disponível para o MemoryDB para Redis versão 7.1 e uma configuração de fragmento único.



Para obter mais informações sobre como usar a pesquisa vetorial com o AWS Management Console, consulteCriação de um cluster (console).

Usando o AWS Command Line Interface

Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

Para criar um cluster do MemoryDB habilitado para pesquisa vetorial, você pode usar o comando create-cluster do MemoryDB transmitindo um grupo de parâmetros `default.memorydb-redis7.search.preview` imutável para habilitar o modo de visualização dos recursos de pesquisa vetorial.

```
aws memorydb create-cluster \
  --cluster-name <value> \
  --node-type <value> \
  --engine redis \
  --engine-version 7.1 \
  --num-shards 1 \
  --acl-name <value> \
  --parameter-group-name default.memorydb-redis7.search.preview
```

Comandos de pesquisa vetorial

Veja a seguir uma lista dos comandos compatíveis com a pesquisa vetorial.

Tópicos

- FT.CREATE
- FT.SEARCH
- FT.AGGREGATE
- FT.DROPINDEX
- FT.INFO
- FT._LIST
- FT.ALIASADD
- FT.ALIASDEL
- FT.ALIASUPDATE
- FT._ALIASLIST
- FT.CONFIG GET
- FT.CONFIG HELP
- FT.CONFIG SET
- FT.PROFILE
- FT.EXPLAIN
- FT.EXPLAINCLI

FT.CREATE

Cria um índice e inicia um preenchimento desse índice. Para obter mais informações, consulte Visão geral da pesquisa vetorial para obter detalhes sobre a construção do índice.

Sintaxe

```
FT.CREATE <index-name>
ON HASH | JSON
[PREFIX <count> <prefix1> [<prefix2>...]]
SCHEMA
(<field-identifier> [AS <alias>]
  NUMERIC
| TAG [SEPARATOR <sep>] [CASESENSITIVE]
| TEXT
| VECTOR [HNSW|FLAT] <attr_count> [<attribute_name> <attribute_value>])

)+
```

Esquema

- Identificador de campo:
  - Para chaves de hash, o identificador de campo é um nome de campo.
  - Para chaves JSON, o identificador de campo é um caminho JSON.

  Para ter mais informações, consulte Tipos de campos de índice.

- Tipos de campos:
  - ETIQUETA: para obter mais informações, consulte Etiquetas.
  - NUMÉRICO: o campo contém um número.
  - TEXTO: o campo contém qualquer blob de dados.
  - VETOR: campo vetorial que oferece suporte para pesquisa vetorial.
    - Algoritmo: pode ser HNSW (Hierarchical Navigable Small World) ou FLAT (força bruta).
    - `attr_count`: número de atributos que serão passados como configuração do algoritmo, incluindo nomes e valores.
    - `{attribute_name} {attribute_value}`: pares de chave/valor específicos do algoritmo que definem a configuração do índice.

Para o algoritmo FLAT, os atributos são:

Obrigatório:

- DIM: número de dimensões no vetor.
- DISTANCE_METRIC: pode ser um dos [L2 | IP | COSINE].
- TYPE: Tipo de vetor. O único tipo com suporte é `FLOAT32`.

Opcional:

- INITIAL_CAP: capacidade vetorial inicial no índice que afeta o tamanho da alocação de memória do índice.

Para o algoritmo HNSW, os atributos são:

Obrigatório:

- TYPE: Tipo de vetor. O único tipo com suporte é `FLOAT32`.
- DIM: dimensão de vetor, especificada como um número inteiro positivo. Máximo: 32768
- DISTANCE_METRIC: pode ser um dos [L2 | IP | COSINE].

Opcional:

- INITIAL_CAP: capacidade vetorial inicial no índice que afeta o tamanho da alocação de memória do índice. O padrão é 1024.
- M: número máximo de bordas de saída permitidas para cada nó no gráfico em cada camada. Na camada zero, o número máximo de bordas de saída será 2M. O padrão é 16 e o máximo é 512.
- EF_CONSTRUCTION: controla o número de vetores examinados durante a construção do índice. Valores mais altos para esse parâmetro melhorarão a taxa de recall às custas de tempos mais longos de criação do índice. O valor padrão é 200. O valor máximo é 4096.
- EF_RUNTIME: controla o número de vetores examinados durante as operações de consulta. Valores mais altos para esse parâmetro podem gerar melhor recuperação à custa de tempos de consulta mais longos. O valor desse parâmetro pode ser substituído para cada consulta. O valor padrão é 10 O valor máximo é 4096.

Return

Retorna uma mensagem simples de texto OK ou uma resposta de erro.

Exemplos

> **Note**
>
> O exemplo a seguir usa argumentos nativos de redis-cli, como remoção de aspas e remoção de escape de dados, antes de enviá-los ao Redis. Para usar outros clientes de linguagem de programação (Python, Ruby, C# etc.), siga as regras de manipulação desses ambientes para lidar com strings e dados binários. Para obter mais informações sobre clientes compatíveis, consulte Ferramentas para desenvolver AWS

Example 1: Crie alguns índices

Crie um índice para vetores de tamanho 2

```
FT.CREATE hash_idx1 ON HASH PREFIX 1 hash: SCHEMA vec AS VEC VECTOR HNSW 6 DIM 2 TYPE
 FLOAT32 DISTANCE_METRIC L2
OK
```

Crie um índice JSON de 6 dimensões usando o algoritmo HNSW:

```
FT.CREATE json_idx1 ON JSON PREFIX 1 json: SCHEMA $.vec AS VEC VECTOR HNSW 6 DIM 6 TYPE
 FLOAT32 DISTANCE_METRIC L2
OK
```

Example Exemplo 2: preencher alguns dados

Os comandos a seguir são formatados para que possam ser executados como argumentos para o programa de terminal redis-cli. Os desenvolvedores que usam clientes de linguagem de programação (como Python, Ruby, C# etc.) precisarão seguir as regras de manipulação do ambiente para lidar com strings e dados binários.

Criando alguns dados de hash e json:

```
HSET hash:0 vec "\x00\x00\x00\x00\x00\x00\x00\x00"
HSET hash:1 vec "\x00\x00\x00\x00\x00\x00\x80\xbf"
JSON.SET json:0 . '{"vec":[1,2,3,4,5,6]}'
JSON.SET json:1 . '{"vec":[10,20,30,40,50,60]}'
JSON.SET json:2 . '{"vec":[1.1,1.2,1.3,1.4,1.5,1.6]}'
```

Observe o seguinte:

- As chaves dos dados de hash e JSON têm os prefixos de suas definições de índice.
- Os vetores estão nos caminhos apropriados das definições do índice.
- Os vetores de hash são inseridos como dados hexadecimais, enquanto os dados JSON são inseridos como números.
- Os vetores têm os comprimentos apropriados, as entradas bidimensionais do vetor hash têm dois valores flutuantes de dados hexadecimais, as entradas vetoriais json de seis dimensões têm seis números.

Example Exemplo 3: Excluir e recriar um índice

```
FT.DROPINDEX json_idx1
OK

FT.CREATE json_idx1 ON JSON PREFIX 1 json: SCHEMA $.vec AS VEC VECTOR FLAT 6 DIM 6 TYPE
 FLOAT32 DISTANCE_METRIC L2
OK
```

Observe que o novo índice JSON usa o algoritmo `FLAT` em vez do algoritmo `HNSW`. Observe também que ele reindexará os dados JSON existentes:

```
FT.SEARCH json_idx1 "*=>[KNN 100 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 3
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
6) "json:1"
7) 1) "__VEC_score"
   2) "9100"
   3) "$"
```

```
   4) "[{\"vec\":[10.0, 20.0, 30.0, 40.0, 50.0, 60.0]}]"
```

FT.SEARCH

Usa a expressão de consulta fornecida para localizar chaves em um índice. Uma vez localizados, a contagem e/ou o conteúdo dos campos indexados dentro dessas chaves podem ser retornados. Para obter mais informações, consulte Expressão de consulta de pesquisa vetorial.

Para criar dados para uso nesses exemplos, consulte o comando FT.CREATE.

Sintaxe

```
FT.SEARCH <index-name> <query>
[RETURN <token_count> (<field-identifier> [AS <alias>])+]
[TIMEOUT timeout]
[PARAMS <count> <name> <value> [<name> <value>]]
[LIMIT <offset> <count>]
[COUNT]
```

- RETURN: essa cláusula identifica quais campos de uma chave são retornados. A cláusula AS opcional em cada campo substitui o nome do campo no resultado. Somente campos que foram declarados para esse índice podem ser especificados.
- LIMIT: <offset><count>: essa cláusula fornece capacidade de paginação, pois somente as chaves que satisfazem os valores de deslocamento e contagem são retornadas. Se ela for omitida, o padrão será “LIMIT 0 10", ou seja, somente um máximo de 10 chaves serão retornadas.
- PARAMS: duas vezes o número de pares de valores-chave. Pares de chave/valor do parâmetro podem ser referenciados de dentro da expressão de consulta. Para obter mais informações, consulte Expressão de consulta de pesquisa vetorial.
- COUNT: essa cláusula suprime o retorno do conteúdo das chaves, somente o número de chaves é retornado. Ela é um alias para “LIMIT 0 0".

Return

Retorna uma matriz ou a resposta de erro.

- Se a operação for concluída com êxito, retornará uma matriz. O primeiro elemento é o número total de chaves correspondentes à consulta. Os elementos restantes são pares de nome de chave e lista de campos. A lista de campos é outra matriz que compreende pares de nomes e valores de campo.

- Se o índice estiver em andamento para preenchimento, o comando retornará imediatamente uma resposta de erro.
- Se o tempo limite for atingido, o comando retornará uma resposta de erro.

Exemplo: faça algumas pesquisas

> **Note**
>
> O exemplo a seguir usa argumentos nativos de redis-cli, como remoção de aspas e remoção de escape de dados, antes de enviá-los ao Redis. Para usar outros clientes de linguagem de programação (Python, Ruby, C# etc.), siga as regras de manipulação desses ambientes para lidar com strings e dados binários. Para obter mais informações sobre clientes compatíveis, consulte Ferramentas para desenvolver AWS

Uma pesquisa de hash

```
FT.SEARCH hash_idx1 "*=>[KNN 2 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00" DIALECT 2
1) (integer) 2
2) "hash:0"
3) 1) "__VEC_score"
   2) "0"
   3) "vec"
   4) "\x00\x00\x00\x00\x00\x00\x00\x00"
4) "hash:1"
5) 1) "__VEC_score"
   2) "1"
   3) "vec"
   4) "\x00\x00\x00\x00\x00\x00\x80\xbf"
```

Isso produz dois resultados, classificados por sua pontuação, que é a distância do vetor de consulta (inserido como hexadecimal).

Pesquisas JSON

```
FT.SEARCH json_idx1 "*=>[KNN 2 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 2
```

```
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
```

Produz os dois resultados mais próximos, classificados por sua pontuação. Observe que os valores vetoriais JSON são convertidos em flutuantes e o vetor de consulta ainda é um dado vetorial. Observe também que, como o parâmetro KNN é 2, há apenas dois resultados. Um valor maior retornará mais resultados:

```
FT.SEARCH json_idx1 "*=>[KNN 100 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 3
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
6) "json:1"
7) 1) "__VEC_score"
   2) "9100"
   3) "$"
   4) "[{\"vec\":[10.0, 20.0, 30.0, 40.0, 50.0, 60.0]}]"
```

FT.AGGREGATE

Um superconjunto do comando FT.SEARCH, permite um processamento adicional substancial das chaves selecionadas pela expressão de consulta.

Sintaxe

```
FT.AGGREGATE index query
  [LOAD * | [count field [field ...]]]
  [TIMEOUT timeout]
  [PARAMS count name value [name value ...]]
  [FILTER expression]
  [LIMIT offset num]
  [GROUPBY count property [property ...] [REDUCE function count arg [arg ...] [AS name]
 [REDUCE function count arg [arg ...] [AS name] ...]] ...]]
  [SORTBY count [ property ASC | DESC [property ASC | DESC ...]] [MAX num]]
  [APPLY expression AS name]
```

- Cláusulas FILTER, LIMIT, GROUPBY, SORTBY e APPLY podem ser repetidas várias vezes em qualquer ordem e ser misturadas livremente. São aplicados na ordem especificada com a saída de uma cláusula alimentando a entrada da próxima cláusula.
- Na sintaxe acima, uma “propriedade” é um campo declarado no comando FT.CREATE para esse índice OU a saída de uma cláusula APPLY ou função REDUCE anterior.
- A cláusula LOAD é restrita ao carregamento de campos que foram declarados no índice. “LOAD *” carregará todos os campos declarados no índice.
- As seguintes funções redutoras têm suporte: COUNT, COUNT_DISTINCTISH, SUM, MIN, MAX, AVG, STDDEV, QUANTILE, TOLIST, FIRST_VALUE e RANDOM_SAMPLE. Para obter mais informações, consulte Agregações.
- LIMIT <offset><count>: retém registros começando em <offset> e continuando por até <count>, todos os outros registros são descartados.
- PARAMS: duas vezes o número de pares de valores-chave. Pares de chave/valor do parâmetro podem ser referenciados de dentro da expressão de consulta. Para obter mais informações, consulte Expressão de consulta de pesquisa vetorial.

Return

Retorna uma matriz ou a resposta de erro.

- Se a operação for concluída com êxito, retornará uma matriz. O primeiro elemento é um número inteiro sem significado específico (deve ser ignorado). Os elementos restantes são os resultados gerados pelo último estágio. Cada elemento é uma matriz de nomes de campos e pares de valores.
- Se o índice estiver em andamento para preenchimento, o comando retornará imediatamente uma resposta de erro.

- Se o tempo limite for atingido, o comando retornará uma resposta de erro.

FT.DROPINDEX

Drop an index. A definição do índice e o conteúdo associado são excluídos. As chaves do Redis não são afetadas.

Sintaxe

```
FT.DROPINDEX <index-name>
```

Return

Retorna uma mensagem simples de texto OK ou uma resposta de erro.

FT.INFO

Sintaxe

```
FT.INFO <index-name>
```

A saída da página FT.INFO é uma matriz de pares de valores-chave, conforme descrito na tabela a seguir:

Chave	Tipo de valor	Descrição
nome_índice	string	Nome do índice
creation_timestamp	inteiro	Carimbo de data e hora da criação no estilo UNIX
key_type	string	HASH ou JSON
key_prefixes	matriz de strings	Prefixos principais para este índice
fields	matriz de informações de campo	Campos desse índice

Chave	Tipo de valor	Descrição
space_usage	inteiro	Bytes de memória usados por esse índice
fullext_space_usage	inteiro	Bytes de memória usados por campos não vetoriais
vector_space_usage	inteiro	Bytes de memória usados por campos vetoriais
num_docs	inteiro	Número de chaves atualmente contidas no índice
num_indexed_vectors	inteiro	Número de vetores atualment e contidos no índice
current_lag	inteiro	Atraso recente na ingestão (milissegundos)
backfill_status	string	Um dos seguintes: Concluído InProgres, Pausado ou Falha

A tabela a seguir descreve as informações de cada campo:

Chave	Tipo de valor	Descrição
Identifier	string	nome do campo
field_name	string	Nome do membro de hash ou caminho JSON
type	string	um dos seguintes: Numérico, Tag, Texto ou Vetor
option	string	ignorar

Se o campo for do tipo Vector, informações adicionais estarão presentes dependendo do algoritmo.

Para o algoritmo HNSW:

Chave	Tipo de valor	Descrição
algoritmo	string	HNSW
data_type	string	FLOAT32
distance_metric	string	um dos seguintes: L2, IP ou Cosine
capacidade_inicial	inteiro	Tamanho inicial do índice do campo vetorial
current_capacity	inteiro	Tamanho atual do índice do campo vetorial
maximum_edges	inteiro	Parâmetro M na criação
ef_construction	inteiro	Parâmetro EF_CONSTR UCTION na criação
ef_runtime	inteiro	Parâmetro EF_RUNTIME na criação

Para o algoritmo FLAT:

Chave	Tipo de valor	Descrição
algoritmo	string	FLAT
data_type	string	FLOAT32
distance_metric	string	um dos seguintes: L2, IP ou Cosine
capacidade_inicial	inteiro	Tamanho inicial do índice do campo vetorial

Chave	Tipo de valor	Descrição
current_capacity	inteiro	Tamanho atual do índice do campo vetorial

FT._LIST

Liste todos os índices.

Sintaxe

```
FT._LIST
```

Return

Retorna uma matriz de nomes de índice

FT.ALIASADD

Adicione um alias para um índice. O novo nome do alias pode ser usado em qualquer lugar em que seja necessário um nome de índice.

Sintaxe

```
FT.ALIASADD <alias> <index-name>
```

Return

Retorna uma mensagem simples de texto OK ou uma resposta de erro.

FT.ALIASDEL

Exclua um alias existente para um índice.

Sintaxe

```
FT.ALIASDEL <alias>
```

Return

Retorna uma mensagem simples de texto OK ou uma resposta de erro.

FT.ALIASUPDATE

Atualize um alias existente para apontar para um índice físico diferente. Esse comando afeta somente referências futuras ao alias. As operações atualmente em andamento (FT.SEARCH, FT.AGGREGATE) não são afetadas por esse comando.

Sintaxe

```
FT.ALIASUPDATE <alias> <index>
```

Return

Retorna uma mensagem simples de texto OK ou uma resposta de erro.

FT._ALIASLIST

Liste os aliases do índice.

Sintaxe

```
FT._ALIASLIST
```

Return

Retorna uma matriz do tamanho do número de aliases atuais. Cada elemento da matriz é o par alias/índice.

FT.CONFIG GET

Retorna o valor do parâmetro TIMEOUT.

Sintaxe

```
FT.CONFIG GET [* | <timeout>]
```

Return

Retorna o valor do parâmetro TIMEOUT.

FT.CONFIG HELP

Recuperar informações sobre o parâmetro TIMEOUT.

Sintaxe

```
FT.CONFIG HELP [* | <timeout>]
```

Return

Retorna informações sobre o parâmetro TIMEOUT

FT.CONFIG SET

Defina o parâmetro TIMEOUT. O valor padrão é 10.000 milissegundos.

> **Note**
>
> Os nomes dos parâmetros configuráveis não diferenciam maiúsculas de minúsculas.

Sintaxe

```
FT.CONFIG SET <timeout> <value>
```

Return

Retorna o valor da configuração TIMEOUT.

FT.PROFILE

Execute uma consulta e retorne informações de perfil sobre essa consulta.

Sintaxe

```
FT.PROFILE

<index>
```

```
SEARCH | AGGREGATE
[LIMITED]
QUERY <query ....>
```

Return

Uma matriz de dois elementos. O primeiro elemento é o resultado do comando `FT.SEARCH` ou `FT.AGGREGATE` cujo perfil foi definido. O segundo elemento é uma matriz de informações de desempenho e definição de perfil.

FT.EXPLAIN

Analise uma consulta e retorne informações sobre como essa consulta foi analisada.

Sintaxe

```
FT.EXPLAIN <index> <query>
```

Return

Uma string contendo os resultados analisados.

FT.EXPLAINCLI

O mesmo que o comando FT.EXPLAIN, exceto que os resultados são exibidos em um formato diferente, mais útil com o redis-cli.

Sintaxe

```
FT.EXPLAINCLI <index> <query>
```

Return

Uma string contendo os resultados analisados.

Segurança no MemoryDB para Redis

A segurança para com a nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se contará com um datacenter e uma arquitetura de rede criados para atender aos requisitos das organizações com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre a AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa produtos da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores de terceiros testam e verificam regularmente a eficácia da nossa segurança como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao MemoryDB, consulte Serviços da AWS no escopo por programa de conformidade.
- Segurança na nuvem: sua responsabilidade é determinada pelo serviço da AWS que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade dos dados, os requisitos da empresa e as leis e os regulamentos aplicáveis

Esta documentação ajuda a entender como aplicar o modelo de responsabilidade compartilhada ao usar o MemoryDB para Redis. Ela mostra como configurar o MemoryDB para atender aos objetivos de segurança e conformidade. Saiba também como usar outros produtos da AWS que ajudam a monitorar e proteger os recursos do MemoryDB.

Índice

Proteção de dados no MemoryDB para Redis

O modelo de responsabilidade compartilhada da AWS se aplica à proteção de dados no . Conforme descrito nesse modelo, a AWS é responsável por proteger a infraestrutura global que executa toda a Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados. Para mais informações sobre a proteção de dados na Europa, consulte o artigo AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja asConta da AWS credenciais da e configure as contas de usuário individuais com o AWS IAM Identity Center ou o AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA [multi-factor authentication]) com cada conta.
- Use SSL/TLS para se comunicar com os atributos da AWS. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure o registro em log das atividades da API e do usuário com o .AWS CloudTrail
- Use AWS as soluções de criptografia da , juntamente com todos os controles de segurança padrão dos Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar a AWS por meio de uma interface de linha de comandos ou uma API, use um endpoint do FIPS. Para ter mais informações sobre endpoints do FIPS, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Nome. Isso inclui trabalhar com os Serviços da AWS ou com outros usando o console, a API, a AWS CLI ou os SDKs da AWS. Quaisquer dados inseridos em tags ou campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você

fornecer um URL para um servidor externo, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse servidor.

Segurança de dados no MemoryDB para Redis

Para ajudar a manter seus dados seguros, o MemoryDB para Redis e o Amazon EC2 fornecem mecanismos de proteção contra o acesso não autorizado aos seus dados no servidor.

O MemoryDB também fornece atributos de criptografia para dados em clusters:

- A criptografia em trânsito criptografa seus dados sempre que eles estão se movendo de um lugar para outro, como entre os nós no seu cluster ou entre seu cluster e o aplicativo.
- A criptografia em repouso criptografa o log de transações e os dados no disco durante as operações de snapshot.

É possível usar também o Autenticação de usuários com listas de controle de acesso (ACLs) para controlar o acesso do usuário aos seus clusters.

Tópicos

Criptografia em repouso no MemoryDB

Para ajudar a manter seus dados seguros, o MemoryDB para Redis e o Amazon S3 oferecem diferentes maneiras de restringir o acesso aos dados em seus clusters. Para obter mais informações, consulte MemoryDB e Amazon VPC e Gerenciamento de identidade e acesso no MemoryDB para Redis.

A criptografia em repouso do MemoryDB está sempre ativada para aumentar a segurança dos dados por meio da criptografia de dados persistentes. Ele criptografa os seguintes aspectos:

- Dados no log de transações
- Disco durante operações de sincronização, snapshot e swap
- Snapshots armazenados no Amazon S3

O MemoryDB oferece criptografia padrão (gerenciada pelo serviço) em repouso, bem como a capacidade de usar suas próprias chaves raiz simétricas gerenciadas pelo cliente no Key Management Service (KMS) da AWS.

Os dados armazenados em Solid-State Drives (SSDs – Unidades de estado sólido) em clusters habilitados para classificação de dados em níveis sempre são criptografados por padrão.

Para obter informações sobre criptografia em trânsito, consulte Criptografia em trânsito (TLS) do MemoryDB

Tópicos

Uso de chaves gerenciadas pelo cliente do KMS da AWS

O MemoryDB oferece suporte a chaves raiz simétricas gerenciadas pelo cliente (chave KMS) para criptografia em repouso. Chaves KMS gerenciadas pelo cliente são chaves de criptografia que você cria, detém e gerencia na sua conta da AWS. Para obter mais informações, consulte Chaves raiz do cliente no Guia do desenvolvedor do serviço de gerenciamento de chaves da AWS. As chaves devem ser criadas no KMS da AWS antes de poderem ser usadas com o MemoryDB.

Para saber como criar chaves de raiz do KMS da AWS, consulte Criação de chaves no Guia do desenvolvedor do serviço de gerenciamento de chaves da AWS.

O MemoryDB permite a integração com o AWS KMS. Para obter mais informações, consulte Uso de concessões no Guia do desenvolvedor do serviço de gerenciamento de chaves da AWS. Não é necessária nenhuma ação do cliente para ativar a integração do MemoryDB com o KMS da AWS.

A chave de condição `kms:ViaService` limita o uso de uma chave do KMS AWS para solicitações provenientes de serviços da AWS especificados. Para usar `kms:ViaService` com o MemoryDB, inclua os dois nomes do ViaService no valor da chave de condição: `memorydb.amazon_region.amazonaws.com`. Para obter mais informações, consulte kms:ViaService.

Você pode usar o AWS CloudTrail para rastrear as solicitações que o MemoryDB envia para o AWS Key Management Service em seu nome. Todas as chamadas de API para o AWS Key Management Service relacionadas a chaves gerenciadas pelo cliente têm logs do CloudTrail correspondentes. Você também pode ver as concessões que o MemoryDB cria chamando a chamada da API ListGrants KMS.

Quando um cluster é criptografado usando uma chave gerenciada pelo cliente, todos os snapshots do cluster são criptografados da seguinte forma:

- Os snapshots diários automáticos são criptografados usando a chave gerenciada pelo cliente associada ao cluster.
- O snapshot final criado quando o cluster é excluído também é criptografado usando a chave gerenciada pelo cliente associada ao cluster.
- Os snapshots criados manualmente são criptografados por padrão para usar a chave KMS associada ao cluster. Você pode substituir escolhendo outra chave gerenciada pelo cliente.
- A cópia de um snapshot tem como padrão o uso da chave gerenciada pelo cliente associada ao snapshot de origem. Você pode substituir escolhendo outra chave gerenciada pelo cliente.

> **Note**
>
> - As chaves gerenciadas pelo cliente não podem ser usadas ao exportar snapshots para o bucket do Amazon S3 selecionado. No entanto, todos os snapshots exportados para o Amazon S3 são criptografados usando a criptografia do lado do servidor. Você pode optar por copiar o arquivo de snapshot para um novo objeto S3 e criptografar usando uma chave KMS gerenciada pelo cliente, copiar o arquivo para outro bucket S3 configurado com criptografia padrão usando uma chave KMS ou alterar uma opção de criptografia no próprio arquivo.

- Você também pode usar chaves gerenciadas pelo cliente para criptografar snapshots criados manualmente que não usam chaves gerenciadas pelo cliente para criptografia. Com essa opção, o arquivo de snapshot armazenado no Amazon S3 é criptografado usando uma chave KMS, mesmo que os dados não sejam criptografados no cluster original.

A restauração a partir de um snapshot permite que você escolha entre as opções de criptografia disponíveis, de forma semelhante às opções de criptografia disponíveis ao criar um novo cluster.

- Se você excluir a chave ou desativar a chave e revogar as concessões para a chave que usou para criptografar um cluster, o cluster se tornará irrecuperável. Em outras palavras, ele não poderá ser modificado nem recuperado depois de uma falha de hardware.AWS O KMS exclui as chaves raiz somente depois de um período de espera de pelo menos sete dias. Depois que a chave for excluída, você poderá usar uma chave gerenciada pelo cliente diferente para criar um snapshot para fins de arquivamento.
- A rotação automática de chaves preserva as propriedades das chaves raiz do KMS da AWS, de modo que a rotação não afeta sua capacidade de acessar os dados do MemoryDB. Os clusters criptografados do MemoryDB não são compatíveis com a rotação manual de chaves, o que envolve a criação de uma nova chave raiz e a atualização de todas as referências à chave antiga. Para saber mais, consulte Rotação das chaves raiz do cliente no Guia do desenvolvedor do Key Management Service daAWS.
- A criptografia de um cluster do MemoryDB usando a chave KMS requer uma concessão por cluster. Essa concessão é usada durante toda a vida útil do cluster. Além disso, uma concessão por snapshot é usada durante a criação do snapshot. Essa concessão é suspensa quando o snapshot é criado.
- Para obter mais informações sobre concessões e limites do KMS da AWS, consulte Cotas no Guia do desenvolvedor do Key Management Service da AWS.

Consulte também

- Criptografia em trânsito (TLS) do MemoryDB
- MemoryDB e Amazon VPC
- Gerenciamento de identidade e acesso no MemoryDB para Redis

Criptografia em trânsito (TLS) do MemoryDB

Para ajudar a manter seus dados seguros, o MemoryDB para Redis e o Amazon EC2 fornecem mecanismos de proteção contra o acesso não autorizado aos seus dados no servidor. Ao fornecer a capacidade de criptografia em trânsito, o MemoryDB oferece uma ferramenta que pode ser usada para ajudar a proteger seus dados quando eles estiverem sendo transferidos de um local para outro. Por exemplo, você pode mover dados de um nó primário para um nó de réplica de leitura em um cluster ou entre o cluster e o aplicativo.

Tópicos

Visão geral da criptografia em trânsito

A criptografia em trânsito do MemoryDB para Redis é um atributo que aumenta a segurança de seus dados em seus pontos mais vulneráveis, quando estão em trânsito de um local para outro.

A criptografia em trânsito do MemoryDB implementa os seguintes atributos:

- Conexões criptografadas – as conexões do servidor e do cliente são criptografadas por Transport Layer Security (TLS).
- Replicação criptografada: os dados em movimento entre um nó primário e nós de réplica são criptografados.
- Autenticação do servidor: os clientes podem autenticar que estão conectados ao servidor certo.

A partir de 20/07/2023, o TLS 1.2 é a versão mínima suportada para clusters novos e existentes. Use este link para saber mais sobre o TLS 1.2 em AWS.

Para obter mais informações sobre como se conectar aos clusters do MemoryDB, consulte Conectando-se aos nós do MemoryDB usando redis-cli.

Consulte também

- Criptografia em repouso no MemoryDB
- Autenticação de usuários com listas de controle de acesso (ACLs)
- MemoryDB e Amazon VPC

- Gerenciamento de identidade e acesso no MemoryDB para Redis

Autenticação de usuários com listas de controle de acesso (ACLs)

Você pode autenticar usuários com listas de controle de acesso (ACLs).

As ACLs permitem que você controle o acesso ao cluster agrupando usuários. Essas listas de controle de acesso são projetadas como uma maneira de organizar o acesso aos clusters.

Com as ACLs, você cria usuários e atribui a eles permissões específicas usando uma string de acesso, conforme descrito na próxima seção. Você atribui os usuários a listas de controle de acesso alinhadas a uma função específica (administradores, recursos humanos) que, em seguida, são implantadas em um ou mais clusters do MemoryDB. Ao fazer isso, você pode estabelecer limites de segurança entre clientes usando o mesmo cluster ou clusters do MemoryDB e impedir que os clientes acessem os dados uns dos outros.

As ACLs foram projetadas para oferecer suporte à introdução da ACL do Redis no Redis 6. Quando você usa ACLs com o cluster do MemoryDB, há algumas limitações:

- Não é possível especificar senhas em uma string de acesso. Você define senhas com CreateUsernossas UpdateUserchamadas.
- Para direitos de usuário, você passa `on` e `off` como parte da string de acesso. Se nenhum deles for especificado na string de acesso, o usuário é atribuído com `off` e não tem direitos de acesso ao cluster.
- Você não pode usar comandos proibidos. Se você especificar um comando proibido, será emitida uma exceção. Para obter uma lista desses comandos, consulte Comandos restritos do Redis.
- Não é possível usar o comando `reset` como parte de uma string de acesso. Você especifica as senhas com parâmetros de API e o MemoryDB gerencia as senhas. Assim, você não pode usar `reset` porque ele removeria todas as senhas de um usuário.
- O Redis 6 apresenta o comando ACL LIST. Esse comando retorna uma lista de usuários junto com as regras da ACL aplicadas a cada usuário. O MemoryDB oferece suporte ao comando `ACL LIST`, mas não inclui suporte para hashes de senha como o Redis faz. Com o MemoryDB, você pode usar a DescribeUsersoperação para obter informações semelhantes, incluindo as regras contidas na string de acesso. No entanto, DescribeUsersnão recupera a senha do usuário.

Outros comandos somente de leitura suportados pelo MemoryDB incluem ACL WHOAMI, ACL USERS e ACL CAT. O MemoryDB não oferece suporte a nenhum outro comando ACL baseado em gravação.

O uso de ACLs com o MemoryDB é descrito em mais detalhes a seguir.

Tópicos

- Especificação de permissões usando uma string de acesso
- Recursos de pesquisa vetorial
- Aplicação de ACLs a um cluster para MemoryDB

Especificação de permissões usando uma string de acesso

Para especificar permissões para um cluster MemoryDB, você cria uma string de acesso e a atribui a um usuário usando o AWS CLI ou. AWS Management Console

As strings de acesso são definidas como uma lista de regras delimitadas por espaço que são aplicadas ao usuário. Eles definem quais comandos um usuário pode executar e em quais chaves um usuário pode operar. Para executar um comando, um usuário deve ter acesso ao comando que está sendo executado e todas as chaves que estão sendo acessadas pelo comando. As regras são aplicadas da esquerda para a direita cumulativamente, e uma string mais simples pode ser usada em vez da fornecida se houver redundâncias na string fornecida.

Para obter informações sobre a sintaxe das regras ACL, consulte ACL.

No exemplo a seguir, a string de acesso representa um usuário ativo com acesso a todas as chaves e comandos disponíveis.

```
on ~* &* +@all
```

A sintaxe da cadeia de acesso é dividida da seguinte forma:

- on: o usuário é um usuário ativo.
- ~*: o acesso é dado a todas as chaves disponíveis.
- &*— O acesso é concedido a todos os canais do pubsub.
- +@all: o acesso é dado a todos os comandos disponíveis.

As configurações anteriores são as menos restritivas. Você pode modificar essas configurações para torná-las mais seguras.

No exemplo a seguir, a string de acesso representa um usuário com acesso restrito ao acesso de leitura em chaves que começam com o keyspace "app::"

```
on ~app::* -@all +@read
```

Você pode refinar mais essas permissões listando comandos aos quais o usuário tem acesso:

`+`*`command1`*: o acesso do usuário aos comandos é limitado a *`command1`*.

`+@category`: o acesso do usuário é limitado a uma categoria de comandos.

Para obter informações sobre como atribuir uma string de acesso a um usuário, consulte Criação de usuários e listas de controle de acesso com o console e a CLI.

Se você estiver migrando uma workload existente para o MemoryDB, poderá recuperar a string de acesso chamando `ACL LIST`, excluindo o usuário e quaisquer hashes de senha.

Recursos de pesquisa vetorial

> **Note**
>
> Esse recurso está em prévia de lançamento para o MemoryDB para Redis e sujeito a alterações.

Para a Pesquisa vetorial, todos os comandos de pesquisa pertencem à categoria `@search`, e as categorias `@read`, `@write`, `@fast` e `@slow` existentes são atualizadas para incluir comandos de pesquisa. Se um usuário não tiver acesso a uma categoria, ele não terá acesso aos comandos dentro dela. Por exemplo, se ele não tiver acesso a `@search`, não poderá executar comandos relacionados a pesquisas.

A tabela a seguir indica o mapeamento de comandos de pesquisa as categorias apropriadas.

Comandos do VSS	@read	@write	@fast	@slow
`FT.CREATE`		Y	Y	

Comandos do VSS	@read	@write	@fast	@slow
FT.DROPINDEX		Y	Y	
FT.LIST	Y			Y
FT.INFO	Y		Y	
FT.SEARCH	Y			Y
FT.AGGREGATE	Y			Y
FT.PROFILE	Y			Y
FT.ALIASADD		Y	Y	
FT.ALIASDEL		Y	Y	
FT.ALIASUPDATE		Y	Y	
FT._ALIASLIST	Y			Y
FT.EXPLAIN	Y		Y	
FT.EXPLAINCLI	Y		Y	
FT.CONFIG	Y		Y	

Aplicação de ACLs a um cluster para MemoryDB

Para usar as ACLs do MemoryDB, siga as etapas a seguir:

1. Crie um ou mais usuários.
2. Crie uma ACL e adicione usuários à lista.
3. Atribua a ACL a um cluster.

Essas etapas estão descritas em detalhes a seguir.

Tópicos

Criação de usuários e listas de controle de acesso com o console e a CLI

As informações de usuário para as ACLs são um nome de usuário e, opcionalmente, uma senha e uma string de acesso. A string de acesso fornece o nível de permissão em chaves e comandos. O nome é exclusivo para o usuário e é passado para o mecanismo.

Verifique se as permissões do usuário fornecidas fazem sentido com a finalidade pretendida da ACL. Por exemplo, se você criar uma ACL chamada `Administrators`, qualquer usuário que você adicionar a esse grupo deve ter sua string de acesso definida para acesso total a chaves e comandos. Para usuários em uma ACL de `e-commerce`, você pode definir suas strings de acesso para somente leitura.

O MemoryDB configura automaticamente um usuário padrão por conta com um nome de usuário `"default"`. Ele não será associado a nenhum cluster, a menos que seja explicitamente adicionado a uma ACL. Você não pode excluir ou modificar esse usuário. Esse usuário destina-se à compatibilidade com o comportamento padrão das versões anteriores do Redis e tem uma string de acesso que permite chamar todos os comandos e acessar todas as chaves.

Uma ACL imutável de “acesso aberto” será criada para cada conta que contém o usuário padrão. Essa é a única ACL da qual o usuário padrão pode se tornar membro. Ao criar um cluster, você deve selecionar uma ACL para associar ao cluster. Embora você tenha a opção de aplicar a ACL de “acesso aberto” com o usuário padrão, é altamente recomendável criar uma ACL com usuários que tenham permissões restritas às suas necessidades comerciais.

Os clusters que não têm o TLS ativado devem usar a ACL de “acesso aberto” para fornecer uma autenticação aberta.

As ACLs podem ser criadas sem usuários. Uma ACL vazia não teria acesso a um cluster e só pode ser associada a clusters habilitados para TLS.

Ao criar um usuário, você pode configurar até duas senhas. Quando você modifica uma senha, todas as conexões existentes para os clusters são mantidas.

Em particular, esteja ciente dessas restrições de senha de usuário ao usar as ACLs com o MemoryDB:

- As senhas devem ter de 16 a 128 caracteres imprimíveis.
- Os seguintes caracteres não alfanuméricos não são permitidos: `, "" / @`.

Gerenciamento de usuários com o console e a CLI

Criação de usuários (console)

Para criar usuários no console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Usuários.
3. Escolha Criar usuário
4. Na página Criar usuário, insira um Nome.

   As restrições de nomenclatura de cluster são as seguintes:

   - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
   - Deve começar com uma letra.
   - Não podem conter dois hifens consecutivos.
   - Não podem terminar com um hífen.
5. Em Senhas, você pode inserir até duas senhas.
6. Em String de acesso, insira uma cadeia de caracteres de acesso. A string de acesso define o nível de permissão para quais chaves e comandos o usuário é permitido.
7. Para tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar seus usuários ou monitorar seus AWS custos.

8. Escolha Criar.

Criando um usuário usando o AWS CLI

Para criar um usuário usando a CLI

- Use o comando create-user para criar um usuário.

  Para Linux, macOS ou Unix:

```
aws memorydb create-user \
  --user-name user-name-1 \
  --access-string "~objects:* ~items:* ~public:*" \
  --authentication-mode \
        Passwords="abc",Type=password
```

  Para Windows:

```
aws memorydb create-user ^
  --user-name user-name-1 ^
  --access-string "~objects:* ~items:* ~public:*" ^
  --authentication-mode \
        Passwords="abc",Type=password
```

Modificação de um usuário (console)

Para modificar usuários no console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Usuários.
3. Escolha o botão de opção ao lado do usuário que você deseja modificar e escolha Ações->Modificar
4. Se você quiser modificar uma senha, escolha o botão de opção Modificar senhas. Observe que, se você tiver duas senhas, deverá inserir as duas ao modificar uma delas.
5. Se você estiver atualizando a string de acesso, insira a nova.

6. Escolha Modificar.

Modificando um usuário usando AWS CLI

Para modificar um usuário usando a CLI

1. Use o comando update-user para modificar um usuário.
2. Quando um usuário é modificado, as listas de controle de acesso associadas ao usuário são atualizadas, juntamente com quaisquer clusters associados à ACL. Todas as conexões existentes são mantidas. Veja os exemplos a seguir.

   Para Linux, macOS ou Unix:

   ```
   aws memorydb update-user \
     --user-name user-name-1 \
     --access-string "~objects:* ~items:* ~public:*"
   ```

   Para Windows:

   ```
   aws memorydb update-user ^
     --user-name user-name-1 ^
     --access-string "~objects:* ~items:* ~public:*"
   ```

Visualizar detalhes do usuário (console)

Para visualizar os detalhes do usuário no console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Usuários.
3. Escolha o usuário em Nome de usuário ou use a caixa de pesquisa para localizar o usuário.
4. Em Configurações do usuário, você pode revisar a string de acesso, a contagem de senhas, o status e o nome do recurso da Amazon (ARN) do usuário.
5. Em Listas de controle de acesso (ACL), você pode revisar a ACL à qual o usuário pertence.
6. Em Tags, você pode revisar todas as tags associadas ao usuário.

Visualizando detalhes do usuário usando o AWS CLI

Use o comando describe-users para ver os detalhes de um usuário.

```
aws memorydb describe-users \
  --user-name my-user-name
```

Exclusão de um usuário (Console)

Para excluir usuários no console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Usuários.
3. Escolha o botão de opção ao lado do usuário que você deseja modificar e escolha Ações->Excluir
4. Para confirmar, digite `delete` na caixa de texto de confirmação e, em seguida, selecione Excluir.
5. Para cancelar, escolha Cancelar.

Excluindo um usuário usando o AWS CLI

Para excluir um usuário usando a CLI

- Use o comando delete-user para excluir um usuário.

  A conta é excluída e removida de todas as listas de controle de acesso às quais pertence. Veja um exemplo a seguir.

  Para Linux, macOS ou Unix:

  ```
  aws memorydb delete-user \
    --user-name user-name-2
  ```

  Para Windows:

  ```
  aws memorydb delete-user ^
    --user-name user-name-2
  ```

Gerenciamento de listas de controle de acesso com o console e a CLI

Você pode criar listas de controle de acesso para organizar e controlar o acesso de usuários a um ou mais clusters, conforme mostrado a seguir.

Use o procedimento a seguir para gerenciar as listas de controle de acesso usando o console.

Criação de uma lista de controle de acesso (ACL) (console)

Como criar uma Lista de controle de acesso usando o console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Listas de controle de acesso (ACL).
3. Escolha Criar ACL.
4. Na página Criar lista de controle de acesso (ACL), insira o nome da ACL.

   As restrições de nomenclatura de cluster são as seguintes:

   - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
   - Deve começar com uma letra.
   - Não podem conter dois hifens consecutivos.
   - Não podem terminar com um hífen.
5. Em Usuários selecionados, siga um destes procedimentos:

   a. Crie um novo usuário selecionando Criar usuário

   b. Adicione usuários escolhendo Gerenciar e, em seguida, selecionando usuários na caixa de diálogo Gerenciar usuários, depois selecione Escolher.
6. Para tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar suas ACLs ou monitorar seus AWS custos.
7. Escolha Criar.

Criando uma Lista de Controle de Acesso (ACL) usando o AWS CLI

Use os procedimentos a seguir para criar uma lista de controle de acesso usando a CLI.

Para criar uma nova ACL e adicionar um usuário usando a CLI

- Use o comando create-acl para criar uma ACL.

  Para Linux, macOS ou Unix:

```
aws memorydb create-acl \
  --acl-name "new-acl-1" \
  --user-names "user-name-1" "user-name-2"
```

  Para Windows:

```
aws memorydb create-acl ^
  --acl-name "new-acl-1" ^
  --user-names "user-name-1" "user-name-2"
```

Modificação de uma lista de controle de acesso (ACL) (console)

Para modificar uma lista de controle de acesso usando o console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Listas de controle de acesso (ACL).
3. Escolha a ACL que você deseja modificar e escolha Modificar
4. Na página Modificar, em Usuários selecionados, faça o seguinte:
   a. Crie um novo usuário escolhendo Criar usuário para adicionar à ACL.
   b. Adicione ou remova usuários escolhendo Gerenciar e, em seguida, selecionando ou desmarcando usuários na caixa de diálogo Gerenciar usuários e depois, selecionando Escolher.
5. Na página Criar lista de controle de acesso (ACL), insira o nome da ACL.

   As restrições de nomenclatura de cluster são as seguintes:

   - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
   - Deve começar com uma letra.
   - Não podem conter dois hifens consecutivos.

- Não podem terminar com um hífen.

6. Em Usuários selecionados, siga um destes procedimentos:

   a. Crie um novo usuário selecionando Criar usuário

   b. Adicione usuários escolhendo Gerenciar e, em seguida, selecionando usuários na caixa de diálogo Gerenciar usuários, depois selecione Escolher.

7. Escolha Modificar para salvar suas alterações ou Cancelar para descartá-las.

Modificando uma Lista de Controle de Acesso (ACL) usando o AWS CLI

Para modificar uma ACL adicionando novos usuários ou removendo membros atuais usando a CLI

- Use o comando update-acl para modificar uma ACL.

  Para Linux, macOS ou Unix:

```
aws memorydb update-acl --acl-name new-acl-1 \
--user-names-to-add user-name-3 \
--user-names-to-remove user-name-2
```

  Para Windows:

```
aws memorydb update-acl --acl-name new-acl-1 ^
--user-names-to-add user-name-3 ^
--user-names-to-remove user-name-2
```

> **Note**
>
> Quaisquer conexões abertas pertencentes a um usuário removido de uma ACL são encerradas por este comando.

Visualização de detalhes da Lista de controle de acesso (ACL) (console)

Para ver os detalhes da ACL no console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.

2. No painel de navegação à esquerda, escolha Listas de controle de acesso (ACL).
3. Escolha a ACL sob nome da ACL ou use a caixa de pesquisa para localizar a ACL.
4. Em Usuários, você pode revisar a lista de usuários associados à ACL.
5. Em Clusters associados, você pode revisar o cluster ao qual a ACL pertence.
6. Em Tags, você pode revisar todas as tags associadas à ACL.

Exibindo listas de controle de acesso (ACL) usando o AWS CLI

Use o comando describe-acls para ver detalhes de uma ACL.

```
aws memorydb describe-acls \
  --acl-name test-group
```

Excluindo uma Lista de controle de acesso (ACL) (console)

Para excluir uma lista de controle de acesso usando o console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação à esquerda, escolha Listas de controle de acesso (ACL).
3. Escolha a ACL que você deseja modificar e, em seguida, escolha Excluir
4. Na página Excluir, insira `delete` na caixa de confirmação e escolha Excluir; ou Cancelar para evitar a exclusão da ACL.

A própria ACL, não os usuários pertencentes ao grupo, é excluída.

Excluindo uma Lista de Controle de Acesso (ACL) usando o AWS CLI

Para excluir uma ACL usando a CLI

- Use o comando delete-acl para excluir uma ACL.

  Para Linux, macOS ou Unix:

  ```
  aws memorydb delete-acl /
     --acl-name
  ```

Para Windows:

```
aws memorydb delete-acl ^
   --acl-name
```

Os exemplos anteriores retornam a seguinte resposta.

```
aws memorydb delete-acl --acl-name "new-acl-1"
{
    "ACLName": "new-acl-1",
    "Status": "deleting",
    "EngineVersion": "6.2",
    "UserNames": [
        "user-name-1",
        "user-name-3"
    ],
    "clusters": [],
    "ARN":"arn:aws:memorydb:us-east-1:493071037918:acl/new-acl-1"
}
```

Atribuição de listas de controle de acesso a clusters

Depois de criar uma ACL e adicionar usuários, a etapa final na implementação de ACLs é atribuir a ACL a um cluster.

Atribuindo listas de controle de acesso a clusters usando o console

Para adicionar uma ACL a um cluster usando o AWS Management Console, consulteCriação de um cluster do MemoryDB.

Atribuindo listas de controle de acesso a clusters usando o AWS CLI

A AWS CLI operação a seguir cria um cluster com a criptografia em trânsito (TLS) ativada e o acl-name parâmetro com o valor*my-acl-name*. Substitua o grupo de sub-rede `subnet-group` por um grupo de sub-rede que exista.

Principais parâmetros

- **`--engine-version`** – deve ser 6.2.
- **`--tls-enabled`** – usado para autenticação e para associar uma ACL.

- **`--acl-name`** – esse valor fornece listas de controle de acesso compostas por usuários com permissões de acesso especificadas para o cluster.

Para Linux, macOS ou Unix:

```
aws memorydb create-cluster \
    --cluster-name "new-cluster" \
    --description "new-cluster" \
    --engine-version "6.2" \
    --node-type db.r6g.large \
    --tls-enabled \
    --acl-name "new-acl-1" \
    --subnet-group-name "subnet-group"
```

Para Windows:

```
aws memorydb create-cluster ^
    --cluster-name "new-cluster" ^
    --cluster-description "new-cluster" ^
    --engine-version "6.2" ^
    --node-type db.r6g.large ^
    --tls-enabled ^
    --acl-name "new-acl-1" ^
    --subnet-group-name "subnet-group"
```

A AWS CLI operação a seguir modifica um cluster com a criptografia em trânsito (TLS) ativada e o acl-name parâmetro com o valor. `new-acl-2`

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name cluster-1 \
    --acl-name "new-acl-2"
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name cluster-1 ^
    --acl-name "new-acl-2"
```

Autenticação com o IAM

Tópicos

Visão geral

Com a autenticação IAM, você pode autenticar uma conexão com o MemoryDB usando identidades IAM da AWS, quando seu cluster estiver configurado para usar a versão 7 ou superior do Redis. Isso possibilita que você fortaleça seu modelo de segurança e simplifique várias tarefas administrativas de segurança. Com a autenticação IAM, é possível configurar o controle de acesso refinado para cada cluster e usuário individual do MemoryDB e seguir os princípios de permissões de privilégio mínimo. A autenticação IAM para MemoryDB Redis fornece um token de autenticação IAM de curta duração em vez de uma senha de usuário MemoryDB de longa duração no comando `AUTH` ou `HELLO` do Redis. Para obter mais informações sobre o token de autenticação do IAM, consulte o processo de assinatura do Signature versão 4 no Guia de referência geral do AWS e no exemplo de código abaixo.

Você pode usar as identidades do IAM e suas políticas associadas para restringir ainda mais o acesso ao Redis. Você também pode conceder acesso aos usuários de seus provedores de identidade federados diretamente aos clusters do MemoryDB.

Para usar o IAM com o MemoryDB da AWS, primeiro é necessário criar um usuário do MemoryDB com o modo de autenticação definido como IAM e, em seguida, criar ou reutilizar uma identidade do IAM. A identidade IAM precisa de uma política associada para conceder a ação `memorydb:Connect` ao cluster do MemoryDB e ao usuário do MemoryDB. Depois de configurado, você pode criar um token de autenticação do IAM usando as credenciais da AWS do usuário ou do perfil do IAM. Por fim, você precisa fornecer o token de autenticação IAM de curta duração como uma senha no seu cliente Redis ao se conectar ao nó do cluster do MemoryDB. Um cliente Redis com suporte para provedor de credenciais pode gerar automaticamente as credenciais temporárias para cada nova conexão. O MemoryDB executará a autenticação do IAM para solicitações de conexão de usuários do MemoryDB habilitados para o IAM e validará as solicitações de conexão com o IAM.

Limitações

Ao usar a autenticação do IAM, as seguintes limitações se aplicam:

- A autenticação IAM está disponível ao usar o mecanismo Redis versão 7.0 ou superior.
- O token de autenticação do IAM é válido por 15 minutos. Para conexões de longa duração, recomendamos usar um cliente Redis que ofereça suporte a uma interface de provedor de credenciais.
- Uma conexão autenticada pelo IAM com o MemoryDB será automaticamente desconectada após 12 horas. A conexão pode ser prolongada por 12 horas enviando um comando `AUTH` ou `HELLO` com um novo token de autenticação do IAM.
- A autenticação do IAM não tem suporte em comandos `MULTI EXEC`.
- Atualmente, a autenticação do IAM não oferece suporte a todas as chaves de contexto de condição global. Para obter mais informações sobre chaves de contexto de condição global, consulte Chaves de contexto de condição global da AWS no Guia do usuário do IAM.

Configuração

Como configurar a autenticação do IAM

1. Criar um cluster do

```
aws memorydb create-cluster \
    --cluster-name cluster-01 \
    --description "MemoryDB IAM auth application"
    --node-type db.r6g.large \
    --engine-version 7.0 \
    --acl-name open-access
```

2. Crie um documento de política de confiança do IAM, conforme mostrado abaixo, para o perfil que permita que sua conta assuma o novo perfil. Salve a política em um arquivo chamado trust-policy.json.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
        "Action": "sts:AssumeRole"
```

```
    }
}
```

3. Crie um documento de política do IAM, conforme mostrado abaixo. Salve a política em um arquivo chamado policy.json.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
      "Action" : [
        "memorydb:connect"
      ],
      "Resource" : [
        "arn:aws:memorydb:us-east-1:123456789012:cluster/cluster-01",
        "arn:aws:memorydb:us-east-1:123456789012:user/iam-user-01"
      ]
    }
  ]
}
```

4. Crie um perfil do IAM.

```
aws iam create-role \
  --role-name "memorydb-iam-auth-app" \
  --assume-role-policy-document file://trust-policy.json
```

5. Crie a política do IAM.

```
aws iam create-policy \
  --policy-name "memorydb-allow-all" \
  --policy-document file://policy.json
```

6. Anexe a política do IAM à função.

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

7. Crie um novo usuário habilitado para o IAM.

```
aws memorydb create-user \
```

```
  --user-name iam-user-01 \
  --authentication-mode Type=iam \
  --access-string "on ~* +@all"
```

8. Crie uma ACL e inclua o usuário.

```
aws memorydb create-acl \
  --acl-name iam-acl-01 \
  --user-names iam-user-01

aws memorydb update-cluster \
  --cluster-name cluster-01 \
  --acl-name iam-acl-01
```

Conexão

Conectar com token como senha

Primeiro, é necessário gerar o token de autenticação do IAM de curta duração usando uma solicitação pré-assinada do AWS SigV4. Depois disso, forneça o token de autenticação do IAM como senha ao se conectar a um cluster do MemoryDB, conforme mostrado no exemplo abaixo.

```
String userName = "insert user name"
String clusterName = "insert cluster name"
String region = "insert region"

// Create a default AWS Credentials provider.
// This will look for AWS credentials defined in environment variables or system
 properties.
AWSCredentialsProvider awsCredentialsProvider = new
 DefaultAWSCredentialsProviderChain();

// Create an IAM authentication token request and signed it using the AWS credentials.
// The pre-signed request URL is used as an IAM authentication token for MemoryDB
 Redis.
IAMAuthTokenRequest iamAuthTokenRequest = new IAMAuthTokenRequest(userName,
 clusterName, region);
String iamAuthToken =
 iamAuthTokenRequest.toSignedRequestUri(awsCredentialsProvider.getCredentials());

// Construct Redis URL with IAM Auth credentials provider
RedisURI redisURI = RedisURI.builder()
```

```
    .withHost(host)
    .withPort(port)
    .withSsl(ssl)
    .withAuthentication(userName, iamAuthToken)
    .build();

// Create a new Lettuce Redis client
RedisClusterClient client = RedisClusterClient.create(redisURI);
client.connect();
```

Veja abaixo a definição de `IAMAuthTokenRequest`.

```
public class IAMAuthTokenRequest {
    private static final HttpMethodName REQUEST_METHOD = HttpMethodName.GET;
    private static final String REQUEST_PROTOCOL = "http://";
    private static final String PARAM_ACTION = "Action";
    private static final String PARAM_USER = "User";
    private static final String ACTION_NAME = "connect";
    private static final String SERVICE_NAME = "memorydb";
    private static final long TOKEN_EXPIRY_SECONDS = 900;

    private final String userName;
    private final String clusterName;
    private final String region;

    public IAMAuthTokenRequest(String userName, String clusterName, String region) {
        this.userName = userName;
        this.clusterName = clusterName;
        this.region = region;
    }

    public String toSignedRequestUri(AWSCredentials credentials) throws
 URISyntaxException {
        Request<Void> request = getSignableRequest();
        sign(request, credentials);
        return new URIBuilder(request.getEndpoint())
            .addParameters(toNamedValuePair(request.getParameters()))
            .build()
            .toString()
            .replace(REQUEST_PROTOCOL, "");
    }

    private <T> Request<T> getSignableRequest() {
```

```
        Request<T> request  = new DefaultRequest<>(SERVICE_NAME);
        request.setHttpMethod(REQUEST_METHOD);
        request.setEndpoint(getRequestUri());
        request.addParameters(PARAM_ACTION, Collections.singletonList(ACTION_NAME));
        request.addParameters(PARAM_USER, Collections.singletonList(userName));
        return request;
    }

    private URI getRequestUri() {
        return URI.create(String.format("%s%s/", REQUEST_PROTOCOL, clusterName));
    }

    private <T> void sign(SignableRequest<T> request, AWSCredentials credentials) {
        AWS4Signer signer = new AWS4Signer();
        signer.setRegionName(region);
        signer.setServiceName(SERVICE_NAME);

        DateTime dateTime = DateTime.now();
        dateTime = dateTime.plus(Duration.standardSeconds(TOKEN_EXPIRY_SECONDS));

        signer.presignRequest(request, credentials, dateTime.toDate());
    }

    private static List<NameValuePair> toNamedValuePair(Map<String, List<String>> in) {
        return in.entrySet().stream()
            .map(e -> new BasicNameValuePair(e.getKey(), e.getValue().get(0)))
            .collect(Collectors.toList());
    }
}
```

Conectar com o provedor de credenciais

O código abaixo mostra como se autenticar no MemoryDB usando o provedor de credenciais de autenticação IAM.

```
String userName = "insert user name"
String clusterName = "insert cluster name"
String region = "insert region"

// Create a default AWS Credentials provider.
// This will look for AWS credentials defined in environment variables or system
 properties.
```

```
AWSCredentialsProvider awsCredentialsProvider = new
 DefaultAWSCredentialsProviderChain();

// Create an IAM authentication token request. Once this request is signed it can be
 used as an
// IAM authentication token for MemoryDB Redis.
IAMAuthTokenRequest iamAuthTokenRequest = new IAMAuthTokenRequest(userName,
 clusterName, region);

// Create a Redis credentials provider using IAM credentials.
RedisCredentialsProvider redisCredentialsProvider = new
 RedisIAMAuthCredentialsProvider(
    userName, iamAuthTokenRequest, awsCredentialsProvider);

// Construct Redis URL with IAM Auth credentials provider
RedisURI redisURI = RedisURI.builder()
    .withHost(host)
    .withPort(port)
    .withSsl(ssl)
    .withAuthentication(redisCredentialsProvider)
    .build();

// Create a new Lettuce Redis cluster client
RedisClusterClient client = RedisClusterClient.create(redisURI);
client.connect();
```

Abaixo está um exemplo de um cliente de cluster do Lettuce Redis que envolve o IAMAuthTokenRequest em um provedor de credenciais para gerar automaticamente credenciais temporárias quando necessário.

```
public class RedisIAMAuthCredentialsProvider implements RedisCredentialsProvider {
    private static final long TOKEN_EXPIRY_SECONDS = 900;

    private final AWSCredentialsProvider awsCredentialsProvider;
    private final String userName;
    private final IAMAuthTokenRequest iamAuthTokenRequest;
    private final Supplier<String> iamAuthTokenSupplier;

    public RedisIAMAuthCredentialsProvider(String userName,
        IAMAuthTokenRequest iamAuthTokenRequest,
        AWSCredentialsProvider awsCredentialsProvider) {
        this.userName = userName;
        this.awsCredentialsProvider = awsCredentialsProvider;
```

```
        this.iamAuthTokenRequest = iamAuthTokenRequest;
        this.iamAuthTokenSupplier =
Suppliers.memoizeWithExpiration(this::getIamAuthToken, TOKEN_EXPIRY_SECONDS,
TimeUnit.SECONDS);
    }

    @Override
    public Mono<RedisCredentials> resolveCredentials() {
        return Mono.just(RedisCredentials.just(userName, iamAuthTokenSupplier.get()));
    }

    private String getIamAuthToken() {
        return
iamAuthTokenRequest.toSignedRequestUri(awsCredentialsProvider.getCredentials());
    }
```

Gerenciamento de identidade e acesso no MemoryDB para Redis

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 (fazer login) e autorizado (tem permissões) a usar os recursos do MemoryDB. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

Público

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz no MemoryDB.

Usuário do serviço: se você usa o serviço MemoryDB para fazer seu trabalho, o administrador fornece as credenciais e as permissões necessárias. À medida que você usar mais atributos do MemoryDB para fazer seu trabalho, poderá precisar de permissões adicionais. Entender como o acesso é gerenciado pode ajudar você a solicitar as permissões corretas ao seu administrador. Se não for possível acessar um atributo no MemoryDB, consulte Solução de problemas de identidade e acesso do MemoryDB para Redis.

Administrador do serviço: se você for o responsável pelos recursos do MemoryDB em sua empresa, provavelmente terá acesso total ao MemoryDB. Cabe a você determinar quais funcionalidades e atributos do MemoryDB os usuários do serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais sobre como a empresa pode usar o IAM com o MemoryDB, consulte Como o MemoryDB para Redis funciona com o IAM.

Administrador do IAM: se você for um administrador de IAM, talvez queira saber detalhes sobre como é possível criar políticas para gerenciar o acesso ao MemoryDB. Para visualizar exemplos de políticas baseadas em identidade do MemoryDB que podem ser usadas no IAM, consulte Exemplos de políticas baseadas em identidade do MemoryDB para Redis.

Autenticando com identidades

A autenticação é como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação Multifator no Guia do Usuário do AWS IAM Identity Center . Usar a autenticação multifator (MFA) na AWS no Guia do Usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do Usuário do IAM.

Identidade federada

Como prática recomendada, exija que usuários humanos, incluindo usuários que precisam de acesso de administrador, usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas acessam Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o AWS IAM Identity Center. Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas da AWS aplicativos. Para obter mais informações sobre o Centro de Identidade do IAM, consulte O que é o Centro de Identidade do IAM? no Manual do Usuário do AWS IAM Identity Center .

Usuários e grupos do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do Usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de um perfil) no Guia do usuário do IAM.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para o uso de perfis, consulte Utilizar perfis do IAM no Guia do usuário do IAM.

Funções do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do Usuário do IAM. Se você usar o Centro de identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de Permissões no Manual do Usuário do AWS IAM Identity Center .
- Permissões temporárias para usuários do IAM –- um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas –- é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado a serviço.
  - Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
  - Função de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

- Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para funções vinculadas ao serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Utilizar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar perfis do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) no Guia do usuário do IAM.

Gerenciando acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do Usuário do IAM.

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissões para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem presumir os perfis.

As políticas do IAM definem permissões para uma ação independente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação

`iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criando políticas do IAM no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda adicionalmente como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do Usuário do IAM.

Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um atributo avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade com seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do Usuário do IAM.
- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em. AWS Organizations AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para obter mais informações sobre o Organizações e SCPs, consulte Como os SCPs Funcionam no Manual do Usuário do AWS Organizations .
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em atributo. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Guia do Usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como o MemoryDB para Redis funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao MemoryDB, saiba quais atributos do IAM estão disponíveis para uso com o MemoryDB.

Atributos do IAM que você pode usar com o MemoryDB para Redis

Atributo do IAM	Suporte do MemoryDB
Políticas baseadas em identidade	Sim
Políticas baseadas em recursos	Não
Ações das políticas	Sim
Atributos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Sim
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Permissões de entidade principal	Sim
Perfis de serviço	Sim
Funções vinculadas a serviço	Sim

Para ter uma visão de alto nível de como o MemoryDB e outros AWS serviços funcionam com a maioria dos recursos do IAM, consulte AWS os serviços que funcionam com o IAM no Guia do usuário do IAM.

Políticas do MemoryDB baseadas em identidade

Suporta políticas baseadas em identidade	Sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criando políticas do IAM no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou perfil ao qual ela está anexada. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elementos da política JSON do IAM no Guia do Usuário do IAM.

Exemplos de políticas baseadas em identidade para o MemoryDB

Para visualizar exemplos de políticas baseadas em identidade do MemoryDB, consulte Exemplos de políticas baseadas em identidade do MemoryDB para Redis.

Políticas baseadas em recursos no MemoryDB

Oferece compatibilidade com políticas baseadas em recursos	Não

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado

pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em atributo. Adicionar uma entidade principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando o principal e o recurso são diferentes Contas da AWS, um administrador do IAM na conta confiável também deve conceder permissão à entidade principal (usuário ou função) para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em recurso conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Para obter mais informações, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Ações de políticas do MemoryDB

Oferece compatibilidade com ações de políticas	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista das ações do MemoryDB, consulte Ações definidas pelo MemoryDB para Redis na Referência de autorização do serviço.

As ações de políticas no MemoryDB usam o seguinte prefixo antes da ação:

```
MemoryDB
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

```
"Action": [
      "MemoryDB:action1",
      "MemoryDB:action2"
         ]
```

Você também pode especificar várias ações usando caracteres-curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a seguinte ação:

```
"Action": "MemoryDB:Describe*"
```

Para visualizar exemplos de políticas baseadas em identidade do MemoryDB, consulte Exemplos de políticas baseadas em identidade do MemoryDB para Redis.

Recursos de políticas do MemoryDB

Oferece compatibilidade com recursos de políticas	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

Para obter uma lista dos tipos de recursos do MemoryDB e seus ARNs, consulte Recursos definidos pelo MemoryDB para Redis na Referência de autorização do serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo MemoryDB para Redis.

Para visualizar exemplos de políticas baseadas em identidade do MemoryDB, consulte Exemplos de políticas baseadas em identidade do MemoryDB para Redis.

Chaves de condição de políticas para o MemoryDB

Suporta chaves de condição de política específicas de serviço	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único `Condition` elemento, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um atributo somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos da política do IAM: variáveis e tags no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

Para visualizar exemplos de políticas baseadas em identidade do MemoryDB, consulte Exemplos de políticas baseadas em identidade do MemoryDB para Redis.

Uso de chaves de condição

Você pode especificar as condições que determinam como uma política do IAM entra em vigor. No MemoryDB, você pode usar o `Condition` elemento de uma política JSON para comparar chaves no

contexto da solicitação com valores de chave que você especifica em sua política. Para obter mais informações, consulte Elementos da política JSON do IAM: condição.

Para ver uma lista das chaves de condição do MemoryDB, consulte Chaves de condição do MemoryDB para Redis na Referência de Autorização de Serviço.

Para obter uma lista de todas as chaves de condição globais, consulte Chaves de contexto de condição global da AWS.

Especificação de condições: uso de chaves de condição

Para implementar um controle refinado, você pode escrever uma política de permissões do IAM que especifica as condições para controlar um conjunto de parâmetros individuais em determinadas solicitações. Em seguida, você pode aplicar a política aos usuários, grupos ou funções do IAM que você cria usando o console do IAM.

Para aplicar uma condição, adicione as informações da condição à declaração de política do IAM. Por exemplo, para proibir a criação de qualquer cluster MemoryDB com o TLS desativado, você pode especificar a seguinte condição em sua declaração de política.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "memorydb:CreateCluster"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Bool": {
          "memorydb:TLSEnabled": "false"
        }
      }
    }
  ]
}
```

Para obter mais informações sobre marcação, consulteUso de tags em seus recursos do MemoryDB.

Para obter mais informações sobre a utilização de operadores de condição de política, consulte Permissões da API do MemoryDB: referência de condições, recursos e ações.

Políticas de exemplo: uso de condições para controle de acesso refinado

Esta seção mostra exemplos de políticas para implementar um controle de acesso refinado nos parâmetros do MemoryDB listados anteriormente.

1. MemoryDB:tlsEnabled — Especifique que os clusters serão criados somente com o TLS ativado.

```
{
    "Version": "2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
            "Action": [
                "memorydb:CreateCluster"
            ],
            "Resource": [
                "arn:aws:memorydb:*:*:parametergroup/*",
                "arn:aws:memorydb:*:*:subnetgroup/*",
                "arn:aws:memorydb:*:*:acl/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "memorydb:CreateCluster"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "Bool": {
                    "memorydb:TLSEnabled": "true"
                }
            }
        }
    ]
}
```

2. memorydb:UserAuthenticationMode: — Especifique que os usuários possam ser criados com um modo de autenticação de tipo específico (IAM, por exemplo).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "memorydb:Createuser"
            ],
            "Resource": [
                "arn:aws:memorydb:*:*:user/*"
            ],
            "Condition": {
                "StringEquals": {
                    "memorydb:UserAuthenticationMode": "iam"
                }
            }
        }
    ]
}
```

Nos casos em que você está definindo políticas baseadas em 'Negar', é recomendável usar a StringEqualsIgnoreCaseoperadora para evitar todas as chamadas com um tipo específico de modo de autenticação de usuário, independentemente do caso.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "memorydb:CreateUser"
      ],
      "Resource": "*",
      "Condition": {
        "StringEqualsIgnoreCase": {
          "memorydb:UserAuthenticationMode": "password"
        }
      }
    }
  ]
}
```

Listas de controle de acesso (ACLs) no MemoryDB

Oferece compatibilidade com ACLs	Sim

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Controle de acesso por atributo (ABAC) com o MemoryDB

Oferece compatibilidade com ABAC (tags em políticas)	Sim

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. Em AWS, esses atributos são chamados de tags. Você pode anexar tags a entidades do IAM (usuários ou funções) e a vários AWS recursos. A marcação de entidades e atributos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela estiver tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações onde o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou chaves de condição `aws:TagKeys`.

Se um serviço oferecer suporte às três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para obter mais informações sobre o ABAC, consulte O que é ABAC? no Guia do Usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar controle de acesso baseado em atributos (ABAC) no Guia do usuário do IAM.

Usar credenciais temporárias com o MemoryDB

Oferece compatibilidade com credenciais temporárias	Sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS "Trabalhe com o IAM" no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login AWS Management Console usando qualquer método, exceto um nome de usuário e senha. Por exemplo, quando você acessa AWS usando o link de login único (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para obter mais informações sobre como alternar funções, consulte Alternar para um perfil (console) no Guia do usuário do IAM.

Você pode criar manualmente credenciais temporárias usando a AWS API AWS CLI ou. Em seguida, você pode usar essas credenciais temporárias para acessar AWS. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para mais informações, consulte Credenciais de segurança temporárias no IAM.

Permissões de entidade principal entre serviços para o MemoryDB

Suporte para o recurso Encaminhamento de sessões de acesso (FAS)	Sim

Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.

Perfis de serviço para o MemoryDB

Oferece compatibilidade com funções de serviço	Sim

O perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

> **Warning**
>
> Alterar as permissões de um perfil de serviço pode prejudicar a funcionalidade do MemoryDB. Edite os perfis de serviço somente quando o MemoryDB orientar você a fazê-lo.

Perfis vinculados ao serviço para o MemoryDB

Oferece suporte a perfis vinculados ao serviço	Sim

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

Para obter detalhes sobre como criar ou gerenciar perfis vinculados a serviços, consulte Serviços da AWS que funcionam com o IAM. Encontre um serviço na tabela que inclua um `Yes` na coluna Função vinculada ao serviço. Escolha o link Sim para visualizar a documentação do perfil vinculado a serviço desse serviço.

Exemplos de políticas baseadas em identidade do MemoryDB para Redis

Por padrão, usuários e funções não têm permissão para criar ou modificar recursos do MemoryDB. Eles também não podem realizar tarefas usando a AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) ou. Para conceder aos usuários permissão para executar ações

nos recursos de que precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem presumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte Criação de políticas do IAM no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recurso definidos pelo MemoryDB, incluindo o formato dos ARNs para cada tipo de recurso, consulte Ações, recursos e chaves de condição do MemoryDB para Redis na Referência de autorização do serviço.

Tópicos

- Melhores práticas de política
- Como usar o console do MemoryDB
- Permitir que usuários visualizem suas próprias permissões

Melhores práticas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do MemoryDB em sua conta. Essas ações podem incorrer em custos para seus Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e passe para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do Usuário do IAM.
- Aplique permissões de privilégio mínimo –- ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em atributos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do Usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso –- você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode gravar uma condição de política para especificar que todas as solicitações devem ser enviadas

usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: Condição no Guia do usuário do IAM.

- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais –- o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudá-lo a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas Recomendadas de Segurança no IAM no Guia do Usuário do IAM.

Como usar o console do MemoryDB

Para acessar o console do MemoryDB para Redis, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos do MemoryDB em seu. Conta da AWS Se você criar uma política baseada em identidade que seja mais restritiva do que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Você não precisa permitir permissões mínimas do console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a AWS API. Em vez disso, permita o acesso somente a ações que correspondam a operação de API que estiverem tentando executar.

Para garantir que usuários e funções ainda possam usar o console do MemoryDB, anexe também o MemoryDB `ConsoleAccess` ou a política `ReadOnly` AWS gerenciada às entidades. Para obter mais informações, consulte Adicionando Permissões a um Usuário no Guia do Usuário do IAM.

Permitir que usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Solução de problemas de identidade e acesso do MemoryDB para Redis

Use as seguintes informações para ajudar a diagnosticar e corrigir problemas comuns que podem ser encontrados ao trabalhar com o MemoryDB e o IAM.

Tópicos

Não tenho autorização para executar uma ação no MemoryDB

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador para obter ajuda. O administrador é a pessoa que forneceu o seu nome de usuário e senha.

O erro do exemplo a seguir ocorre quando o usuário `mateojackson` tenta usar o console para visualizar detalhes sobre um recurso do *`my-example-widget`* fictício, mas não tem as permissões fictícias do `MemoryDB:`*`GetWidget`*.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 MemoryDB:GetWidget on resource: my-example-widget
```

Neste caso, Mateo pede ao administrador para atualizar suas políticas e permitir o acesso ao recurso *`my-example-widget`* usando a ação `MemoryDB:`*`GetWidget`*.

Não estou autorizado a realizar iam: PassRole

Se você receber uma mensagem de erro informando que não tem autorização para executar a ação `iam:PassRole`, as suas políticas deverão ser atualizadas para permitir a passagem de um perfil para o ACM.

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 exemplo de erro a seguir ocorre quando uma usuário do IAM chamado `marymajor` tenta usar o console para executar uma ação no MemoryDB. 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.

Quero permitir que pessoas fora da minha AWS conta acessem meus recursos do MemoryDB

Você pode criar um perfil que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. Você pode especificar quem é confiável para assumir o perfil. Para serviços que oferecem compatibilidade com políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o MemoryDB oferece suporte a esses atributos, consulte Como o MemoryDB para Redis funciona com o IAM.
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre usar funções e políticas baseadas em recursos para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Controle de acesso

Você pode ter credenciais válidas para autenticar suas solicitações, mas a menos que tenha permissões, você não pode criar ou acessar recursos do MemoryDB para Redis. Por exemplo, você deve ter permissões para criar um cluster do MemoryDB.

As seções a seguir descrevem como gerenciar permissões para o MemoryDB para Redis. Recomendamos que você leia a visão geral primeiro.

- Visão geral do gerenciamento de permissões de acesso aos recursos do MemoryDB
- Usar políticas baseadas em identidade (políticas do IAM) para o MemoryDB para Redis

Visão geral do gerenciamento de permissões de acesso aos recursos do MemoryDB

Cada AWS recurso pertence a uma AWS conta, e as permissões para criar ou acessar um recurso são regidas por políticas de permissões. Um administrador de conta pode anexar políticas de permissões a identidades do IAM (ou seja, usuários, grupos e funções). Além disso, o MemoryDB para Redis também oferece suporte à anexação de políticas de permissões aos recursos.

> **Note**
>
> Um administrador da conta (ou usuário administrador) é um usuário com privilégios de administrador. Para obter mais informações, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em AWS IAM Identity Center:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Tópicos

- Usar políticas baseadas em identidade (políticas do IAM) para o MemoryDB para Redis
- Permissões em nível de recurso
- Uso de funções vinculadas ao serviço para o Amazon MemoryDB para Redis
- Políticas gerenciadas pela AWS para MemoryDB para Redis
- Permissões da API do MemoryDB: referência de condições, recursos e ações

Recursos e operações do MemoryDB para Redis

No MemoryDB para Redis, o recurso principal é um cluster.

Esses recursos têm nomes de recurso da Amazon (ARNs) exclusivos associados, conforme mostrado a seguir.

> **Note**
>
> Para que as permissões de nível de recurso sejam efetivas, o nome do recurso na string ARN deve ser minúsculo.

Tipo de recurso	Formato ARN
Usuário	arn:aws:memorydb:*us-east-1:123456789012* :user/user1
Lista de controle de acesso (ACL)	arn:aws:memorydb:*us-east-1:123456789012* :acl/myacl
Cluster	arn:aws:memorydb:*us-east-1:123456789012* :cluster/my-cluster
Snapshot	arn:aws:memorydb:*us-east-1:123456789012* :snapshot/my-snapshot
Grupo de parâmetros	*arn: aws: memorydb: us-east-1:123456789012: grupo de parâmetros/* my-parameter-group

Tipo de recurso	Formato ARN
Grupo de sub-redes	*arn: aws: memorydb: us-east-1:123456789012: subnetgroup/* my-subnet-group

O MemoryDB fornece um conjunto de operações para trabalhar com recursos do MemoryDB. Para obter uma lista das operações disponíveis, consulte MemoryDB para Redis Ações.

Informações sobre propriedade de recursos

O proprietário do recurso é a AWS conta que criou o recurso. Ou seja, o proprietário do recurso é a AWS conta da entidade principal que autentica a solicitação que cria o recurso. Uma entidade principal pode ser a conta raiz, um usuário do IAM ou uma perfil do IAM. Os seguintes exemplos mostram como isso funciona:

- Suponha que você use as credenciais da conta raiz da sua AWS conta para criar um cluster. Nesse caso, sua AWS conta é a proprietária do recurso. No MemoryDB, o recurso é o cluster.
- Suponha que você crie um usuário do IAM em sua AWS conta e conceda permissões para criar um cluster para esse usuário. Nesse caso, o usuário pode criar um cluster. No entanto, sua AWS conta, à qual o usuário pertence, é proprietária do recurso de cluster.
- Suponha que você crie uma função do IAM em sua AWS conta com permissões para criar um cluster. Nesse caso, qualquer pessoa que possa assumir a função poderá criar um cluster. Sua AWS conta, à qual a função pertence, é proprietária do recurso de cluster.

Gerenciamento de acesso aos recursos

A política de permissões descreve quem tem acesso a quê. A seção a seguir explica as opções disponíveis para a criação das políticas de permissões.

> **Note**
>
> Esta seção discute o uso do IAM no contexto do MemoryDB para Redis. Não são fornecidas informações detalhadas sobre o serviço IAM. Para obter a documentação completa do IAM, consulte O que é o IAM? no Guia do usuário do IAM. Para obter mais informações sobre a

sintaxe e as descrições da política do IAM, consulte a Referência de políticas do AWS IAM no Guia do usuário do IAM.

As políticas anexadas a uma identidade do IAM são conhecidas como políticas baseadas em identidade (políticas do IAM). As políticas anexadas a um recurso são chamadas de políticas baseadas em recursos.

Tópicos

- Políticas baseadas em identidade (políticas do IAM)
- Especificar elementos da política: ações, efeitos, recursos e entidades principais
- Especificar condições em uma política

Políticas baseadas em identidade (políticas do IAM)

Você pode anexar políticas a identidades do IAM. Por exemplo, você pode fazer o seguinte:

- Anexar uma política de permissões a um usuário ou grupo na sua conta: um administrador de conta pode usar uma política de permissões associada a determinado usuário para conceder permissões. Nesse caso, as permissões são para o usuário criar um recurso do MemoryDB, como um cluster, um grupo de parâmetros ou um grupo de segurança.
- Anexar uma política de permissões a uma função: você pode anexar uma política de permissões baseada em identidade a um perfil do IAM para conceder permissões entre contas. Por exemplo, o administrador na Conta A pode criar uma função para conceder permissões entre contas a outra AWS conta (por exemplo, Conta B) ou a um AWS serviço da seguinte forma:
  1. Um administrador da Conta A cria uma função do IAM e anexa uma política de permissões à função que concede permissões em recursos da Conta A.
  2. Um administrador da Conta A anexa uma política de confiança à função identificando a Conta B como a entidade principal, que pode assumir a função.
  3. O administrador da Conta B pode então delegar permissões para assumir a função a qualquer usuário na Conta B. Isso permite que os usuários da Conta B criem ou acessem recursos na Conta A. Em alguns casos, talvez você queira conceder permissões a um AWS serviço para assumir a função. Para oferecer suporte a essa abordagem, o principal da política de confiança também pode ser um principal de serviço da AWS .

Para obter mais informações sobre o uso do IAM para delegar permissões, consulte Gerenciamento de acesso no Guia do usuário do IAM.

Veja a seguir um exemplo de política que permite que um usuário execute a `DescribeClusters` ação AWS em sua conta. O MemoryDB também permite identificar recursos específicos usando os ARNs de recurso para as ações da API. Essa abordagem também é chamada de permissões no nível do recurso.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "DescribeClusters",
      "Effect": "Allow",
      "Action": [
         "memorydb:DescribeClusters"],
      "Resource": resource-arn
      }
   ]
}
```

Para obter mais informações sobre como usar políticas baseadas em identidade com o MemoryDB, consulte Usar políticas baseadas em identidade (políticas do IAM) para o MemoryDB para Redis. Para obter mais informações sobre usuários, grupos, funções e permissões, consulte Identidades (usuários, grupos e funções) no Guia do usuário do IAM.

Especificar elementos da política: ações, efeitos, recursos e entidades principais

Para cada recurso do MemoryDB para Redis (consulte Recursos e operações do MemoryDB para Redis), o serviço define um conjunto de operações de API (consulte Ações). Para conceder permissões a essas operações da API, o MemoryDB define um conjunto de ações que podem ser especificadas em uma política. Por exemplo, para o recurso de cluster do MemoryDB, as seguintes ações são definidas: `CreateCluster`, `DeleteCluster`, e `DescribeClusters`. A execução de uma operação de API pode exigir permissões para mais de uma ação.

Estes são os elementos de política mais básicos:

- Recurso: em uma política, você usa um Amazon Resource Name (ARN – Nome do recurso da Amazon) para identificar o recurso a que a política se aplica. Para ter mais informações, consulte Recursos e operações do MemoryDB para Redis.

- Ação: você usa palavras-chave de ação para identificar operações de recursos que deseja permitir ou negar. Por exemplo, dependendo do `Effect` especificado, a permissão `memorydb:CreateCluster` permite ou nega as permissões do usuário para executar a operação `CreateCluster` no MemoryDB para Redis.
- Efeito: você especifica o efeito quando o usuário solicita a ação específica, que pode ser permitir ou negar. Se você não conceder (permitir) explicitamente acesso a um recurso, o acesso estará implicitamente negado. Você também pode negar acesso explicitamente a um recurso. Por exemplo, você poderia fazer isso para garantir que um usuário não possa acessar o recurso, mesmo se uma política diferente conceder o acesso.
- Entidade principal: em políticas baseadas em identidade (políticas do IAM), o usuário ao qual a política é anexada é a entidade principal implícita. Para as políticas baseadas em recursos, você especifica quais usuários, contas, serviços ou outras entidades deseja que recebam permissões (isso se aplica somente a políticas baseadas em recursos).

Para saber mais sobre a sintaxe e as descrições de políticas do IAM, consulte a Referência de políticas do AWS IAM da no Guia do usuário do IAM.

Para obter uma tabela que mostra todas as ações de API do MemoryDB, consulte Permissões da API do MemoryDB: referência de condições, recursos e ações.

Especificar condições em uma política

Ao conceder permissões, você pode usar a linguagem da política do IAM para especificar as condições de quando uma política deverá entrar em vigor. Por exemplo, é recomendável aplicar uma política somente após uma data específica. Para obter mais informações sobre como especificar condições em uma linguagem de política, consulte Condition no Guia do usuário do IAM.

Usar políticas baseadas em identidade (políticas do IAM) para o MemoryDB para Redis

Este tópico fornece exemplos de políticas baseadas em identidade em que um administrador de conta pode anexar políticas de permissões a identidades do IAM (ou seja, usuários, grupos e funções).

> **Important**
>
> Recomendamos que você primeiro leia os tópicos que explicam os conceitos básicos e as opções para gerenciar o acesso aos recursos do MemoryDB para Redis. Para ter mais informações, consulte Visão geral do gerenciamento de permissões de acesso aos recursos do MemoryDB.

As seções neste tópico abrangem o seguinte:

- Permissões necessárias para usar o console do MemoryDB para Redis
- Políticas (predefinidas) gerenciadas pela AWS do MemoryDB para Redis
- Exemplos de política gerenciada pelo cliente

A seguir, um exemplo de uma política de permissões.

```
{
   "Version": "2012-10-17",
   "Statement": [{
       "Sid": "AllowClusterPermissions",
       "Effect": "Allow",
       "Action": [
          "memorydb:CreateCluster",
          "memorydb:DescribeClusters",
          "memorydb:UpdateCluster"],
       "Resource": "*"
       },
       {
         "Sid": "AllowUserToPassRole",
         "Effect": "Allow",
         "Action": [ "iam:PassRole" ],
         "Resource": "arn:aws:iam::123456789012:role/EC2-roles-for-cluster"
```

```
        }
    ]
}
```

A política tem duas instruções:

- A primeira instrução concede permissões para as ações do MemoryDB para Redis (`memorydb:CreateCluster`, `memorydb:DescribeClusters`, e `memorydb:UpdateCluster`) em qualquer cluster pertencente à conta.
- A segunda instrução concede permissões para a ação do IAM (`iam:PassRole`) no nome da função do IAM especificado no final do valor `Resource`.

A política não especifica o elemento `Principal` porque, em uma política baseada em identidade, a entidade principal que obtém as permissões não é especificada. Quando você anexar uma política um usuário, o usuário será a entidade principal implícita. Quando você anexa uma política de permissões a um perfil do IAM, o principal identificado na política de confiança do perfil obtém as permissões.

Para ver uma tabela que mostra todas as ações da API do MemoryDB para Redis e os recursos aos quais elas se aplicam, consulte Permissões da API do MemoryDB: referência de condições, recursos e ações.

Permissões necessárias para usar o console do MemoryDB para Redis

A tabela de referência de permissões lista as operações de API do MemoryDB para Redis e mostra as permissões necessárias para cada operação. Para obter mais informações sobre as operações de API do MemoryDB, consulte Permissões da API do MemoryDB: referência de condições, recursos e ações.

Para usar o console do MemoryDB para Redis, primeiro conceda permissões para ações adicionais, como mostrado na política de permissões a seguir.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "MinPermsForMemDBConsole",
        "Effect": "Allow",
        "Action": [
            "memorydb:Describe*",
```

```
            "memorydb:List*",
            "ec2:DescribeAvailabilityZones",
            "ec2:DescribeVpcs",
            "ec2:DescribeAccountAttributes",
            "ec2:DescribeSecurityGroups",
            "cloudwatch:GetMetricStatistics",
            "cloudwatch:DescribeAlarms",
            "s3:ListAllMyBuckets",
            "sns:ListTopics",
            "sns:ListSubscriptions" ],
        "Resource": "*"
        }
    ]
}
```

O console do MemoryDB precisa dessas permissões adicionais pelas seguintes razões:

- As permissões para ações do MemoryDB habilitam o console para exibir recursos do MemoryDB na conta.
- O console precisa de permissões para que as ações do `ec2` consultem o Amazon EC2, de forma que ele possa exibir zonas de disponibilidade, VPCs, grupos de segurança e os atributos da conta.
- As permissões para `cloudwatch` ações permitem que o console recupere CloudWatch métricas e alarmes da Amazon e os exiba no console.
- As permissões para ações do `sns` permitem que o console recupere tópicos e assinaturas do Amazon Simple Notification Service (Amazon SNS) e os exiba no console.

Exemplos de política gerenciada pelo cliente

Se você não estiver usando uma política padrão e optar por usar uma política gerenciada personalizada, realize uma destas ações: Você deve ter permissões para chamar `iam:createServiceLinkedRole` (para obter mais informações, consulte Exemplo 4: permitir que um usuário chame a CreateServiceLinkedRole API IAM). Ou você deve ter criado uma função vinculada ao serviço do MemoryDB.

Quando combinado com as permissões mínimas necessárias para usar o console do MemoryDB para Redis, as políticas de exemplo nesta seção concedem permissões adicionais. Os exemplos também são relevantes para os AWS SDKs e o. AWS CLI Para obter mais informações sobre quais permissões são necessárias para usar o console do MemoryDB, consulte Permissões necessárias para usar o console do MemoryDB para Redis.

Para obter instruções sobre como configurar usuários e grupos do IAM, consulte Criação do seu primeiro usuário do IAM e grupo de administradores no Guia do usuário do IAM.

> **Important**
>
> Sempre teste suas políticas do IAM completamente antes de usá-las em produção. Algumas ações do MemoryDB que parecem simples podem exigir outras ações para oferecer suporte quando você estiver usando o console do MemoryDB. Por exemplo, `memorydb:CreateCluster` concede permissões para criar clusters de cache do MemoryDB. No entanto, para realizar essa operação, o console do MemoryDB usa várias ações `Describe` e `List` para preencher listas de consoles.

Exemplos

- Exemplo 1: permitir a um usuário com acesso somente de leitura a recursos do MemoryDB
- Exemplo 2: permitir que um usuário realize tarefas comuns de administrador do sistema do MemoryDB
- Exemplo 3: permitir que um usuário acesse todas as ações de API do MemoryDB
- Exemplo 4: permitir que um usuário chame a CreateServiceLinkedRole API IAM

Exemplo 1: permitir a um usuário com acesso somente de leitura a recursos do MemoryDB

A seguinte política concede permissões a ações do MemoryDB que permitem que um usuário liste recursos. Normalmente, você anexa esse tipo de política de permissões a um grupo de gerentes.

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "MemDBUnrestricted",
      "Effect":"Allow",
      "Action": [
          "memorydb:Describe*",
          "memorydb:List*"],
      "Resource":"*"
      }
   ]
}
```

Exemplo 2: permitir que um usuário realize tarefas comuns de administrador do sistema do MemoryDB

As tarefas comuns do administrador do sistema incluem a modificação de clusters de cache, parâmetros e grupo de parâmetros. Um administrador do sistema também pode querer obter informações sobre eventos do MemoryDB. A seguinte política concede permissões de usuário para executar ações do MemoryDB para essas tarefas comuns de administrador de sistema. Normalmente, você anexa esse tipo de política de permissões ao grupo de administradores do sistema.

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "MDBAllowSpecific",
      "Effect":"Allow",
      "Action":[
          "memorydb:UpdateCluster",
          "memorydb:DescribeClusters",
          "memorydb:DescribeEvents",
          "memorydb:UpdateParameterGroup",
          "memorydb:DescribeParameterGroups",
          "memorydb:DescribeParameters",
          "memorydb:ResetParameterGroup",],
      "Resource":"*"
      }
   ]
}
```

Exemplo 3: permitir que um usuário acesse todas as ações de API do MemoryDB

A seguinte política permite que um usuário acesse todas as ações do MemoryDB. Recomendamos que você conceda esse tipo de política de permissões apenas a um usuário administrador.

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "MDBAllowAll",
      "Effect":"Allow",
      "Action":[
          "memorydb:*" ],
      "Resource":"*"
      }
```

```
    ]
}
```

Exemplo 4: permitir que um usuário chame a CreateServiceLinkedRole API IAM

A política a seguir permite que o usuário chame a API `CreateServiceLinkedRole` do IAM. Recomendamos conceder esse tipo de política de permissões para o usuário que invoca operações mutativas do MemoryDB.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"CreateSLRAllows",
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"*",
      "Condition":{
        "StringLike":{
          "iam:AWS ServiceName":"memorydb.amazonaws.com"
        }
      }
    }
  ]
}
```

Permissões em nível de recurso

Você pode restringir o escopo das permissões especificando recursos em uma política do IAM. Muitas ações de AWS CLI API oferecem suporte a um tipo de recurso que varia de acordo com o comportamento da ação. Cada instrução de política do IAM concede permissão a uma ação realizada em um recurso. Quando a ação não atua em um recurso indicado, ou quando você concede permissão para executar a ação em todos os recursos, o valor do recurso na política é um curinga (*). Para muitas ações de API, restrinja os recursos que um usuário pode modificar especificando o Amazon Resource Name (ARN – Nome de recurso da Amazon) de um recurso ou um padrão de ARN correspondente a vários recursos. Para restringir as permissões por recurso especifique o recurso por ARN.

Formato ARN do recurso MemoryDB

> **Note**
>
> Para que as permissões de nível de recurso sejam efetivas, o nome do recurso na string ARN deve ser minúsculo.

- Usuário: arn:aws:memorydb:*us-east-1:123456789012*:user/user1
- ACL: arn:aws:memorydb:*us-east-1:123456789012*:acl/my-acl
- Cluster: arn:aws:memorydb:*us-east-1:123456789012*:cluster/my-cluster
- Snapshot: arn:aws:memorydb:*us-east-1:123456789012*:snapshot/my-snapshot
- *Grupo de parâmetros – arn:aws:memorydb: us-east-1:123456789012:parametergroup/* my-parameter-group
- *Grupo de sub-redes – arn:aws:memorydb: us-east-1:123456789012:subnetgroup/* my-subnet-group

Exemplos

- Exemplo 1: permitir que um usuário tenha acesso total a tipos de recurso específicos do MemoryDB
- Exemplo 2: negar a um usuário o acesso a um cluster.

Exemplo 1: permitir que um usuário tenha acesso total a tipos de recurso específicos do MemoryDB

A seguinte política permite explicitamente o acesso total da `account-id` especificada a todos os recursos do tipo grupo de sub-redes, grupo de segurança e cluster.

```
{
        "Sid": "Example1",
        "Effect": "Allow",
        "Action": "memorydb:*",
        "Resource": [
             "arn:aws:memorydb:us-east-1:account-id:subnetgroup/*",
             "arn:aws:memorydb:us-east-1:account-id:securitygroup/*",
             "arn:aws:memorydb:us-east-1:account-id:cluster/*"
        ]
}
```

Exemplo 2: negar a um usuário o acesso a um cluster.

O exemplo a seguir nega explicitamente o acesso especificado da `account-id` a um determinado cluster.

```
{
        "Sid": "Example2",
        "Effect": "Deny",
        "Action": "memorydb:*",
        "Resource": [
                "arn:aws:memorydb:us-east-1:account-id:cluster/name"
        ]
}
```

Uso de funções vinculadas ao serviço para o Amazon MemoryDB para Redis

O Amazon MemoryDB para Redis usa funções vinculadas a AWS Identity and Access Management serviços (IAM). Uma função vinculada ao serviço é um tipo exclusivo de função do IAM vinculada diretamente a um AWS serviço, como o Amazon MemoryDB for Redis. As funções vinculadas ao serviço do Amazon MemoryDB para Redis são predefinidas pelo Amazon MemoryDB para Redis. Elas incluem todas as permissões que o serviço exige para chamar os serviços da AWS em nome dos seus clusters.

Uma função vinculada ao serviço facilita a configuração do Amazon MemoryDB para Redis porque você não precisa adicionar manualmente as permissões necessárias. As funções já existem na sua AWS conta, mas estão vinculadas aos casos de uso do Amazon MemoryDB para Redis e têm permissões predefinidas. Somente o Amazon MemoryDB para Redis pode assumir essas funções, e somente essas funções podem usar a política de permissões predefinida. Você pode excluir os perfis somente depois de primeiro excluir seus recursos relacionados. Isso protege seus recursos do Amazon MemoryDB para Redis porque você não pode remover inadvertidamente as permissões necessárias para acessar os recursos.

Para obter informações sobre outros serviços compatíveis com funções vinculadas a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Sim na coluna Função vinculada a serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado a esse serviço.

Sumário

Permissões de função vinculadas ao serviço para o Amazon MemoryDB para Redis

O Amazon MemoryDB para Redis usa a função vinculada ao serviço chamada AWSServiceRoleForMemoryDB— Essa política permite que o MemoryDB gerencie AWS recursos em seu nome conforme necessário para gerenciar seus clusters.

A política de permissões AWSServiceRoleForMemoryDB de função vinculada ao serviço permite que o Amazon MemoryDB for Redis conclua as seguintes ações nos recursos especificados:

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "ec2:CreateTags"
                ],
                "Resource": "arn:aws:ec2:*:*:network-interface/*",
                "Condition": {
                    "StringEquals": {
                        "ec2:CreateAction": "CreateNetworkInterface"
                    },
                    "ForAllValues:StringEquals": {
                        "aws:TagKeys": [
```

```
                    "AmazonMemoryDBManaged"
                ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:CreateNetworkInterface"
        ],
        "Resource": [
            "arn:aws:ec2:*:*:network-interface/*",
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:security-group/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:DeleteNetworkInterface",
            "ec2:ModifyNetworkInterfaceAttribute"
        ],
        "Resource": "arn:aws:ec2:*:*:network-interface/*",
        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/AmazonMemoryDBManaged": "true"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:DeleteNetworkInterface",
            "ec2:ModifyNetworkInterfaceAttribute"
        ],
        "Resource": "arn:aws:ec2:*:*:security-group/*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeAvailabilityZones",
            "ec2:DescribeSubnets",
```

```
                "ec2:DescribeVpcs"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/MemoryDB"
                }
            }
        }
    ]
}
```

Para ter mais informações, consulte Política gerenciada pela AWS: MemoryDBServiceRolePolicy.

Para permitir que uma entidade do IAM crie funções AWSServiceRoleForMemoryDB vinculadas ao serviço

Adicione a seguinte declaração de política às permissões dessa entidade IAM:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/memorydb.amazonaws.com/
AWSServiceRoleForMemoryDB*",
    "Condition": {"StringLike": {"iam:AWS ServiceName": "memorydb.amazonaws.com"}}
}
```

Para permitir que uma entidade do IAM exclua funções AWSServiceRoleForMemoryDB vinculadas ao serviço

Adicione a seguinte declaração de política às permissões dessa entidade IAM:

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/memorydb.amazonaws.com/
AWSServiceRoleForMemoryDB*",
    "Condition": {"StringLike": {"iam:AWS ServiceName": "memorydb.amazonaws.com"}}
}
```

Como alternativa, você pode usar uma política AWS gerenciada para fornecer acesso total ao Amazon MemoryDB for Redis.

Criação de uma função vinculada ao serviço (IAM)

Você pode criar uma função vinculada ao serviço usando o console do IAM, a CLI ou a API.

Criação de uma função vinculada ao serviço (console do IAM)

Você pode usar o console do IAM para criar uma função vinculada ao serviço.

Para criar uma função vinculada ao serviço (console)

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação esquerdo do console IAM, escolha Funções. Em seguida, escolha Criar nova função.
3. Em Selecionar tipo de entidade confiável, selecione Serviço da AWS .
4. Em Ou selecione um serviço para visualizar seus casos de uso, escolha MemoryDB.
5. Escolha Próximo: permissões.
6. Em Nome da política, observe que `MemoryDBServiceRolePolicy` é necessário para esta função. Escolha Próximo: tags.
7. Observe que não há suporte para as tags para funções vinculadas ao serviço. Escolha Próximo: análise.
8. (Opcional) Em Descrição da função, edite a descrição para a nova função vinculada ao serviço.
9. Revise a função e escolha Criar função.

Criação de uma função vinculada ao serviço (CLI do IAM)

Você pode usar as operações do IAM do AWS Command Line Interface para criar uma função vinculada ao serviço. Essa função pode incluir a política de confiança e as políticas em linha de que o serviço precisa para assumir a função.

Para criar uma função vinculada ao serviço (CLI)

Use a seguinte operação:

```
$ aws iam create-service-linked-role --aws-service-name memorydb.amazonaws.com
```

Criação de uma função vinculada ao serviço (API do IAM)

Você pode usar a API do IAM para excluir uma função vinculada ao serviço. Essa função pode conter a política de confiança e as políticas em linha de que o serviço precisa para assumir a função.

Para criar uma função vinculada ao serviço (API)

Use a chamada da API CreateServiceLinkedRole. Na solicitação, especifique o nome do serviço na forma de `memorydb.amazonaws.com`.

Edição da descrição de uma função vinculada a um serviço para o Amazon MemoryDB para Redis

O Amazon MemoryDB para Redis não permite que você edite a função vinculada ao AWSServiceRoleForMemoryDB 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 da função usando o IAM.

Edição da descrição de uma função vinculada ao serviço (console do IAM)

Também é possível usar o console do IAM para editar a descrição de uma função vinculada ao serviço.

Para editar a descrição de uma função vinculada ao serviço (console)

1. No painel de navegação esquerdo do console IAM, escolha Funções.
2. Escolha o nome da função a ser modificada.
3. No extremo direito da Descrição da função, escolha Editar.
4. Insira uma nova descrição na caixa e escolha Salvar.

Edição da descrição de uma função vinculada ao serviço (CLI do IAM)

Você pode usar as operações do IAM do AWS Command Line Interface para editar uma descrição de função vinculada ao serviço.

Para alterar a descrição de uma função (CLI)

1. (Opcional) Para ver a descrição atual de uma função, use a operação AWS CLI for IAM`get-role`.

   Example

   ```
   $ aws iam get-role --role-name AWSServiceRoleForMemoryDB
   ```

   Use o nome da função, não o nome de recurso da Amazon (ARN), para fazer referência às funções com as operações da CLI. Por exemplo, se uma função tiver o seguinte nome de recurso da Amazon (ARN): `arn:aws:iam::123456789012:role/myrole`, você fará referência à função como **`myrole`**.
2. Para atualizar a descrição de uma função vinculada ao serviço, use a operação AWS CLI for IAM. `update-role-description`

   Para Linux, macOS ou Unix:

   ```
   $ aws iam update-role-description \
       --role-name AWSServiceRoleForMemoryDB \
       --description "new description"
   ```

   Para Windows:

   ```
   $ aws iam update-role-description ^
       --role-name AWSServiceRoleForMemoryDB ^
       --description "new description"
   ```

Edição da descrição de uma função vinculada ao serviço (API do IAM)

Você pode usar a API do IAM para editar uma descrição de função vinculada ao serviço.

Para alterar a descrição de uma função (API)

1. (Opcional) Para visualizar a descrição atual de uma função, use a operação da API do IAM GetRole.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=GetRole
      &RoleName=AWSServiceRoleForMemoryDB
      &Version=2010-05-08
      &AUTHPARAMS
   ```

2. Para atualizar uma descrição de função, use a operação da API do IAM UpdateRoleDescription.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=UpdateRoleDescription
      &RoleName=AWSServiceRoleForMemoryDB
      &Version=2010-05-08
      &Description="New description"
   ```

Excluir uma função vinculada a serviço para Amazon MemoryDB para Redis

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 limpar sua função vinculada ao serviço antes de excluí-la.

O Amazon MemoryDB para Redis não exclui a função vinculada ao serviço para você.

Limpar uma função vinculada ao serviço

Antes de usar o IAM para excluir uma função vinculada a um serviço, primeiro confirme se a função não tem recursos (clusters) associados a ela.

Para verificar se a função vinculada ao serviço tem uma sessão ativa no console do IAM

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

2. No painel de navegação esquerdo do console IAM, escolha Funções. Em seguida, escolha o nome (não a caixa de seleção) da AWSServiceRoleForMemoryDB função.
3. Na página Resumo para a função selecionada, escolha a guia Consultor de Acesso.
4. Na guia Consultor de Acesso, revise a atividade recente para a função vinculada ao serviço.

Para excluir recursos do Amazon MemoryDB para Redis que exigem AWSServiceRoleForMemoryDB (console)

- Para excluir um cluster, consulte o seguinte:
  - Usando o AWS Management Console
  - Usando o AWS CLI
  - Usando a API do MemoryDB

Exclusão de uma função vinculada ao serviço (console do IAM)

É possível usar o console do IAM para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (console)

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação esquerdo do console IAM, escolha Funções. Selecione a caixa de marcação ao lado do nome da função que você deseja excluir, não o nome ou a linha em si.
3. Em ações de Função na parte superior da página, escolha a função Excluir.
4. Na página de confirmação, revise os dados do último acesso ao serviço, que mostram quando cada uma das funções selecionadas acessou um AWS serviço pela última vez. Isso ajuda você a confirmar se a função está ativo no momento. Se quiser prosseguir, escolha Sim, Excluir para enviar a função vinculada ao serviço para exclusão.
5. Monitore as notificações do console do IAM para progresso da exclusão da função vinculada ao serviço. Como a exclusão da função vinculada ao serviço do IAM é assíncrona, depois de enviar a função para exclusão, a tarefa pode ou não ser bem-sucedida. Se a tarefa obtiver êxito, você poderá escolher Visualizar Detalhes ou Visualizar Recursos a partir das notificações para saber por que a exclusão falhou.

Exclusão de uma função vinculada ao serviço (CLI do IAM)

Você pode usar as operações do IAM do AWS Command Line Interface para excluir uma função vinculada ao serviço.

Para excluir uma função vinculado ao serviço (CLI)

1. Se você não souber o nome da função vinculada ao serviço que deseja excluir, insira o seguinte comando. Esse comando lista as funções e os nomes de recursos da Amazon (ARNs) em sua conta.

   ```
   $ aws iam get-role --role-name role-name
   ```

   Use o nome da função, não o nome de recurso da Amazon (ARN), para fazer referência às funções com as operações da CLI. Por exemplo, se uma função tiver o ARN `arn:aws:iam::123456789012:role/myrole`, você fará referência à função como **`myrole`**.

2. Como uma função vinculada ao serviço não podem ser excluída se estiver sendo usada ou tiver recursos associados, você deverá enviar uma solicitação de exclusão. Essa solicitação poderá ser negada se essas condições não forem atendidas. Você deve capturar o `deletion-task-id` da resposta para verificar o status da tarefa de exclusão. Insira o seguinte para enviar uma solicitação de exclusão de função vinculada ao serviço.

   ```
   $ aws iam delete-service-linked-role --role-name role-name
   ```

3. Insita o seguinte para verificar o estado da tarefa de exclusão.

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada informará o motivo de falha para que você possa solucionar o problema.

Exclusã de uma função vinculada ao serviço (API do IAM)

É possível usar a API do IAM para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (API)

1. Para enviar uma solicitação de exclusão de um roll vinculada ao serviço, chame DeleteServiceLinkedRole. Na solicitação, especifique o nome da função.

   Como uma função vinculada ao serviço não podem ser excluída se estiver sendo usada ou tiver recursos associados, você deverá enviar uma solicitação de exclusão. Essa solicitação poderá ser negada se essas condições não forem atendidas. Você deve capturar o `DeletionTaskId` da resposta para verificar o status da tarefa de exclusão.
2. Para verificar o status da exclusão, chame GetServiceLinkedRoleDeletionStatus. Na solicitação, especifique o `DeletionTaskId`.

   O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada informará o motivo de falha para que você possa solucionar o problema.

Políticas gerenciadas pela AWS para MemoryDB para Redis

Para adicionar permissões a usuários, grupos e funções, é mais fácil usar políticas gerenciadas pela AWS do que gravar políticas por conta própria. É necessário tempo e experiência para criar políticas gerenciadas pelo cliente do IAM que fornecem à sua equipe apenas as permissões de que precisam. Para começar rapidamente, é possível usar nossas políticas gerenciadas pela AWS. Essas políticas abrangem casos de uso comuns e estão disponíveis na sua conta da AWS. Para obter mais informações sobre políticas gerenciadas pela AWS, consulte Políticas gerenciadas pela AWS no Guia do usuário do IAM.

Os serviços da AWS mantêm e atualizam políticas gerenciadas pela AWS. Não é possível alterar as permissões em políticas gerenciadas pela AWS. Os serviços ocasionalmente acrescentam permissões adicionais a uma política gerenciada pela AWS para oferecer suporte a novos recursos. Esse tipo de atualização afeta todas as identidades (usuários, grupos e funções) em que a política está anexada. É mais provável que os serviços atualizem uma política gerenciada pela AWS quando um novo recurso for iniciado ou novas operações se tornarem disponíveis. Os serviços não removem permissões de uma política gerenciada pela AWS, portanto, as atualizações de políticas não suspendem suas permissões existentes.

Além disso, a AWS oferece suporte a políticas gerenciadas para funções de trabalho que abrangem vários serviços. Por exemplo, a política gerenciada pela AWS denominada ReadOnlyAccess fornece

acesso somente leitura a todos os serviços e recursos da AWS. Quando um serviço executa um novo recurso, a AWS adiciona permissões somente leitura para novas operações e recursos. Para obter uma lista e descrições das políticas de funções de trabalho, consulte Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.

Política gerenciada pela AWS: MemoryDBServiceRolePolicy

Você não pode anexar a política gerenciada pela AWS MemoryDBServiceRolePolicy às identidades em sua conta. Essa política é parte da função vinculada ao serviço MemoryDB da AWS. Essa função permite que o serviço gerencie interfaces de rede e grupos de segurança em sua conta.

O MemoryDB usa as permissões desta política para gerenciar grupos de segurança e interfaces de rede do EC2. Isso é necessário para gerenciar clusters do MemoryDB.

Detalhes da permissão

Esta política inclui as seguintes permissões.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "ec2:CreateTags"
                ],
                "Resource": "arn:aws:ec2:*:*:network-interface/*",
                "Condition": {
                    "StringEquals": {
                        "ec2:CreateAction": "CreateNetworkInterface"
                    },
                    "ForAllValues:StringEquals": {
                        "aws:TagKeys": [
                            "AmazonMemoryDBManaged"
                        ]
                    }
                }
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteNetworkInterface",
                "ec2:ModifyNetworkInterfaceAttribute"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/AmazonMemoryDBManaged": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteNetworkInterface",
                "ec2:ModifyNetworkInterfaceAttribute"
            ],
            "Resource": "arn:aws:ec2:*:*:security-group/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs"
            ],
            "Resource": "*"
        },
```

```
            {
                "Effect": "Allow",
                "Action": [
                    "cloudwatch:PutMetricData"
                ],
                "Resource": "*",
                "Condition": {
                    "StringEquals": {
                        "cloudwatch:namespace": "AWS/MemoryDB"
                    }
                }
            }
        ]
    }
```

Políticas (predefinidas) gerenciadas pela AWS do MemoryDB para Redis

A AWS aborda muitos casos de uso comuns fornecendo políticas autônomas do IAM que são criadas e administradas pela AWS. As políticas gerenciadas concedem permissões necessárias para casos de uso comuns, de maneira que você possa evitar a necessidade de investigar quais permissões são necessárias. Para obter mais informações, consulte Políticas gerenciadas da AWS no Guia do usuário do IAM.

As seguintes políticas gerenciadas pela AWS, que podem ser associadas a usuários na sua conta, são específicas do MemoryDB:

AmazonMemoryDBReadOnlyAccess

É possível anexar a política `AmazonMemoryDBReadOnlyAccess` a suas identidades do IAM. Esta política concede permissões administrativas que oferecem acesso somente leitura a todos os recursos do MemoryDB.

AmazonMemoryDBReadOnlyAccess: concede acesso somente leitura aos recursos do MemoryDB para Redis.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "memorydb:Describe*",
   "memorydb:List*"
```

```
  ],
  "Resource": "*"
 }]
}
```

AmazonMemoryDBFullAccess

É possível anexar a política `AmazonMemoryDBFullAccess` a suas identidades do IAM. Essa política concede permissões administrativas que oferecem acesso total a todos os recursos do MemoryDB.

AmazonMemoryDBFullAccess: concede acesso total a todos os recursos do MemoryDB para Redis.

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Effect": "Allow",
   "Action": "memorydb:*",
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": "iam:CreateServiceLinkedRole",
   "Resource": "arn:aws:iam::*:role/aws-service-role/memorydb.amazonaws.com/
AWSServiceRoleForMemoryDB",
   "Condition": {
    "StringLike": {
     "iam:AWSServiceName": "memorydb.amazonaws.com"
    }
   }
  }
 ]
}
```

Além disso, você pode criar políticas do IAM personalizadas para conceder permissões para ações de API do MemoryDB para Redis. Você pode anexar essas políticas personalizadas a usuários ou grupos do IAM que exijam essas permissões.

Atualizações do MemoryDB para políticas gerenciadas da AWS

Visualizar detalhes sobre atualizações em políticas gerenciadas pela AWS para o MemoryDB desde que esse serviço começou a rastrear essas alterações. Para receber alertas automáticos sobre alterações feitas nesta página, inscreva-se no feed de RSS na página Document History (Histórico do documento) do MemoryDB.

Alteração	Descrição	Data
AmazonMemoryDBFullAccess – Adicionar uma política	O MemoryDB adicionou novas permissões para descrever e listar recursos compatíveis. Essas permissões são necessárias para que o MemoryDB consulte todos os recursos compatíveis em uma conta.	07/10/2021
AmazonMemoryDBReadOnlyAccess – Adicionar uma política	O MemoryDB adicionou novas permissões para descrever e listar recursos compatíveis. Essas permissões são necessárias para que o MemoryDB crie aplicações baseadas em conta consultando todos os recursos compatíveis em uma conta.	07/10/2021
O MemoryDB começou a monitorar alterações	Inicialização do serviço	19/08/2021

Permissões da API do MemoryDB: referência de condições, recursos e ações

Ao configurar controle de acesso e escrever políticas de permissões para anexar a uma política do IAM (baseada em identidade ou em recursos), use a tabela a seguir como referência. A tabela lista cada operação da API do MemoryDB para Redis e as ações correspondentes para as quais você pode conceder permissões para executar a ação. Você especifica as ações no campo `Action` da política e um valor de recurso no campo `Resource`da política. Salvo indicação em contrário, o recurso é obrigatório. Alguns campos incluem um recurso obrigatório e recursos opcionais. Quando não há ARN de recurso, o recurso na política é um caractere curinga (*).

> **Note**
>
> Para especificar uma ação, use o prefixo `memorydb:` seguido do nome da operação da API (por exemplo, `memorydb:DescribeClusters`).

Logging e monitoramento

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do MemoryDB for Redis e de suas outras soluções. AWS AWS fornece as seguintes ferramentas de monitoramento para monitorar o MemoryDB, relatar quando algo está errado e realizar ações automáticas quando apropriado:

- A Amazon CloudWatch monitora seus AWS recursos e os aplicativos em que você executa AWS em tempo real. É possível coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode CloudWatch rastrear o uso da CPU ou outras métricas de suas instâncias do Amazon EC2 e iniciar automaticamente novas instâncias quando necessário. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.
- O Amazon CloudWatch Logs permite que você monitore, armazene e acesse seus arquivos de log a partir de instâncias do Amazon EC2 e de outras fontes. 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.
- AWS CloudTrailcaptura chamadas de API e eventos relacionados feitos por ou em nome de sua AWS conta e entrega os arquivos de log para um bucket do Amazon S3 que você especificar. Você pode identificar quais usuários e contas ligaram AWS, o endereço IP de origem a partir do

qual as chamadas foram feitas e quando elas ocorreram. Para obter mais informações, consulte o AWS CloudTrail Guia de usuário do .

Monitorando o MemoryDB para Redis com a Amazon CloudWatch

Você pode monitorar o MemoryDB for Redis usando 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.

As seções a seguir listam as métricas e dimensões do MemoryDB.

Tópicos

- Métricas em nível de host
- Métricas para MemoryDB
- Quais métricas devo monitorar?
- Escolher estatísticas e períodos de métricas
- CloudWatch Métricas de monitoramento

Métricas em nível de host

O namespace `AWS/MemoryDB` inclui as seguintes métricas em nível de host para nós individuais.

Consulte também

- Métricas para MemoryDB

Métrica	Descrição	Unidade
`CPUUtilization`	A percentagem de utilização da CPU para o host inteiro. Como o Redis é de thread único, recomendamos que você monitore a métrica `EngineCPUUtilization` , para nós com quatro ou mais vCPUs.	Percentual

Métrica	Descrição	Unidade
FreeableMemory	A quantidade de memória livre disponível no host. Ela é derivada da RAM, dos buffers e do cache que o sistema operacional relata como passíveis de liberação.	Bytes
NetworkBytesIn	O número de bytes que o host leu da rede.	Bytes
NetworkBytesOut	O número de bytes enviados em todas as interfaces de rede pela instância.	Bytes
NetworkPacketsIn	O número de pacotes recebidos em todas as interfaces de rede pela instância. Essa métrica identifica o volume de tráfego de entrada em termos do número de pacotes em uma única instância.	Contagem
NetworkPacketsOut	O número de pacotes enviados em todas as interfaces de rede pela instância. Essa métrica identifica o volume de tráfego de saída em termos do número de pacotes em uma única instância.	Contagem
NetworkBandwidthInAllowanceExceeded	Número de pacotes moldados porque a largura de banda agregada de entrada excedeu o máximo para a instância.	Contagem
NetworkConntrackAllowanceExceeded	Número de pacotes moldados porque o monitoramento da conexão excedeu o máximo para a instância e não foi possível estabelecer novas conexões. Isso pode resultar em perda de pacotes para tráfego indo para a instância ou vindo da instância	Contagem
NetworkBandwidthOutAllowanceExceeded	Número de pacotes moldados porque a largura de banda agregada de saída excedeu o máximo para a instância.	Contagem

Métrica	Descrição	Unidade
NetworkPacketsPerSecondAllowanceExceeded	Número de pacotes moldados porque o valor bidirecional de pacotes por segundo excedeu o máximo para a instância.	Contagem
NetworkMaxBytesIn	A intermitência máxima de bytes recebidos em cada minuto.	Bytes
NetworkMaxBytesOut	A intermitência máxima de bytes transmitidos em cada minuto.	Bytes
NetworkMaxPacketsIn	A intermitência máxima de pacotes recebidos em cada minuto.	Contagem
NetworkMaxPacketsOut	A intermitência máxima de pacotes transmitidos em cada minuto.	Contagem
SwapUsage	A quantidade de troca usada no host.	Bytes

Métricas para MemoryDB

O namespace AWS/MemoryDB inclui as métricas de Redis a seguir.

Com exceção de ReplicationLag e EngineCPUUtilization, essas métricas são derivadas do comando info do Redis. Cada métrica é calculada no nível do nó.

Para a documentação completa do comando info do Redis, consulte http://redis.io/commands/info.

Consulte também

- Métricas em nível de host

Métrica	Descrição	Unidade
ActiveDefragHits	O número de realocações de valor por minuto executada pelo processo de desfragme	Número

Métrica	Descrição	Unidade
	ntação ativo. Deriva da estatística `active_defrag_hits` no comando INFO do Redis.	
`AuthenticationFailures`	O número total de tentativas falhadas de autenticação para Redis usando o comando AUTH. É possível encontrar mais informações sobre falhas de autenticação individuais usando o comando ACL LOG. Sugerimos definir um alarme para detectar tentativas de acesso não autorizadas.	Contagem
`BytesUsedForMemoryDB`	O número total de bytes alocados pelo MemoryDB para todos os fins, incluindo o conjunto de dados, buffers e assim por diante.	Bytes
	`Dimension: Tier=SSD` para clusters que usam Classificação de dados em níveis: o número total de bytes usados pela SSD.	Bytes
	`Dimension: Tier=Memory` para clusters que usam Classificação de dados em níveis: o número total de bytes usados pela memória. Esse é o valor de `used_memory` estática em INFO do Redis.	Bytes
`BytesReadFromDisk`	O número total de bytes lidos no disco por minuto. Compatível somente para clusters usando Classificação de dados em níveis.	Bytes
`BytesWrittenToDisk`	O número total de bytes gravados no disco por minuto. Compatível somente para clusters usando Classificação de dados em níveis.	Bytes

Métrica	Descrição	Unidade
`CommandAuthorizationFailures`	O número total de tentativas falhadas por usuários para executar comandos que eles não têm permissão para chamar. É possível encontrar mais informações sobre falhas de autenticação individuais usando o comando ACL LOG. Sugerimos definir um alarme para detectar tentativas de acesso não autorizadas.	Contagem
`CurrConnections`	O número de conexões de clientes, excluindo conexões de réplicas de leitura. O MemoryDB usa de duas a quatro das conexões para monitorar o cluster em cada caso. Deriva da estatística `connected_clients` no comando INFO do Redis.	Contagem
`CurrItems`	O número de itens no cache. Deriva da estatística `keyspace` do Redis, somando todas as chaves em todo o keyspace.	Contagem
	`Dimension: Tier=Memory` para clusters usando Classificação de dados em níveis. O número de itens em memória.	Contagem
	`Dimension: Tier=SSD` (unidades de estado sólido) para clusters usando Classificação de dados em níveis. O número de itens em SSD.	Contagem
`DatabaseMemoryUsagePercentage`	Percentual de memória disponível para o cluster que está em uso. Isso é calculado usando `used_memory/maxmemory` de Redis INFO.	Percentual

Métrica	Descrição	Unidade
`DatabaseCapacityUsagePercentage`	Porcentagem da capacidade total de dados para o cluster que está em uso. Em instâncias com camadas de dados, a métrica é calculada como`(used_memory - mem_not_counted_for_evict + SSD used) / (maxmemory + SSD total capacity)` , onde `used_memory` e `maxmemory` é obtida do Redis INFO. Em todos os outros casos, a métrica é calculada usando`used_memory/maxmemory` .	Percentual
`DB0AverageTTL`	Expõe o `avg_ttl` de DBO a partir da estatística `keyspace` do comando INFO do Redis.	Milissegundos

Métrica	Descrição	Unidade
`EngineCPUUtilization`	Fornece utilização de CPU do thread de mecanismo do Redis. Como o Redis é de thread único, você pode usar essa métrica para analisar a carga do próprio processo do Redis. A métrica `EngineCPUUtilization` fornece uma visibilidade mais precisa do processo do Redis. Você pode usá-la em conjunto com a métrica `CPUUtilization`. `CPUUtilization` expõe a utilização de CPU da instância do servidor como um todo, incluindo outros processos de sistema operacional e de gerenciamento. Para tipos de nós maiores com quatro vCPUs ou mais, use a métrica `EngineCPUUtilization` para monitorar e definir limites para a escalabilidade. **Note** Em um host MemoryDB, os processos em segundo plano monitoram o host para oferecer uma experiência de banco de dados gerenciado. Esses processos em segundo plano podem ocupar uma parte significativa da workload da CPU. Isso não é significativo em hosts maiores com mais de duas vCPUs. Mas pode afetar hosts menores com 2vCPUs ou menos. Se você monitorar apenas a métrica `EngineCPUUtilization`, desconhecerá situações em que o host está sobrecarregado com o alto uso da CPU do Redis e o alto uso da CPU dos processos de monitoramento em segundo plano. Portanto, recomenda	Percentual

Métrica	Descrição	Unidade
	mos monitorar a métrica `CPUUtilization` para hosts com duas vCPUs ou menos.	
`Evictions`	O número de chaves que foram removidas devido ao limite `maxmemory` . Deriva da estatística `evicted_keys` no comando INFO do Redis.	Contagem
`IsPrimary`	Indica se o nó é o nó primário do fragmento atual. A métrica pode ser 0 (não primária) ou 1 (primária).	Contagem
`KeyAuthorizationFailures`	O número total de tentativas falhadas por usuários de acessar chaves que eles não têm permissão para acessar. É possível encontrar mais informações sobre falhas de autenticação individuais usando o comando ACL LOG. Sugerimos definir um alarme para detectar tentativas de acesso não autorizadas.	Contagem
`KeyspaceHits`	O número de buscas de chaves somente leitura bem-sucedidas no dicionário principal . Deriva da estatística `keyspace_hits` no comando INFO do Redis.	Contagem
`KeyspaceMisses`	O número de buscas de chaves somente leitura malsucedidas no dicionário principal. Deriva da estatística `keyspace_misses` no comando INFO do Redis.	Contagem

Métrica	Descrição	Unidade
KeysTracked	O número de chaves que estão sendo monitoradas pelo monitoramento de chaves do Redis como um percentual de `tracking-table-max-keys` . O monitoramento de chaves é usado para ajudar o cache do lado do cliente e notifica os clientes quando as chaves são modificadas.	Contagem
MaxReplicationThroughput	O throughput de replicação máximo observado durante o último ciclo de medição.	Bytes por segundo
MemoryFragmentationRatio	Indica a eficiência na alocação de memória do mecanismo Redis. Certos limites significarão comportamentos diferentes. O valor recomendado é ter fragmentação acima de 1,0. Isso é calculado a partir de `mem_fragmentation_ratio statistic` do comando INFO do Redis.	Número
NewConnections	O número total de conexões que foram aceitas pelo servidor durante esse período. Deriva da estatística `total_connections_received` no comando INFO do Redis.	Contagem
NumItemsReadFromDisk	O número total de itens recuperados do disco por minuto. Compatível somente para clusters usando Classificação de dados em níveis.	Contagem
NumItemsWrittenToDisk	O número total de itens gravados no disco por minuto. Compatível somente para clusters usando Classificação de dados em níveis.	Contagem

Métrica	Descrição	Unidade
`PrimaryLinkHealthStatus`	Esse status tem dois valores: 0 ou 1. O valor 0 indica que os dados no nó primário do MemoryDB não estão sincronizados com o Redis no EC2. O valor de 1 indica que os dados não estão sincronizados.	Booleano
`Reclaimed`	O número total de eventos de expiração de chaves. Deriva da estatística `expired_keys` no comando INFO do Redis.	Contagem
`ReplicationBytes`	Para nós em uma configuração replicada, `ReplicationBytes` informa o número de bytes que a primária está enviando para todas as suas réplicas. Essa métrica é represent ativa da carga de gravação no cluster. Deriva da estatística `master_repl_offset` no comando INFO do Redis.	Bytes
`ReplicationDelayedWriteCommands`	Número de comandos de gravação que foram atrasados devido à replicação síncrona. A replicação pode ser adiada devido a vários fatores, por exemplo, congestionamento da rede ou excesso da taxa de transferência máxima de replicação.	Contagem
`ReplicationLag`	Essa métrica é aplicável somente para um nó de em execução como uma réplica de leitura. Ela representa o tempo decorrido, em segundos, até a réplica aplicar alterações do nó primário.	Segundos

A seguir estão agregações de determinados tipos de comandos, derivados de info commandstats: A seção commandstats fornece estatísticas com base no tipo de comando, incluindo o número de chamadas.

Para obter uma lista completa dos comandos disponíveis, consulte comandos do redis na documentação do Redis.

Métrica	Descrição	Unidade
`EvalBasedCmds`	O número total de comandos para comandos baseados em avaliação. Isso é derivado da estatística `commandstats` do Redis. Isso é derivado da estatística `commandstats` do Redis pela soma de eval, evalsha.	Contagem
`GeoSpatialBasedCmds`	O número total de comandos para comandos baseados em dados geoespaciais. Isso é derivado da estatística `commandstats` do Redis. Ele é derivado somando todos o tipos de comandos geo: geoadd, geodist, geohash, geopos, georadius, e georadiusbymember.	Contagem
`GetTypeCmds`	O número total de comandos do tipo read-only. Isso é derivado da estatística `commandstats` do Redis, somando todos os comandos do tipo read-only (get, hget, scard, lrange, etc.).	Contagem
`HashBasedCmds`	O número total de comandos baseados em hash. Isso é derivado da estatística `commandstats` do Redis, somando todos os comandos que atuam em um ou mais hashes (hget, hkeys, hvals, hdel, etc.).	Contagem
`HyperLogLogBasedCmds`	O número total de comandos baseados em `HyperLogLog` . Isso é derivado da estatística `commandstats` do Redis, somando todos os comandos do tipo pf (pfadd, pfcount, pfmerge, etc.).	Contagem
`JsonBasedCmds`	O número total de comandos que são baseados em JSON. Deriva da estatística `commandstats` do Redis somando todos os	Contagem

Métrica	Descrição	Unidade
	comandos que atuam em uma ou mais objetos do documento JSON.	
KeyBasedCmds	O número total de comandos baseados em chave. Isso é derivado da estatística commandstats do Redis, somando todos os comandos que atuam em uma ou mais chaves em várias estruturas de dados (del, expire, rename, etc.).	Contagem
ListBasedCmds	O número total de comandos baseados em lista. Isso é derivado da estatística commandstats do Redis, somando todos os comandos que atuam em uma ou mais listas (lindex, lrange, lpush, ltrim, etc.).	Contagem
PubSubBasedCmds	O número total de comandos para a funcionalidade pub/sub. Isso é derivado das estatísticas commandstats do Redis somando todos os comandos usados para a funcionalidade pub/sub: psubscribe, publish, pubsub, punsubscribe, subscribe e unsubscribe.	Contagem
SearchBasedCmds	O número total de comandos de pesquisa e índice secundário, incluindo comandos de leitura e gravação. Isso é derivado da estatística commandstats do Redis, somando todos os comandos de pesquisa que atuam em índices secundários.	Contagem
SearchBasedGetCmds	Número total de comandos somente leitura secundários de índice e pesquisa. Isso é derivado da estatística commandstats do Redis, somando todos os comandos Get secundários de índice e pesquisa.	Contagem

Métrica	Descrição	Unidade
SearchBasedSetCmds	Número total de comandos de gravação secundários de índice e pesquisa. Isso é derivado da estatística commandstats do Redis, somando todos os comandos Set secundários de índice e pesquisa.	Contagem
SearchNumberOfIndexes	Número total de índices.	Contagem
SearchNumberOfIndexedKeys	Número total de chaves do Redis indexadas	Contagem
SearchTotalIndexSize	Memória (bytes) usada por todos os índices.	Bytes
SetBasedCmds	O número total de comandos que são baseados em conjuntos. Isso é derivado da estatística commandstats do Redis, somando todos os comandos que atuam em um ou mais conjuntos (scard, sdiff, sadd, sunion, etc.).	Contagem
SetTypeCmds	O número total de tipos de comando write. Isso é derivado da estatística commandstats do Redis, somando todos os tipos de comando mutative que operam em dados (set, hset, sadd, lpop, etc.).	Contagem
SortedSetBasedCmds	O número total de comandos que são classificados com base em conjuntos. Isso é derivado da estatística commandstats do Redis, somando todos os comandos que atuam em um ou mais conjuntos classificados (zcount, zrange, zrank, zadd, etc.).	Contagem

Métrica	Descrição	Unidade
`StringBasedCmds`	O número total de comandos baseados em string. Isso é derivado da estatística `commandstats` do Redis, somando todos os comandos que atuam em uma ou mais strings (strlen, setex, setrange, etc.).	Contagem
`StreamBasedCmds`	O número total de comandos que são baseados em fluxo. Isso é derivado da estatística `commandstats` do Redis, somando todos os comandos que atuam em um ou mais tipos de dados de fluxos (xrange, xlen, xadd, xdel, etc.).	Contagem

Quais métricas devo monitorar?

As CloudWatch métricas a seguir oferecem uma boa visão do desempenho do MemoryDB. Na maioria dos casos, recomendamos que você defina CloudWatch alarmes para essas métricas para que você possa tomar medidas corretivas antes que ocorram problemas de desempenho.

Métricas para monitorar

- CPUUtilization
- EngineCPUUtilization
- SwapUsage
- Evictions
- CurrConnections
- Memória
- Rede
- Replicação

CPUUtilization

Essa é uma métrica em nível de host relatada como uma porcentagem. Para ter mais informações, consulte Métricas em nível de host.

Para tipos de nós menores com 2 vCPUs ou menos, use a métrica `CPUUtilization` para monitorar a workload.

De modo geral, sugerimos que você defina o limite para 90% da CPU disponível. Como o Redis é de thread único, o valor limite real deve ser calculado como uma fração da capacidade total do nó. Por exemplo, suponha que você esteja usando um tipo de nó com dois núcleos. Nesse caso, o limite para CPUUtilization seria 90/2 ou 45%. Para saber o número de núcleos (vCPUs) que seu tipo de nó possui, consulte Precificação do MemoryDB.

Você precisará determinar seu próprio limite, com base no número de núcleos no nó que está usando. Se você exceder esse limite e sua workload principal for de solicitações de leitura, escale seu cluster adicionando réplicas de leitura. Se a workload principal for proveniente de solicitações de gravação, recomendamos que você adicione mais fragmentos para distribuir a workload de gravação em mais nós primários.

> **Tip**
>
> Em vez de usar a métrica de nível de host `CPUUtilization`, talvez você possa usar a métrica do Redis `EngineCPUUtilization`, que informa a porcentagem de uso no núcleo do mecanismo do Redis. Para ver se essa métrica está disponível em seus nós e para obter mais informações, consulte Métricas para MemoryDB.

Para tipos de nós maiores com 4 vCPUs ou mais, talvez você queira usar a métrica `EngineCPUUtilization`, que informa a porcentagem de uso no núcleo do mecanismo do Redis. Para ver se essa métrica está disponível em seus nós e para obter mais informações, consulte Métricas para MemoryDB.

EngineCPUUtilization

Para tipos de nós maiores com 4 vCPUs ou mais, talvez você queira usar a métrica `EngineCPUUtilization`, que informa a porcentagem de uso no núcleo do mecanismo do Redis. Para ver se essa métrica está disponível em seus nós e para obter mais informações, consulte Métricas para MemoryDB.

SwapUsage

Esta é uma métrica em nível de host relatada em bytes. Para ter mais informações, consulte Métricas em nível de host.

essa métrica não deve exceder 50 MB.

Evictions

Essa é uma métrica de mecanismo. Recomendamos que você determine seu próprio limite de alarme para essa métrica com base nas necessidades do seu aplicativo.

CurrConnections

Essa é uma métrica de mecanismo. Recomendamos que você determine seu próprio limite de alarme para essa métrica com base nas necessidades do seu aplicativo.

Um número crescente de CurrConnectionspode indicar um problema com seu aplicativo; você precisará investigar o comportamento do aplicativo para resolver esse problema.

Memória

A memória é um aspecto central do Redis. Compreender a utilização da memória do seu cluster é necessário para evitar a perda de dados e acomodar o crescimento futuro do seu conjunto de dados. Estatísticas sobre a utilização de memória de um nó estão disponíveis na seção de memória do comando INFO do Redis.

Rede

Um dos fatores determinantes para a capacidade de largura de banda da rede do cluster é o tipo de nó selecionado. Para obter mais informações sobre a capacidade de rede de seu nó, consulte Precificação do Amazon MemoryDB.

Replicação

O volume de dados que está sendo replicado é visível através da métrica `ReplicationBytes`. Você pode monitorar o `MaxReplicationThroughput` em relação ao throughput da capacidade de replicação. Recomenda-se adicionar mais fragmentos ao atingir o throughput máximo da capacidade de replicação.

`ReplicationDelayedWriteCommands` também pode indicar se a workload está excedendo o throughput da capacidade máxima de replicação. Para obter mais informações sobre a replicação no MemoryDB, consulte Entendendo a replicação do MemoryDB

Escolher estatísticas e períodos de métricas

Embora CloudWatch permita que você escolha qualquer estatística e período para cada métrica, nem todas as combinações serão úteis. Por exemplo, as estatísticas Average, Minimum e Maximum para CPUUtilization são úteis, mas a estatística Sum não é.

Todas as amostras do MemoryDB são publicadas por um período de 60 segundos para cada nó individual. Em qualquer período de 60 segundos, uma métrica de nó conterá apenas uma única amostra.

CloudWatch Métricas de monitoramento

O MemoryDB e o MemoryDB CloudWatch são integrados para que você possa reunir uma variedade de métricas. Você pode monitorar essas métricas usando CloudWatch o.

> **Note**
>
> Os exemplos a seguir exigem as ferramentas de linha de CloudWatch comando. Para obter mais informações CloudWatch e fazer o download das ferramentas para desenvolvedores, consulte a página CloudWatch do produto.

Os procedimentos a seguir mostram como usar CloudWatch para coletar estatísticas de espaço de armazenamento de um cluster na última hora.

> **Note**
>
> Os valores `StartTime` e `EndTime` fornecidos nos exemplos a seguir são para fins ilustrativos. Verifique se fez a substituição dos valores de hora inicial e final apropriados para seus nós.

Para obter informações sobre os limites do MemoryDB, consulte limites de serviço da AWS para o MemoryDB.

CloudWatch Métricas de monitoramento (console)

Para reunir estatísticas de utilização da CPU em um cluster

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Selecione os nós para os quais você deseja visualizar métricas.

   > **Note**
   >
   > Selecionar mais de 20 nós desabilita as métricas de visualização no console.

   a. Na página Clusters do AWS Management Console, clique no nome de um ou mais clusters.

      A página de detalhes do cluster é exibida.

   b. Clique na guia Nodes na parte superior da janela.

   c. Na guia Nós da janela de detalhes, selecione os nós de cache para os quais você deseja visualizar as métricas.

      Uma lista das CloudWatch métricas disponíveis aparece na parte inferior da janela do console.

   d. Clique na métrica CPU Utilization.

      O CloudWatch console será aberto, exibindo as métricas selecionadas. Você pode usar as caixas de listagem suspensas Statistic e Period e a guia Time Range para alterar as métricas exibidas.

Monitoramento de CloudWatch métricas usando a CloudWatch CLI

Para reunir estatísticas de utilização da CPU em um cluster

- Use o CloudWatch comando aws cloudwatch get-metric-statistics com os seguintes parâmetros (observe que os horários de início e término são mostrados apenas como exemplos; você precisará substituir os horários de início e término apropriados):

  Para Linux, macOS ou Unix:

```
aws cloudwatch get-metric-statistics CPUUtilization \
    --dimensions=ClusterName=mycluster,NodeId=0002" \
    --statistics=Average \
    --namespace="AWS/MemoryDB" \
    --start-time 2013-07-05T00:00:00 \
```

```
    --end-time 2013-07-06T00:00:00 \
    --period=60
```

Para Windows:

```
mon-get-stats CPUUtilization ^
    --dimensions=ClusterName=mycluster,NodeId=0002" ^
    --statistics=Average ^
    --namespace="AWS/MemoryDB" ^
    --start-time 2013-07-05T00:00:00 ^
    --end-time 2013-07-06T00:00:00 ^
    --period=60
```

Monitoramento de CloudWatch métricas usando a CloudWatch API

Para reunir estatísticas de utilização da CPU em um cluster

- Chame a CloudWatch API `GetMetricStatistics` com os seguintes parâmetros (observe que os horários de início e término são mostrados apenas como exemplos; você precisará substituir os horários de início e término apropriados):
  - `Statistics.member.1=Average`
  - `Namespace=AWS/MemoryDB`
  - `StartTime=2013-07-05T00:00:00`
  - `EndTime=2013-07-06T00:00:00`
  - `Period=60`
  - `MeasureName=CPUUtilization`
  - `Dimensions=ClusterName=mycluster,NodeId=0002`

  Example

```
http://monitoring.amazonaws.com/
    ?SignatureVersion=4
    &Action=GetMetricStatistics
    &Version=2014-12-01
    &StartTime=2013-07-16T00:00:00
    &EndTime=2013-07-16T00:02:00
```

```
&Period=60
&Statistics.member.1=Average
&Dimensions.member.1="ClusterName=mycluster"
&Dimensions.member.2="NodeId=0002"
&Namespace=Amazon/memorydb
&MeasureName=CPUUtilization
&Timestamp=2013-07-07T17%3A48%3A21.746Z
&AWS;AccessKeyId=<&AWS; Access Key ID>
&Signature=<Signature>
```

Monitoramento de eventos do MemoryDB para Redis

Quando ocorrem eventos significativos em um cluster, o MemoryDB envia uma notificação para um tópico específico do Amazon SNS. Exemplos incluem uma falha ao adicionar um nó, êxito ao adicionar um nó, a modificação de um grupo de segurança, e outros. Ao monitorar eventos chave, você pode se manter informado sobre o atual estado dos seus clusters e, dependendo do evento, poderá executar uma ação corretiva.

Tópicos

- Gerenciamento de notificações do Amazon SNS do MemoryDB
- Visualizar eventos do MemoryDB
- Notificações de eventos e o Amazon SNS

Gerenciamento de notificações do Amazon SNS do MemoryDB

Você pode configurar o MemoryDB para enviar notificações sobre eventos importantes do cluster usando o Amazon Simple Notification Service (Amazon SNS). Nestes exemplos, você configurará um cluster com o nome de recurso da Amazon (ARN) de um tópico do Amazon SNS para receber notificações.

> **Note**
>
> Esse tópico pressupõe que você tenha se cadastrado no Amazon SNS e configurado e assinado um tópico do Amazon SNS. Para obter informações sobre como fazer isso, consulte o Guia do desenvolvedor do Amazon Simple Notification Service.

Adição de um tópico do Amazon SNS

As seções a seguir mostram como adicionar um tópico do Amazon SNS usando o AWS console, o ou a API AWS CLI MemoryDB.

Adição de um tópico do Amazon SNS (console)

O procedimento a seguir mostra como adicionar um tópico do Amazon SNS para um cluster.

> **Note**
>
> Esse processo também pode ser usado para modificar o tópico do Amazon SNS.

Para adicionar ou modificar um tópico do Amazon SNS para um cluster (console)

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Em Clusters, escolha o cluster para o qual deseja adicionar ou modificar um ARN de tópico do Amazon SNS.
3. Escolha Modificar.
4. Em Modificar cluster em Tópico para notificação do SNS, escolha o tópico SNS que você deseja adicionar ou escolha Entrada manual de ARN e insira o ARN do tópico do Amazon SNS.
5. Escolha Modificar.

Adicionar um tópico do Amazon SNS (CLI AWS)

Para adicionar ou modificar um tópico do Amazon SNS para um cluster, use o AWS CLI comando. `update-cluster`

O seguinte exemplo de código adiciona um ARN de tópico do Amazon SNS a my-cluster.

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

Para obter mais informações, consulte UpdateCluster.

Adição de um tópico do Amazon SNS (API do MemoryDB)

Para adicionar ou atualizar um tópico do Amazon SNS para um cluster, chame a ação `UpdateCluster` com os seguintes parâmetros:

- `ClusterName=my-cluster`
- `SnsTopicArn=arn%3Aaws%3Asns%3Aus-east-1%3A565419523791%3AmemorydbNotifications`

Para adicionar ou atualizar um tópico do Amazon SNS para um cluster, chame a ação `UpdateCluster`.

Para obter mais informações, consulte UpdateCluster.

Habilitação e desabilitação de notificações do Amazon SNS

Você pode ativar ou desativar notificações para um cluster. Os procedimentos a seguir mostram como desativar notificações do Amazon SNS.

Habilitação e desabilitação de notificações do Amazon SNS (console)

Para desativar as notificações do Amazon SNS usando o AWS Management Console

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. Selecione o botão de opções à esquerda do cluster para o qual você deseja modificar a notificação.
3. Escolha Modificar.
4. Em Modify Cluster, em Topic for SNS Notification, escolha Disable Notifications.
5. Escolha Modificar.

Ativando e desativando as notificações do Amazon SNS (CLI)AWS

Para desabilitar notificações do Amazon SNS, use o comando `update-cluster` com os seguintes parâmetros:

Para Linux, macOS ou Unix:

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --sns-topic-status inactive
```

Para Windows:

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --sns-topic-status inactive
```

Habilitação e desabilitação de notificações do Amazon SNS (API do MemoryDB)

Para desabilitar notificações do Amazon SNS, chame a ação `UpdateCluster` com os seguintes parâmetros:

- `ClusterName=my-cluster`
- `SnsTopicStatus=inactive`

Essa chamada retorna uma saída semelhante à seguinte:

Example

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateCluster
    &ClusterName=my-cluster
    &SnsTopicStatus=inactive
    &Version=2021-01-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20210801T220302Z
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
```

```
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Visualizar eventos do MemoryDB

O MemoryDB faz o evento de logs relacionado aos seus clusters, grupos de segurança e grupos de parâmetros. Essas informações incluem a data e a hora do evento, o nome da origem e o tipo de origem do evento, bem como uma descrição do evento. Você pode facilmente recuperar eventos do registro usando o console MemoryDB, o AWS CLI `describe-events` comando ou a ação da API MemoryDB. `DescribeEvents`

Os procedimentos a seguir mostram como visualizar todos os eventos do MemoryDB das últimas 24 horas (1440 minutos).

Visualização de eventos do MemoryDB (console)

O procedimento a seguir exibe eventos usando o console do MemoryDB.

Para visualizar eventos usando o console do MemoryDB

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, escolha Events.

   A tela Eventos exibe a listagem de todos os eventos disponíveis. Cada linha da lista representa um evento e exibe a origem do evento, o tipo de evento (como cluster, grupo de parâmetros, acl, grupo de segurança ou grupo de sub-rede), a hora GMT do evento e a descrição do evento.

   Usando a opção Filtro, você pode especificar se deseja ver todos os eventos ou apenas eventos de um tipo específico na lista de eventos.

Visualizando eventos do MemoryDB (CLI AWS)

Para gerar uma lista de eventos do MemoryDB usando o AWS CLI, use o comando. `describe-events` Você pode usar parâmetros opcionais para controlar os tipos de eventos listados, o período de tempo dos eventos listados, o número máximo de eventos a serem listados e muito mais.

O código a seguir lista até 40 eventos de cluster.

```
aws memorydb describe-events --source-type cluster --max-results 40
```

O código a seguir lista todos os eventos nas últimas 24 horas (1440 minutos).

```
aws memorydb describe-events --duration 1440
```

A saída do comando `describe-events` é semelhante a esta.

```
{
    "Events": [
        {
            "Date": "2021-03-29T22:17:37.781Z",
            "Message": "Added node 0001 in Availability Zone us-east-1a",
            "SourceName": "memorydb01",
            "SourceType": "cluster"
        },
        {
            "Date": "2021-03-29T22:17:37.769Z",
            "Message": "cluster created",
            "SourceName": "memorydb01",
            "SourceType": "cluster"
        }
    ]
}
```

Para obter mais informações, como os parâmetros disponíveis e os valores de parâmetros permitidos, consulte `describe-events`.

Visualizando eventos do MemoryDB (API do MemoryDB)

Para gerar uma lista de eventos do MemoryDB usando a API do MemoryDB, use a ação `DescribeEvents`. Você pode usar parâmetros opcionais para controlar os tipos de eventos listados, o período de tempo dos eventos listados, o número máximo de eventos a serem listados e muito mais.

O código a seguir lista os 40 eventos de cluster mais recentes.

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeEvents
   &MaxResults=40
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &SourceType=cluster
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
```

```
   &X-Amz-Credential=<credential>
```

O código a seguir lista os eventos de cluster nas últimas 24 horas (1440 minutos).

```
https://memory-db.us-east-1.amazonaws.com/
   ?Action=DescribeEvents
   &Duration=1440
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &SourceType=cluster
   &Timestamp=20210802T192317Z
   &Version=2021-01-01
   &X-Amz-Credential=<credential>
```

As ações acima devem produzir uma saída semelhante à seguinte.

```
<DescribeEventsResponse xmlns="http://memory-db.us-east-1.amazonaws.com/
doc/2021-01-01/">
    <DescribeEventsResult>
        <Events>
            <Event>
                <Message>cluster created</Message>
                <SourceType>cluster</SourceType>
                <Date>2021-08-02T18:22:18.202Z</Date>
                <SourceName>my-memorydb-primary</SourceName>
            </Event>

 (...output omitted...)

        </Events>
    </DescribeEventsResult>
    <ResponseMetadata>
        <RequestId>e21c81b4-b9cd-11e3-8a16-7978bb24ffdf</RequestId>
    </ResponseMetadata>
</DescribeEventsResponse>
```

Para obter mais informações, como os parâmetros disponíveis e os valores de parâmetros permitidos, consulte `DescribeEvents`.

Notificações de eventos e o Amazon SNS

O MemoryDB pode publicar mensagens usando o Serviço de notificação simples da Amazon (Amazon Simple Notification Service (SNS)) quando houver eventos significativos em um cluster. Esse atributo pode ser usado para atualizar as listas de servidores em máquinas clientes conectadas a endpoints de nó individuais de um cluster.

> **Note**
>
> Para obter mais informações sobre o Amazon Simple Notification Service (SNS), incluindo informações sobre preços e links para a documentação do Amazon SNS, consulte a página do produto Amazon SNS.

As notificações são publicadas em um tópico do Amazon SNS especificado. Os seguintes são requisitos para notificações:

- Apenas um tópico pode ser configurado para notificações do MemoryDB.
- A conta da AWS que possui o tópico do Amazon SNS deve ser a mesma conta que possui o cluster em que as notificações estão habilitadas.

Eventos do MemoryDB

Os seguintes eventos do MemoryDB acionam notificações do Amazon SNS:

Nome do evento	Message	Descrição
MemoryDB:AddNodeComplete	`"Modified number of nodes from %d to %d"`	Um nó foi adicionado ao cluster e está pronto para uso.
MemoryDB:AddNodeFailed devido a endereços IP livres insuficientes	`"Failed to modify number of nodes from %d to %d due to insufficient free IP addresses"`	Um nó não pôde ser adicionado porque não há endereços IP suficientes disponíveis.
MemoryDB:ClusterParametersChanged	`"Updated parameter group for the cluster"`	Um ou mais parâmetros de cluster foram alterados.

Nome do evento	Message	Descrição
	Em caso de criar, envie também `"Updated to use a ParameterGroup %s"`	
MemoryDB:ClusterProvisioningComplete	`"Cluster created."`	O provisionamento de um cluster está concluído, e os nós no cluster estão prontos para uso.
MemoryDB:ClusterProvisioningFailed devido a estado de rede incompatível	`"Failed to create cluster due to incompatible network state. %s"`	Foi feita uma tentativa de executar um novo cluster em uma nuvem privada virtual (VPC) inexistente.
MemoryDB:ClusterRestoreFailed	`"Restore from %s failed for node %s. %s"`	O MemoryDB não conseguiu preencher o cluster com os dados do snapshot do Redis. Isso pode ser devido a um arquivo de snapshot inexistente no Amazon S3 ou permissões incorretas nesse arquivo. Se você descrever o cluster, o status será `restore-failed` . Você precisará excluir o cluster e começar de novo. Para obter mais informações, consulte Propagação de um novo cluster com um snapshot criado externamente.
MemoryDB:ClusterScalingComplete	`"Succeeded applying modification to node type to %s."`	Aumento vertical da escala para cluster concluído com sucesso.

Nome do evento	Message	Descrição
MemoryDB:ClusterScalingFailed	`"Failed applying modification to node type to %s."`	A operação de aumentar a escala verticalmente no cluster falhou.
MemoryDB:ClusterSecurityGroupModified	`"Modified security group for cluster."`	Um dos seguintes eventos ocorreu: • A lista de grupos de segurança autorizados para o cluster foi modificada. • Um ou mais novos grupos de segurança do EC2 foram autorizados em qualquer um dos grupos de segurança associados ao cluster. • Um ou mais grupos de segurança do EC2 foram revogados de qualquer um dos grupos de segurança associados ao cluster.

Nome do evento	Message	Descrição
MemoryDB:NodeReplaceStarted	`"Recovering node %s"`	O MemoryDB detectou que o host que está executando um nó está degradado ou inacessível e iniciou a substituição do nó. **Note** A entrada de DNS para o nó substituído não é alterada. Na maioria dos casos, você não precisa atualizar a lista de servidores para seus clientes quando esse evento ocorre. No entanto, algumas bibliotecas de clientes podem parar de usar o nó mesmo após o MemoryDB ter substituído o nó. Neste caso, o aplicativo deve atualizar a lista de servidores quando esse evento ocorrer.

Nome do evento	Message	Descrição
MemoryDB:NodeReplaceComplete	`"Finished recovery for node %s"`	O MemoryDB detectou que o host que executa um nó está degradado ou inacessível e concluiu a substituição do nó. **Note** A entrada de DNS para o nó substituído não é alterada. Na maioria dos casos, você não precisa atualizar a lista de servidores para seus clientes quando esse evento ocorre. No entanto, algumas bibliotecas de clientes podem parar de usar o nó mesmo após o MemoryDB ter substituído o nó. Neste caso, o aplicativo deve atualizar a lista de servidores quando esse evento ocorrer.
MemoryDB:CreateClusterComplete	`"Cluster created"`	O cluster foi criado com sucesso.
MemoryDB:CreateClusterFailed	`"Failed to create cluster due to unsuccessful creation of its node(s)."` e `"Deleting all nodes belonging to this cluster."`	O cluster não foi criado.

Nome do evento	Message	Descrição
MemoryDB:DeleteClusterComplete	`"Cluster deleted."`	A exclusão de um cluster e todos os nós associados foi concluída.
MemoryDB:FailoverComplete	`"Failover to replica node %s completed"`	O failover para um nó de réplica foi bem-sucedido.
MemoryDB:NodeReplacementCanceled	`"The replacement of node %s which was scheduled during the maintenance window from start time: %s, end time: %s has been canceled"`	Um nó no seu cluster que estava programado para substituição já não está programado para substituição.
MemoryDB:NodeReplacementRescheduled	`"The replacement in maintenance window for node %s has been re-scheduled from previous start time: %s, previous end time: %s to new start time: %s, new end time: %s"`	Um nó no seu cluster previamente programado para substituição foi reprogramado para substituição durante a nova janela descrita na notificação. Para obter informações sobre quais ações você pode realizar, consulte Substituição de nós.
MemoryDB:NodeReplacementScheduled	`"The node %s is scheduled for replacement during the maintenance window from start time: %s to end time: %s"`	Um nó no seu cluster está programado para substituição durante a janela descrita na notificação. Para obter informações sobre quais ações você pode realizar, consulte Substituição de nós.

Nome do evento	Message	Descrição
MemoryDB:RemoveNodeComplete	`"Removed node %s"`	Um nó foi removido do cluster.
MemoryDB:SnapshotComplete	`"Snapshot %s succeeded for node %s"`	Um snapshot foi concluído com sucesso.
MemoryDB:SnapshotFailed	`"Snapshot %s failed for node %s"`	O snapshot falhou. Consulte os eventos do cluster para obter mais detalhes sobre a causa. Se você descrever o snapshot, consulte DescribeSnapshots, o status será `failed`.

Registrando chamadas da API do MemoryDB para Redis com AWS CloudTrail

O MemoryDB para Redis é integrado ao AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou serviço da AWS no MemoryDB para Redis. O CloudTrail captura todas as chamadas de API do MemoryDB para Redis como eventos, incluindo chamadas do console do MemoryDB para Redis e de chamadas de código para as operações da API do MemoryDB para Redis. Se você criar uma trilha, poderá ativar o fornecimento contínuo de eventos do CloudTrail para um bucket do Amazon S3, incluindo eventos do MemoryDB para Redis. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail em Histórico de eventos. Com as informações coletadas pelo CloudTrail, determine a solicitação feita para os planos do MemoryDB para Redis, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e os detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Informações do MemoryDB para Redis no CloudTrail

O CloudTrail é habilitado em sua conta da AWS quando ela é criada. Quando ocorre uma atividade no MemoryDB para Redis, essa atividade é registrada em um evento CloudTrail junto com outros

eventos de serviço da AWS no Histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua conta da AWS. Para obter mais informações, consulte Como visualizar eventos com o histórico de eventos do CloudTrail.

Para um registro contínuo de eventos em sua conta da AWS, incluindo eventos para o MemoryDB para Redis, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as regiões. A trilha registra em log eventos de todas as regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. Além disso, é possível configurar outros serviços da AWS para analisar mais ainda mais e agir com base nos dados de eventos coletados nos logs do CloudTrail. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- Serviços e integrações compatíveis com o CloudTrail
- Configurar notificações do Amazon SNS para o CloudTrail
- Receber arquivos de log do CloudTrail de várias regiões e receber arquivos de log do CloudTrail de várias contas

Todas as ações do MemoryDB para Redis são registradas em log pelo CloudTrail. Por exemplo, as chamadas para as ações `CreateCluster`, `DescribeClusters` e `UpdateCluster` geram entradas nos arquivos de log do CloudTrail.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário raiz ou do usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro serviço da AWS.

Para obter mais informações, consulte o Elemento userIdentity do CloudTrail.

Compreendendo as entradas do arquivo de log do MemoryDB para Redis

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket do Amazon S3 especificado. Os arquivos de log do CloudTrail contêm uma ou mais entradas de log.

Um evento representa uma única solicitação de qualquer origem e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros de solicitação e assim por diante. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada das chamadas de API pública. Dessa forma, eles não são exibidos em uma ordem específica.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a ação `CreateCluster`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EKIAUAXQT3SWDEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/john",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "john"
    },
    "eventTime": "2021-07-10T17:56:46Z",
    "eventSource": "memorydb.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.01",
    "userAgent": "aws-cli/2.2.29 Python/3.9.6 Darwin/19.6.0 source/x86_64 prompt/off
 command/memorydb.create-cluster",
    "requestParameters": {
        "clusterName": "memorydb-cluster",
        "nodeType": "db.r6g.large",
        "subnetGroupName": "memorydb-subnet-group",
        "aCLName": "open-access"
    },
    "responseElements": {
        "cluster": {
            "name": "memorydb-cluster",
            "status": "creating",
            "numberOfShards": 1,
            "availabilityMode": "MultiAZ",
            "clusterEndpoint": {
                "port": 6379
            },
            "nodeType": "db.r6g.large",
            "engineVersion": "6.2",
            "enginePatchVersion": "6.2.6",
```

```
            "parameterGroupName": "default.memorydb-redis6",
            "parameterGroupStatus": "in-sync",
            "subnetGroupName": "memorydb-subnet-group",
            "tLSEnabled": true,
            "aRN": "arn:aws:memorydb:us-east-1:123456789012:cluster/memorydb-cluster",
            "snapshotRetentionLimit": 0,
            "maintenanceWindow": "tue:06:30-tue:07:30",
            "snapshotWindow": "09:00-10:00",
            "aCLName": "open-access",
            "dataTiering": "false",
            "autoMinorVersionUpgrade": true
        }
    },
    "requestID": "506fc951-9ae2-42bb-872c-98028dc8ed11",
    "eventID": "2ecf3dc3-c931-4df0-a2b3-be90b596697e",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a ação `DescribeClusters`. Observe que, para todas as chamadas Descrever e Listar do MemoryDB para Redis (`Describe*` e `List*`), a seção `responseElements` é removida e aparece como `null`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EKIAUAXQT3SWDEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/john",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "john"
    },
    "eventTime": "2021-07-10T18:39:51Z",
    "eventSource": "memorydb.amazonaws.com",
    "eventName": "DescribeClusters",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.01",
    "userAgent": "aws-cli/2.2.29 Python/3.9.6 Darwin/19.6.0 source/x86_64 prompt/off
 command/memorydb.describe-clusters",
```

```
    "requestParameters": {
        "maxResults": 50,
        "showShardDetails": true
    },
    "responseElements": null,
    "requestID": "5e831993-52bb-494d-9bba-338a117c2389",
    "eventID": "32a3dc0a-31c8-4218-b889-1a6310b7dd50",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

O exemplo a seguir mostra uma entrada de log do CloudTrail que registra uma ação `UpdateCluster`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EKIAUAXQT3SWDEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/john",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "john"
    },
    "eventTime": "2021-07-10T19:23:20Z",
    "eventSource": "memorydb.amazonaws.com",
    "eventName": "UpdateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.01",
    "userAgent": "aws-cli/2.2.29 Python/3.9.6 Darwin/19.6.0 source/x86_64 prompt/off
 command/memorydb.update-cluster",
    "requestParameters": {
        "clusterName": "memorydb-cluster",
        "snapshotWindow": "04:00-05:00",
        "shardConfiguration": {
            "shardCount": 2
        }
    },
    "responseElements": {
        "cluster": {
```

```
            "name": "memorydb-cluster",
            "status": "updating",
            "numberOfShards": 2,
            "availabilityMode": "MultiAZ",
            "clusterEndpoint": {
                "address": "clustercfg.memorydb-cluster.cde8da.memorydb.us-
east-1.amazonaws.com",
                "port": 6379
            },
            "nodeType": "db.r6g.large",
            "engineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "parameterGroupName": "default.memorydb-redis6",
            "parameterGroupStatus": "in-sync",
            "subnetGroupName": "memorydb-subnet-group",
            "tLSEnabled": true,
            "aRN": "arn:aws:memorydb:us-east-1:123456789012:cluster/memorydb-cluster",
            "snapshotRetentionLimit": 0,
            "maintenanceWindow": "tue:06:30-tue:07:30",
            "snapshotWindow": "04:00-05:00",
            "autoMinorVersionUpgrade": true,
            "DataTiering": "false"
        }
    },
    "requestID": "dad021ce-d161-4365-8085-574133afab54",
    "eventID": "e0120f85-ab7e-4ad4-ae78-43ba15dee3d8",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a ação `CreateUser`. Observe que, para chamadas do MemoryDB para Redis que contêm dados confidenciais, esses dados serão editados no evento correspondente do CloudTrail, conforme mostrado na seção `requestParameters` abaixo.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EKIAUAXQT3SWDEXAMPLE",
```

```
        "arn": "arn:aws:iam::123456789012:user/john",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "john"
    },
    "eventTime": "2021-07-10T19:56:13Z",
    "eventSource": "memorydb.amazonaws.com",
    "eventName": "CreateUser",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.01",
    "userAgent": "aws-cli/2.2.29 Python/3.9.6 Darwin/19.6.0 source/x86_64 prompt/off
 command/memorydb.create-user",
    "requestParameters": {
        "userName": "memorydb-user",
        "authenticationMode": {
            "type": "password",
            "passwords": [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "accessString": "~* &* -@all +@read"
    },
    "responseElements": {
        "user": {
            "name": "memorydb-user",
            "status": "active",
            "accessString": "off ~* &* -@all +@read",
            "aCLNames": [],
            "minimumEngineVersion": "6.2",
            "authentication": {
                "type": "password",
                "passwordCount": 1
            },
            "aRN": "arn:aws:memorydb:us-east-1:123456789012:user/memorydb-user"
        }
    },
    "requestID": "ae288b5e-80ab-4ff8-989a-5ee5c67cd193",
    "eventID": "ed096e3e-16f1-4a23-866c-0baa6ec769f6",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
```

```
}
```

Validação de conformidade do MemoryDB para Redis

Auditores terceirizados avaliam a segurança e a conformidade do MemoryDB para Redis como parte de vários programas de conformidade da AWS. Isso inclui:

- Payment Card Industry Data Security Standard (PCI DSS – Padrão de segurança de dados do setor de cartão de pagamento) Para obter mais informações, consulte PCI DSS.
- Acordo de associado comercial da Lei de Portabilidade e Responsabilidade de Provedores de Saúde (HIPAA BAA) Para obter mais informações, consulte Conformidade com a HIPAA.
- System and Organization Controls (SOC – Controles do Sistema e da Organização) 1, 2 e 3. Para obter mais informações, consulte SOC.
- Programa Federal de Gerenciamento de Riscos e Autorizações (Federal Risk and Authorization Management Program, FedRAMP) Moderado. Para obter mais informações, consulte FedRAMP.
- ISO/IEC 27001:2013, 27017:2015, 27018:2019, and ISO/IEC 9001:2015. Para obter mais informações, consulte as certificações e os serviços ISO e CSA STAR da AWS.

Para obter uma lista dos produtos da AWS no escopo de programas de conformidade específicos, consulte Produtos da AWS no escopo por programa de conformidade.

É possível fazer download de relatórios de auditoria de terceiros usando o AWS Artifact. Para obter mais informações, consulte Fazer download de relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar o MemoryDB é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentos aplicáveis. A AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e compatibilidade – Esses guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- Recursos de conformidade da AWS: essa coleção de manuais e guias pode ser aplicada a seu setor e local.
- Avaliação de recursos com regras no AWS ConfigGuia do desenvolvedor – AWS Configavalia a conformidade das configurações de seus recursos com práticas internas, diretrizes do setor e regulamentos.

- AWS Security Hub: esse serviço da AWS fornece uma visão abrangente do estado de sua segurança na AWS que ajuda você a conferir sua conformidade com padrões e práticas recomendadas de segurança do setor.
- Audit Manager da AWS: esse serviço da AWS ajuda a auditar continuamente o uso da AWS para simplificar a forma como você gerencia os riscos e a conformidade com regulamentos e padrões do setor.

Segurança da infraestrutura no Amazon MemoryDB para Redis

Como um serviço gerenciado, o MemoryDB é protegido pelos procedimentos de segurança de rede global da AWS descritos no whitepaper Amazon Web Services: Overview of Security Processes.

Você usa chamadas de API publicadas na AWS para acessar o MemoryDB por meio da rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.2 ou posterior. Recomendamos usar o TLS 1.3 ou posterior. Os clientes também devem ter suporte a conjuntos de criptografia com perfect forward secrecy (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE). A maioria dos sistemas modernos como Java 7 e versões posteriores oferece suporte a esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Privacidade do tráfego entre redes

O MemoryDB para Redis usa as seguintes técnicas para guardar seus dados e protegê-los contra o acesso não autorizado:

- MemoryDB e Amazon VPC explica o tipo de grupo de segurança de que você precisa para sua instalação.
- API do MemoryDB para Redis e interface de endpoints da VPC (AWS PrivateLink)permite estabelecer uma conexão privada entre a VPC e os endpoints da API do MemoryDB for Redis.
- Gerenciamento de identidade e acesso no MemoryDB para Redis para conceder e limitar ações de usuários, grupos e funções.

MemoryDB e Amazon VPC

O serviço da Amazon Virtual Private Cloud (Amazon VPC) define uma rede virtual que lembra muito um datacenter tradicional. Ao configurar uma nuvem privada virtual (VPC) com a Amazon VPC, você pode selecionar seu intervalo de endereços IP, criar sub-redes e configurar tabelas de rotas, gateways de rede e configurações de segurança. Você também pode adicionar um cluster à rede virtual e controlar o acesso ao cluster usando os grupos de segurança da Amazon VPC.

Esta seção explica como configurar manualmente um cluster do MemoryDB em uma VPC. Essas informações destinam-se a usuários que desejam ter uma compreensão mais profunda de como o MemoryDB e a Amazon VPC funcionam juntos.

Tópicos

Entendendo o MemoryDB e as VPCs

O MemoryDB é totalmente integrado à Amazon VPC. Para usuários do MemoryDB, isso significa o seguinte:

- O MemoryDB sempre inicia seu cluster em uma VPC.
- Se você for novato na AWS, uma VPC padrão será criada automaticamente para você.
- Se você tiver uma VPC padrão e não especificar uma sub-rede quando executar um cluster, este será iniciado na sua Amazon VPC padrão.

Para mais informações, consulte Detecção de suas plataformas compatíveis e se você tem um VPC padrão.

Com a Amazon VPC, você pode criar uma rede virtual na nuvem da AWS que se assemelha muito com um datacenter tradicional. É possível configurar sua VPC, incluindo selecionar o intervalo de endereços IP, criar sub-redes e definir tabelas de rotas, gateways de rede e configurações de segurança.

O MemoryDB gerencia atualizações de software, patches, detecção de falhas e recuperação.

Visão geral do MemoryDB em uma VPC

1	A VPC é uma parte isolada da nuvem da AWS que recebe seu próprio bloco de endereços IP.
2	Um gateway da Internet conecta sua VPC diretamente à Internet e fornece acesso a outros recursos da AWS, como o Amazon Simple Storage Service (Amazon S3), que estão em execução fora da sua VPC.
3	Uma sub-rede do Amazon VPC é um segmento do intervalo de endereços IP de um VPC em que você pode isolar recursos da AWS de acordo com suas necessidades operacionais e de segurança.
4	Uma tabela de rotas em sua VPC direciona o tráfego de rede entre a sub-rede e a Internet. A Amazon VPC tem um roteador implícito.

5	Um grupo de segurança do Amazon VPC controla o tráfego de entrada e saída de seus clusters do MemoryDB e instâncias do Amazon EC2.
6	Você pode ativar um cluster do MemoryDB na sub-rede. Os nós possuem endereços IP privados a partir do intervalo de endereços da sub-rede.
7	Você também pode ativar instâncias do Amazon EC2 na sub-rede. Cada instância do Amazon EC2 tem um endereço IP privado do intervalo de endereços da sub-rede. A instância do Amazon EC2 pode se conectar a qualquer nó na mesma sub-rede.
8	Para que uma instância do Amazon EC2 em sua VPC possa ser acessada pela Internet, você precisa atribuir um endereço público estático chamado endereço Elastic IP à instância .

Pré-requisitos

Para criar um cluster do MemoryDB em uma VPC, sua VPC deve atender aos seguintes requisitos:

- A VPC deve permitir instâncias do Amazon EC2 não dedicadas. Você não pode usar o MemoryDB em uma VPC que está configurada para a locação de instâncias dedicadas.
- Um grupo de sub-rede de cache deve ser definido para a sua VPC. O MemoryDB utiliza esse grupo de sub-redes para selecionar uma sub-rede e endereços IP nessa sub-rede para associar aos seus nós.
- Um grupo de segurança deve ser definido para a sua VPC, ou você pode usar o padrão fornecido.
- Os blocos CIDR para cada sub-rede devem ser suficientemente grandes para fornecer endereços IP de reposição para o MemoryDB usar durante atividades de manutenção.

Roteamento e segurança

Você pode configurar o roteamento na sua VPC para controlar para onde o tráfego flui (por exemplo, para o gateway da Internet ou o gateway privado virtual). Com um gateway da Internet, sua VPC tem acesso direto a outros recursos da AWS que não estão sendo executados na sua VPC. Se você optar por ter apenas um gateway virtual privado com uma conexão com a rede local da sua organização, poderá rotear seu tráfego vinculado à Internet através da VPN e usar políticas de

segurança locais e um firewall para controlar a saída. Nesse caso, você está sujeito a cobranças adicionais de largura de banda ao acessar os recursos da AWS pela Internet.

Você pode usar grupos de segurança da Amazon VPC para ajudar a proteger os clusters do MemoryDB e as instâncias do Amazon EC2 na sua Amazon VPC. Os security groups atuam como um firewall no nível da instância e não no nível da sub-rede.

> **Note**
>
> Recomendamos enfaticamente que você use nomes DNS para se conectar aos seus nós, pois o endereço IP subjacente pode mudar com o tempo.

Documentação da Amazon VPC

A Amazon VPC tem seu próprio conjunto de documentação para descrever como criar e usar sua Amazon VPC. A tabela a seguir mostra onde encontrar informações nos guias sobre a Amazon VPC.

Descrição	Documentação
Como começar a usar a Amazon VPC	Conceitos básicos do Amazon VPC
Como usar a Amazon VPC por meio do AWS Management Console	Guia do usuário da Amazon VPC
Descrições completas de todos os comandos da Amazon VPC	Referência da linha de comando do Amazon EC2 (os comandos da Amazon VPC são encontrados na referência do Amazon EC2)
Descrições completas das operações da API da Amazon VPC, tipos de dados e erros da Amazon VPC	Referência da API do Amazon EC2 (as operações da Amazon VPC são encontrados na referência do Amazon EC2)
Informações para o administrador da rede que precisa configurar o gateway no final de uma conexão VPN IPsec opcional	O que é a AWS VPN Site-to-Site?

Para obter informações mais detalhadas sobre a Amazon Virtual Private Cloud, consulte Amazon Virtual Private Cloud.

Padrões de acesso para acessar um cluster do MemoryDB em uma Amazon VPC

O MemoryDB para Redis oferece suporte para os seguintes cenários para acessar um cluster em uma Amazon VPC:

Sumário

Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão na mesma Amazon VPC

O caso de uso mais comum é quando uma aplicação implantada em uma instância do EC2 precisa se conectar a um cluster na mesma VPC.

A maneira mais simples de gerenciar o acesso entre instâncias do EC2 e clusters na mesma VPC é fazer o seguinte:

1. Crie um security group de VPC para o seu cluster. Esse grupo de segurança pode ser usado para restringir o acesso aos clusters. Por exemplo, é possível criar uma regra personalizada para esse security group que permite o acesso TCP usando a porta atribuída ao cluster quando você o criou e um endereço IP que será usado para acessar o cluster.

   A porta padrão dos clusters do MemoryDB é 6379.

2. Crie um security group de VPC para suas instâncias do EC2 (servidores Web e de aplicativos). Esse grupo de segurança pode, se necessário, permitir o acesso à instância do EC2 da Internet através da tabela de roteamento da VPC. Por exemplo, você pode definir regras nesse grupo de segurança para permitir o acesso TCP à instância do EC2 pela porta 22.
3. Crie regras personalizadas no grupo de segurança para o seu cluster que permitam conexões do grupo de segurança que você criou para suas instâncias do EC2. Isso permitiria que qualquer membro de grupo de segurança acessasse os clusters.

Para criar uma regra em um security group de VPC que permita conexões de outro security group

1. Faça login no Console de Gerenciamento da AWS e abra o console da Amazon VPC em https://console.aws.amazon.com/vpc.
2. No painel de navegação esquerdo, escolha Security Groups.
3. Selecione ou crie um grupo de segurança que você usará para seus clusters. Em Regras de entrada, selecione Editar regras de entrada e escolha Adicionar regra. Esse security group permitirá o acesso a membros de outro security group.
4. Em Type, escolha Custom TCP Rule.
    a. Para Port Range, especifique a porta que você usou quando criou seu cluster.

        A porta padrão dos clusters do MemoryDB é 6379.
    b. Na caixa Source, comece a digitar o ID do security group. Na lista, selecione o grupo de segurança que você usará para o suas instâncias do Amazon EC2.
5. Escolha Save quando terminar.

Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão em diferentes Amazon VPCs

Quando seu cluster está em uma VPC diferente da instância do EC2 que você está usando para acessá-lo, existem várias maneiras de acessar o cluster. Se o cluster e a instância do EC2 estiverem em VPCs diferentes, mas na mesma região, você poderá usar o emparelhamento de VPCs. Se o cluster e a instância do EC2 estiverem em regiões diferentes, você poderá criar conectividade via VPN entre regiões.

Tópicos

- Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão em diferentes Amazon VPCs na mesma região
- Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão em diferentes Amazon VPCs em regiões diferentes

Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão em diferentes Amazon VPCs na mesma região

Cluster acessado por uma instância do Amazon EC2 em uma Amazon VPC diferente na mesma região - conexão de emparelhamento de VPCs

Uma conexão de emparelhamento da VPC é uma conexão de redes entre duas VPCs que permite direcionar o tráfego entre elas usando endereços IP privados. Instâncias em qualquer VPC podem se comunicar umas com as outras como se estivessem na mesma rede. Você pode criar uma conexão de emparelhamento de VPCs entre suas próprias Amazon VPCs ou com uma Amazon VPC em outra conta da AWS em uma única região. Para saber mais sobre o emparelhamento de Amazon VPCs, consulte a documentação da VPC.

Para acessar um cluster em uma Amazon VPC diferente por emparelhamento

1. Certifique-se de que as duas VPCs não tenham um intervalo de IP sobreposto, ou você não poderá compará-las.
2. Emparelhe as duas VPCs. Para obter mais informações, consulte Criação e aceitação de uma conexão de emparelhamento da Amazon VPC.
3. Atualize sua tabela de roteamento. Para obter mais informações, consulte Atualizar as tabelas de rotas para uma conexão de emparelhamento de VPC
4. Modifique o grupo de segurança do cluster do MemoryDB para permitir a conexão de entrada do grupo de segurança do aplicativo na VPC emparelhada. Para obter mais informações, consulte a Referência para security groups de VPC de emparelhamento.

O acesso a um cluster por meio de uma conexão de emparelhamento implicará custos adicionais de transferência de dados.

Uso do Transit Gateway

Um Transit Gateway permite anexar VPCs e conexões VPN na mesma região da AWS e rotear tráfego entre elas. Um Transit Gateway funciona em contas da AWS, e você pode usar o Resource Access Manager da AWS para compartilhar o Transit Gateway com outras contas. Depois de compartilhar um gateway de trânsito com outra conta da AWS, o proprietário da conta poderá anexar as VPCs dele ao gateway de trânsito. Um usuário de qualquer uma das contas pode excluir o anexo a qualquer momento.

É possível ativar o multicast em um gateway de trânsito e, depois, criar um domínio de multicast do gateway de trânsito que permita ao tráfego de multicast ser enviado da origem de multicast para membros do grupo de multicast em anexos da VPC associados ao domínio.

Também é possível criar um anexo da conexão de emparelhamento entre gateways de trânsito em diferentes regiões da AWS. Isso permite que você roteie o tráfego entre os anexos dos gateways de trânsito em regiões diferentes.

Para obter mais informações, consulte Gateways de trânsito.

Acesso a um cluster do MemoryDB quando ele e a instância do Amazon EC2 estão em diferentes Amazon VPCs em regiões diferentes

Uso da VPC de trânsito

Uma alternativa ao uso do emparelhamento de VPC, outra estratégia comum para conectar várias VPCs geograficamente dispersas e redes remotas é criar uma VPC de trânsito que serve como um centro de trânsito de rede global. Uma VPC de trânsito simplifica o gerenciamento da rede e minimiza o número de conexões necessárias para conectar várias VPCs e redes remotas. Esse design pode economizar tempo e esforços e também reduzir custos, uma vez que é implementado praticamente sem as despesas tradicionais de estabelecer uma presença física em um hub de trânsito de colocação ou implantar equipamentos de rede física.

Conexão entre diferentes VPCs em regiões distintas

Depois de estabelecida a Amazon VPC de trânsito, um aplicativo implantado em uma VPC “spoke” em uma região pode se conectar a um cluster do MemoryDB em uma VPC “spoke” de outra região.

Para acessar um cluster em um VPC diferente dentro de uma região da AWS diferente

1. Implante uma solução de VPC de trânsito. Para obter mais informações, consulte Transit Gateway da AWS.

2. Atualize as tabelas de rotas VPC no aplicativo e as VPCs para rotear o tráfego por meio do Gateway privado virtual (Virtual Private Gateway, VGW) e do dispositivo VPN. No caso do Roteamento dinâmico com o protocolo BGP, suas rotas podem ser propagadas automaticamente.
3. Modifique o grupo de segurança do seu cluster do MemoryDB para permitir a conexão de entrada do intervalo IP de instâncias do aplicativo. Observe que você não poderá fazer referência ao security group do servidor de aplicativos nesse cenário.

O acesso a um cluster entre regiões introduzirá latências de rede e custos adicionais de transferência de dados entre regiões.

Acessar um cluster do MemoryDB a partir de um aplicativo executado no datacenter de um cliente

Outro cenário possível é uma arquitetura híbrida em que clientes ou aplicativos no datacenter do cliente podem precisar acessar um cluster do MemoryDB na VPC. Esse cenário também tem suporte, desde que haja conectividade entre a VPC dos clientes e o datacenter via VPN ou Direct Connect.

Tópicos

- Acessar um cluster do MemoryDB a partir de um aplicativo executado no datacenter de um cliente usando conectividade de VPN
- Acessar um cluster do MemoryDB a partir de um aplicativo executado no datacenter de um cliente usando o Direct Connect

Acessar um cluster do MemoryDB a partir de um aplicativo executado no datacenter de um cliente usando conectividade de VPN

Conectar ao MemoryDB a partir do seu datacenter através de uma VPN

Para acessar um cluster em uma VPC a partir do aplicativo no local via conexão VPN

1. Estabeleça a conectividade de VPN adicionando um gateway privado virtual de hardware à sua VPC. Para obter mais informações, consulte o tópico sobre como Adicionar um gateway privado virtual de hardware à sua VPC.

2. Atualize a tabela de rotas de VPC para a sub-rede na qual seu cluster do MemoryDB está implantado para permitir o tráfego do seu servidor de aplicativos on-premises. No caso do Roteamento dinâmico com o BGP, suas rotas podem ser propagadas automaticamente.
3. Modifique o grupo de segurança do seu cluster do MemoryDB para permitir a conexão de entrada dos servidores de aplicativos on-premises.

Acessar um cluster através de uma conexão VPN introduzirá latências de rede e custos adicionais de transferência de dados.

Acessar um cluster do MemoryDB a partir de um aplicativo executado no datacenter de um cliente usando o Direct Connect

Conectar-se ao MemoryDB a partir do seu datacenter via Direct Connect

Para acessar um cluster do MemoryDB de um aplicativo executado em sua rede usando o Direct Connect

1. Estabeleça a conectividade Direct Connect. Para obter mais informações, consulte Conceitos básicos do Direct Connect da AWS.
2. Modifique o grupo de segurança do seu cluster do MemoryDB para permitir a conexão de entrada dos servidores de aplicativos on-premises.

O acesso a um cluster por meio de uma conexão DX pode introduzir latências de rede e taxas adicionais de transferência de dados.

Criar uma nuvem privada virtual (VPC)

Neste exemplo, você cria uma nuvem privada virtual (VPC) com base no serviço Amazon VPC com uma sub-rede privada para cada zona de disponibilidade.

Criação de uma VPC (console)

Para criar um cluster do MemoryDB em um Amazon Virtual Private Cloud

1. Faça login no Console de Gerenciamento da AWS e abra o console da Amazon VPC em https://console.aws.amazon.com/vpc.
2. No painel da VPC, escolha Criar VPC.
3. Em Recursos a serem criados, escolha VPC e mais.
4. Em Número de zonas de disponibilidade (ZAs), escolha o número de zonas de disponibilidade nas quais iniciar suas sub-redes.
5. Em Número de sub-redes públicas, escolha o número de sub-redes públicas que você deseja adicionar à sua VPC.
6. Em Número de sub-redes privadas, escolha o número de sub-redes públicas que você deseja adicionar à sua VPC.

   > **Tip**
   >
   > Anote os identificadores das sub-redes e indique quais são públicas e quais são privadas. Você precisará dessas informações mais tarde quando ativar seus clusters de cache e adicionar uma instância do Amazon EC2 à sua Amazon VPC.

7. Crie um grupo de segurança da Amazon VPC. Você usará esse grupo para seu cluster e sua instância do Amazon EC2.

   a. No painel de navegação esquerdo da tela do AWS Management Console, selecione Grupos de segurança.
   b. Escolha Criar grupo de segurança.
   c. Digite um nome e uma descrição do seu grupo de segurança nas caixas correspondentes. ParaVPC, escolha o identificador da sua VPC.
   d. Quando estiver satisfeito com as configurações, clique em Yes, Create.

8. Defina uma regra de entrada de rede para seu security group. Essa regra permitirá que você se conecte à sua instância do Amazon EC2 usando Secure Shell (SSH).

a. No painel de navegação esquerdo, escolha Security Groups.
b. Localize seu security group na lista e escolha-o.
c. Em Security Group, escolha a guia Inbound. Na caixa Create a new rule, escolha SSH e depois Add Rule.

   Defina os seguintes valores para a sua nova regra de entrada a fim de permitir o acesso HTTP.

   - Tipo: HTTP
   - Origem: 0.0.0.0/0

d. Defina os seguintes valores para a sua nova regra de entrada a fim de permitir o acesso HTTP.

   - Tipo: HTTP
   - Origem: 0.0.0.0/0

   Escolha Apply Rule Changes.

Agora você está pronto para criar um grupo de sub-rede e criar um cluster em sua VPC.

Sub-redes e grupos de sub-redes

Um grupo de sub-redes é um conjunto de sub-redes (normalmente privadas) que você pode designar para seus clusters em execução em um ambiente Amazon Virtual Private Cloud (VPC).

Ao criar um cluster em na Amazon VPC, você pode especificar um grupo de sub-rede ou usar o padrão fornecido. O MemoryDB usa esse grupo de sub-rede para escolher uma sub-rede e endereços IP dentro dessa sub-rede para associar aos seus nós.

Esta seção aborda como criar e aproveitar sub-redes e grupos de sub-redes para gerenciar o acesso aos recursos do MemoryDB.

Para obter mais informações sobre o uso de grupos de sub-redes em um ambiente da Amazon VPC, consulte Etapa 2: autorizar o acesso ao cluster.

IDs de AZs do MemoryDB compatíveis

Nome da região/região	IDs de AZ compatíveis		
Região Leste dos EUA (Ohio) `us-east-2`	`use2-az1, use2-az2, use2-az3`		
Região Leste dos EUA (N. da Virgínia) `us-east-1`	`use1-az2, use1-az4, use1-az6`		
Região Oeste dos EUA (Norte da Califórnia) `us-west-1`	`usw1-az1, usw1-az2, usw1-az3`		
Região Oeste dos EUA (Oregon) `us-west-2`	`usw2-az1, usw2-az2, usw2-az3`		

Nome da região/região	IDs de AZ compatíveis		
Região Canadá (Central) `ca-central-1`	`cac1-az1, cac1-az2, cac1-az4`		
Região Ásia-Pacífico (Hong Kong) `ap-east-1`	`ape1-az1, ape1-az2, ape1-az3`		
Região Ásia-Pacífico (Mumbai) `ap-south-1`	`aps1-az1, aps1-az2, aps1-az3`		
Região Ásia-Pacífico (Tóquio) `ap-northeast-1`	`apne1-az1, apne1-az2, apne1-az4`		
Região Ásia-Pacífico (Seul) `ap-northeast-2`	`apne2-az1, apne2-az2, apne2-az3`		
Região Ásia-Pacífico (Singapura) `ap-southeast-1`	`apse1-az1, apse1-az2, apse1-az3`		
Asia Pacific (Sydney) Region `ap-southeast-2`	`apse2-az1, apse2-az2, apse2-az3`		

Nome da região/região	IDs de AZ compatíveis		
Região Europa (Frankfurt) `eu-central-1`	`euc1-az1, euc1-az2, euc1-az3`		
Região Europa (Irlanda) `eu-west-1`	`euw1-az1, euw1-az2, euw1-az3`		
Região Europa (Londres) `eu-west-2`	`euw2-az1, euw2-az2, euw2-az3`		
Região Europa (Paris) `eu-west-3`	`euw3-az1, euw3-az2, euw3-az3`		
Região Europa (Estocolmo) `eu-north-1`	`eun1-az1, eun1-az2, eun1-az3`		
Região Europa (Milão) `eu-south-1`	`eus1-az1, eus1-az2, eus1-az3`		
Região América do Sul (São Paulo) `sa-east-1`	`sae1-az1, sae1-az2, sae1-az3`		
Região da China (Pequim) `cn-north-1`	`cnn1-az1, cnn1-az2`		

Nome da região/região	IDs de AZ compatíveis		
Região da China (Ningxia) `cn-northwest-1`	`cnw1-az1, cnw1-az2, cnw1-az3`		

Tópicos

- Criação de um grupo de sub-redes
- Criação de um grupo de sub-redes
- Visualização de detalhes do grupo de sub-redes
- Exclusão de um grupo de sub-redes

Criação de um grupo de sub-redes

Quando você criar um novo grupo de sub-rede, observe o número de endereços IP disponíveis. Se a sub-rede tiver muito poucos endereços IP livres, talvez haja um limite no que diz respeito ao número de nós adicionais que é possível acrescentar ao cluster. Para resolver esse problema, você pode atribuir uma ou mais sub-redes a um grupo de sub-redes para ter um número suficiente de endereços IP na zona de disponibilidade do seu cluster. Depois disso, você pode adicionar mais nós ao seu cluster.

Os procedimentos a seguir mostram como criar um grupo de sub-redes chamado `mysubnetgroup` (console) AWS CLI, o e a API MemoryDB.

Criação de um grupo de sub-redes (console)

O procedimento a seguir mostra como criar um grupo de sub-redes (console).

Como criar um grupo de sub-redes (console)

1. Faça login no AWS Management Console e abra o console MemoryDB em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, escolha Subnet Groups.
3. Selecione Criar grupo de sub-redes.
4. Na página Criar grupo de sub-redes, faça o seguinte:
   a. Na caixa Nome, digite um nome para o seu grupo de sub-redes.

      As restrições de nomenclatura de cluster são as seguintes:

      - Devem conter 1 a 40 caracteres alfanuméricos ou hifens.
      - Deve começar com uma letra.
      - Não podem conter dois hifens consecutivos.
      - Não podem terminar com um hífen.

   b. Na caixa Descrição, digite uma descrição para seu grupo de sub-redes.
   c. Na caixa VPC ID (ID da VPC_, escolha a Amazon VPC que você criou. Se você ainda não criou uma, escolha o botão Criar VPC e siga as etapas para criar uma.
   d. Em Sub-redes selecionadas, escolha a Zona de Disponibilidade e o ID da sua sub-rede privada e, em seguida, escolha Escolher.

5. Para Tags, você pode, opcionalmente, aplicar tags para pesquisar e filtrar suas sub-redes ou monitorar seus custos. AWS
6. Quando estiver satisfeito com as configurações, escolha Criar.
7. Na mensagem de confirmação exibida, escolha Fechar.

Seu novo grupo de sub-rede aparece na lista Grupos de sub-redes do console do MemoryDB. Na parte inferior da janela, você pode escolher o grupo de sub-redes para ver detalhes, como todas as sub-redes associadas a esse grupo.

Criação de um grupo de sub-redes (AWS CLI)

No prompt de comando, use o comando `create-subnet-group` para criar um grupo de sub-redes.

Para Linux, macOS ou Unix:

```
aws memorydb create-subnet-group \
    --subnet-group-name mysubnetgroup \
    --description "Testing" \
    --subnet-ids subnet-53df9c3a
```

Para Windows:

```
aws memorydb create-subnet-group ^
    --subnet-group-name mysubnetgroup ^
    --description "Testing" ^
    --subnet-ids subnet-53df9c3a
```

Esse comando deve produzir um resultado semelhante ao seguinte:

```
    {
        "SubnetGroup": {
            "Subnets": [
                {
                    "Identifier": "subnet-53df9c3a",
                    "AvailabilityZone": {
                    "Name": "us-east-1a"
                    }
                }
            ],
            "VpcId": "vpc-3cfaef47",
```

```
            "Name": "mysubnetgroup",
            "ARN": "arn:aws:memorydb:us-east-1:012345678912:subnetgroup/
mysubnetgroup",
            "Description": "Testing"
        }
    }
```

Para obter mais informações, consulte o AWS CLI tópicocreate-subnet-group.

Criação de um grupo de sub-redes (API do MemoryDB)

Usando a API do MemoryDB, chame `CreateSubnetGroup` com os seguintes parâmetros:

- `SubnetGroupName=`*`mysubnetgroup`*
- `Description=`*`Testing`*
- `SubnetIds.member.1=`*`subnet-53df9c3a`*

Criação de um grupo de sub-redes

Você pode atualizar a descrição de um grupo de sub-rede ou modificar a lista de IDs de sub-rede associados ao grupo de sub-rede. Você não poderá excluir um ID de sub-rede de um grupo de sub-redes se um cluster estiver usando essa sub-rede atualmente.

Os procedimentos a seguir mostram como atualizar um grupo de sub-rede.

Atualização de grupos de sub-redes (console)

Como atualizar um grupo de sub-redes

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, escolha Subnet Groups.
3. Na lista de grupos de sub-redes, escolha aquele que deseja modificar.
4. Os campos Nome, VPCId e Descrição não são modificáveis.
5. Na seção sub-redes selecionadas, clique em Gerenciar para fazer alterações nas zonas de disponibilidade necessárias para as sub-redes. Para salvar suas alterações, selecione Salvar.

Atualizando grupos de sub-redes (AWS CLI)

No prompt de comando, use o comando `update-subnet-group` para modificar um grupo de sub-redes.

Para Linux, macOS ou Unix:

```
aws memorydb update-subnet-group \
    --subnet-group-name mysubnetgroup \
    --description "New description" \
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Para Windows:

```
aws memorydb update-subnet-group ^
    --subnet-group-name mysubnetgroup ^
    --description "New description" ^
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Esse comando deve produzir um resultado semelhante ao seguinte:

```
{
    "SubnetGroup": {
        "VpcId": "vpc-73cd3c17",
        "Description": "New description",
        "Subnets": [
            {
                "Identifier": "subnet-42dcf93a",
                "AvailabilityZone": {
                    "Name": "us-east-1a"
                }
            },
            {
                "Identifier": "subnet-48fc12a9",
                "AvailabilityZone": {
                    "Name": "us-east-1a"
                }
            }
        ],
        "Name": "mysubnetgroup",
        "ARN": "arn:aws:memorydb:us-east-1:012345678912:subnetgroup/mysubnetgroup",
    }
}
```

Para obter mais informações, consulte o AWS CLI tópico update-subnet-group.

Atualização de grupos de sub-redes (API do MemoryDB)

Usando a API do MemoryDB, chame `UpdateSubnetGroup` com os seguintes parâmetros:

- `SubnetGroupName=`*`mysubnetgroup`*
- Quaisquer outros parâmetros cujos valores você deseja alterar. Este exemplo usa `Description=`*`New%20description`* para alterar a descrição do grupo de sub-redes.

Example

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateSubnetGroup
    &Description=New%20description
    &SubnetGroupName=mysubnetgroup
    &SubnetIds.member.1=subnet-42df9c3a
```

```
    &SubnetIds.member.2=subnet-48fc21a9
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Credential=<credential>
    &X-Amz-Date=20141201T220302Z
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Signature=<signature>
    &X-Amz-SignedHeaders=Host
```

> **Note**
>
> Quando você criar um novo grupo de sub-rede, anote o número de endereços IP disponíveis. Se a sub-rede tiver muito poucos endereços IP livres, talvez haja um limite no que diz respeito ao número de nós adicionais que é possível acrescentar ao cluster. Para resolver esse problema, você pode atribuir uma ou mais sub-redes a um grupo de sub-redes para ter um número suficiente de endereços IP na zona de disponibilidade do seu cluster. Depois disso, você pode adicionar mais nós ao seu cluster.

Visualização de detalhes do grupo de sub-redes

Os procedimentos a seguir mostram como visualizar detalhes de um grupo de sub-redes.

Visualizando detalhes de grupos de sub-redes (console)

Visualizar detalhes de um grupo de sub-redes (console)

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, escolha Subnet Groups.
3. Na página Grupos de sub-redes, escolha o grupo de sub-redes em Nome ou digite o nome do grupo de sub-redes na barra de pesquisa.
4. Na página Grupos de sub-redes, escolha o grupo de sub-redes em Nome ou digite o nome do grupo de sub-redes na barra de pesquisa.
5. Em Configurações do grupo de sub-rede, você pode ver o nome, a descrição, o ID da VPC e o nome do recurso da Amazon (ARN) do grupo de sub-redes.

6. Em Sub-redes, você pode visualizar as zonas de disponibilidade, os IDs de sub-rede e os blocos CIDR do grupo de sub-redes
7. Em Tags, você pode ver todas as tags associadas ao grupo de sub-redes.

Visualizando detalhes de grupos de sub-redes (AWS CLI)

No prompt de comando, use o comando `describe-subnet-groups` para visualizar os detalhes de um grupo de sub-rede especificado.

Para Linux, macOS ou Unix:

```
aws memorydb describe-subnet-groups \
    --subnet-group-name mysubnetgroup
```

Para Windows:

```
aws memorydb describe-subnet-groups ^
    --subnet-group-name mysubnetgroup
```

Esse comando deve produzir um resultado semelhante ao seguinte:

```
{
  "subnetgroups": [
    {
      "Subnets": [
        {
          "Identifier": "subnet-060cae3464095de6e",
          "AvailabilityZone": {
            "Name": "us-east-1a"
          }
        },
        {
          "Identifier": "subnet-049d11d4aa78700c3",
          "AvailabilityZone": {
            "Name": "us-east-1c"
          }
        },
        {
          "Identifier": "subnet-0389d4c4157c1edb4",
          "AvailabilityZone": {
            "Name": "us-east-1d"
```

```
          }
        }
      ],
      "VpcId": "vpc-036a8150d4300bcf2",
      "Name": "mysubnetgroup",
      "ARN": "arn:aws:memorydb:us-east-1:53791xzzz7620:subnetgroup/mysubnetgroup",
      "Description": "test"
    }
  ]
}
```

Para ver detalhes sobre todos os grupos de sub-redes, use o mesmo comando, mas sem especificar um nome de grupo de sub-rede.

```
aws memorydb describe-subnet-groups
```

Para obter mais informações, consulte o AWS CLI tópicodescribe-subnet-groups.

Visualizando grupos de sub-redes (API do MemoryDB)

Usando a API do MemoryDB, chame `DescribeSubnetGroups` com os seguintes parâmetros:

`SubnetGroupName=mysubnetgroup`

Example

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=UpdateSubnetGroup
    &Description=New%20description
    &SubnetGroupName=mysubnetgroup
    &SubnetIds.member.1=subnet-42df9c3a
    &SubnetIds.member.2=subnet-48fc21a9
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Timestamp=20211801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Credential=<credential>
    &X-Amz-Date=20210801T220302Z
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Signature=<signature>
```

```
    &X-Amz-SignedHeaders=Host
```

Exclusão de um grupo de sub-redes

Se você decidir que não precisa mais do seu grupo de sub-redes, poderá excluí-lo. Não será possível excluir um grupo de sub-redes se ele estiver sendo usado atualmente por um cluster. Também não é possível excluir um grupo de sub-redes em um cluster com Multi-AZ habilitado se isso deixar esse cluster com menos de duas sub-redes. É necessário primeiro desabilitar o Multi-AZ e excluir a sub-rede.

Os procedimentos a seguir mostram como excluir um grupo de sub-redes.

Exclusão de um grupo de sub-redes (console)

Para excluir um grupo de sub-redes

1. Faça login AWS Management Console e abra o console MemoryDB for Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação esquerdo, escolha Subnet Groups.
3. Na lista de grupos de sub-rede, escolha o que você deseja excluir, selecione Ações e, em seguida, selecione Excluir.

   > **Note**
   >
   > Não é possível excluir um grupo de sub-rede padrão ou que esteja associado a qualquer cluster.

4. A tela de confirmação Excluir grupos de subrede será exibida.
5. Para excluir o grupo de sub-redes, insira `delete` na caixa de texto de confirmação. Para manter o grupo de sub-redes, escolha Cancelar.

Excluindo um grupo de sub-redes (CLI AWS)

Usando o AWS CLI, chame o comando delete-subnet-group com o seguinte parâmetro:

- `--subnet-group-name` *`mysubnetgroup`*

Para Linux, macOS ou Unix:

```
aws memorydb delete-subnet-group \
```

```
    --subnet-group-name mysubnetgroup
```

Para Windows:

```
aws memorydb delete-subnet-group ^
    --subnet-group-name mysubnetgroup
```

Para obter mais informações, consulte o AWS CLI tópico delete-subnet-group.

Exclusão de um grupo de sub-redes (API do MemoryDB)

Usando a API do MemoryDB, chame `DeleteSubnetGroup` com o seguinte parâmetro:

- `SubnetGroupName=mysubnetgroup`

Example

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DeleteSubnetGroup
    &SubnetGroupName=mysubnetgroup
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Credential=<credential>
    &X-Amz-Date=20210801T220302Z
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Signature=<signature>
    &X-Amz-SignedHeaders=Host
```

Este comando não produz saída.

Para obter mais informações, consulte o tópico da API MemoryDB. DeleteSubnetGroup

API do MemoryDB para Redis e interface de endpoints da VPC (AWS PrivateLink)

Você pode estabelecer uma conexão privada entre sua VPC e os endpoint da API da Amazon MemoryDB para Redis criando uma interface de endpoint da VPC. Os endpoints da interface são desenvolvidos pela AWS PrivateLink. O AWS PrivateLink permite que você acesse de forma privada

o MemoryDB para operações da API do Redis sem um gateway da Internet, dispositivo NAT, conexão VPN ou conexão Direct Connect da AWS.

As instâncias em sua VPC não precisam de endereços IP públicos para se comunicar com os endpoints da API do MemoryDB para Redis. As instâncias também não precisam de endereços IP públicos para usar qualquer uma das operações de API do MemoryDB disponíveis. O tráfego entre sua VPC e o MemoryDB não deixa a rede da Amazon. Cada endpoint de interface é representado por uma ou mais interfaces de rede elástica nas sub-redes. Para obter mais informações sobre interfaces de rede elástica, consulte Interfaces de rede elástica no Guia do usuário do Amazon EC2.

- Para obter mais informações sobre endpoints da VPC , consulte endpoints da VPC de interface (AWS PrivateLink) no Guia do usuário da Amazon VPC.
- Para obter mais informações sobre as operações da API do MemoryDB, consulte Operações da API do MemoryDB.

Depois de criar uma interface de endpoint da VPC, se você ativar nomes de host DNS privados para o endpoint, o endpoint padrão do MemoryDB (https://memorydb.*Region*.amazonaws.com) será resolvido para o seu endpoint da VPC. Se você não habilitar nomes de host DNS privados, o Amazon VPC fornecerá um nome de endpoint DNS que poderá ser usado no seguinte formato:

```
VPC_Endpoint_ID.memorydb.Region.vpce.amazonaws.com
```

Para obter mais informações, consulte Endpoints da VPC de interface (AWS PrivateLink) no Manual do usuário do Amazon VPC. O MemoryDB oferece suporte a chamadas para todas as suas ações de API dentro de sua VPC.

> **Note**
>
> Os nomes de host DNS privados podem ser habilitados para apenas um endpoint da VPC na VPC. Se você quiser criar um endpoint da VPC adicional, o nome de host DNS privado deve ser desabilitado para ele.

Considerações sobre endpoints da VPC do

Antes de configurar uma interface de endpoint da VPC para os endpoints da API do MemoryDB para Redis, verifique as propriedades e limitações do endpoint de interface no Guia do Usuário do Amazon VPC. Todas as operações da API do MemoryDB que são relevantes para gerenciar

os recursos do MemoryDB para Redis estão disponíveis em sua VPC usando o AWS PrivateLink. As políticas de endpoint da VPC têm suporte para endpoints da API do MemoryDB. Por padrão, o acesso total às operações de API do MemoryDB é permitido através do endpoint. Para obter mais informações, consulte Controlar o acesso a serviços com endpoints da VPC no Guia do usuário da Amazon VPC.

Criação de uma interface de endpoint da VPC para a API do MemoryDB

É possível criar um endpoint da VPC para a API do MemoryDB para Redis usando o console da Amazon VPC ou a AWS CLI. Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário da Amazon VPC.

Depois de criar um endpoint da interface da VPC, você poderá habilitar nomes de host DNS privados para o endpoint. Ao fazer isso, o endpoint padrão do MemoryDB para Redis (https://memorydb.*Region*.amazonaws.com) é resolvido para seu endpoint da VPC. Para obter mais informações, consulte Acessar um serviço por um endpoint de interface no Guia do usuário da Amazon VPC.

Criação de uma política de endpoint da VPC para a API do MemoryDB da Amazon

É possível anexar uma política de endpoint ao seu endpoint da VPC que controla o acesso à API do MemoryDB. A política especifica o seguinte:

- A entidade principal que pode executar ações.
- As ações que podem ser executadas.
- Os recursos sobre os quais as ações podem ser realizadas.

Para obter mais informações, consulte Controlar o acesso a serviços com endpoints da VPC no Guia do usuário da Amazon VPC.

Example Política de endpoint da VPC para ações da API do MemoryDB

Veja a seguir um exemplo de uma política de endpoint para a API do MemoryDB. Quando anexada a um endpoint, essa política concede acesso às ações indicadas da API do MemoryDB para todas as entidades principais em todos os recursos.

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
```

```
  "Action": [
   "memorydb:CreateCluster",
   "memorydb:UpdateCluster",
   "memorydb:CreateSnapshot"
  ],
  "Resource": "*"
 }]
}
```

Example Política de endpoint da VPC que nega todo o acesso de uma conta da AWS especificada

A política de VPC endpoint a seguir nega à conta da AWS *123456789012* todos os acessos aos recursos que usam o endpoint. A política permite todas as ações de outras contas.

```
{
 "Statement": [{
   "Action": "*",
   "Effect": "Allow",
   "Resource": "*",
   "Principal": "*"
  },
  {
   "Action": "*",
   "Effect": "Deny",
   "Resource": "*",
   "Principal": {
    "AWS": [
     "123456789012"
    ]
   }
  }
 ]
}
```

Atualizações de serviço no MemoryDB para Redis

O MemoryDB para Redis monitora automaticamente sua frota de clusters e nós para aplicar atualizações de serviços à medida que elas se tornam disponíveis. Normalmente, você configura uma janela de manutenção predefinida para que o MemoryDB possa aplicar essas atualizações. No entanto, em alguns casos, você pode achar que essa abordagem é muito rígida e que provavelmente restringirá os fluxos de negócios.

Com as atualizações de serviço, é possível controlar quando e quais atualizações são aplicadas. Também é possível monitorar o andamento dessas atualizações dos clusters do MemoryDB selecionados em tempo real.

Gerenciamento das atualizações de serviços

As atualizações de serviços do MemoryDB são liberadas regularmente. Se tiver um ou mais clusters qualificados para essas atualizações de serviços, você receberá notificações por e-mail, SNS, Personal Health Dashboard (PHD), e Amazon CloudWatch Events quando as atualizações forem lançadas. As atualizações também são exibidas na página Atualizações de serviços no console do Atualizações de serviços. Usando este painel, é possível visualizar todas as atualizações de serviço e os status em relação à sua frota do MemoryDB.

Você controla quando aplicar uma atualização antes do início da atualização automática. Recomendamos veementemente que você aplique qualquer atualização do tipo security-update para garantir que o seu MemoryDB esteja sempre atualizado com os patches de segurança atuais.

As seguintes seções analisam essas opções em detalhes.

Tópicos

Como aplicar as atualizações de serviço

Será possível começar a aplicar as atualizações de serviços à sua frota desde o momento em que as atualizações tiverem um status disponível. As atualizações de serviço são cumulativas. Em outras palavras, qualquer atualização que você ainda não tiver aplicado serão incluídas na sua atualização mais recente.

Se uma atualização de serviço tiver a atualização automática habilitada, será possível optar por não realizar nenhuma ação quando ela estiver disponível. O MemoryDB agendará a aplicação da atualização durante a janela de manutenção dos clusters após a Data de início da atualização automática. Você receberá notificações relacionadas a cada etapa da atualização.

> **Note**
>
> É possível aplicar somente as atualizações de serviço que tenham um status disponível ou programado.

Para obter mais informações sobre a análise e a aplicação de qualquer atualização específica ao serviço aos clusters do MemoryDB aplicáveis, consulte Aplicação de atualizações de serviço usando o console.

Quando uma nova atualização de serviço está disponível para um ou mais dos clusters do MemoryDB, é possível usar o console do MemoryDB, a API ou a AWS CLI para aplicar a atualização. As seções a seguir explicam as opções que você pode usar para aplicar as atualizações.

Aplicação de atualizações de serviço usando o console

Para visualizar a lista de atualizações de serviço disponíveis, além de outras informações, acesse a página Atualizações de serviço no console.

1. Faça login no AWS Management Console e abra o console do MemoryDB para Redis em https://console.aws.amazon.com/memorydb/.
2. No painel de navegação, selecione Atualizações de serviço.

Em Atualizações de serviço, é possível visualizar o seguinte:

- Nome da atualização de serviço: o nome exclusivo da atualização de serviço
- Atualização de serviço: fornece informações detalhadas sobre a atualização de serviço
- Data de início da atualização automática: se esse atributo for definido, o MemoryDB começará a programar seus clusters para serem atualizados automaticamente nas janelas de manutenção apropriadas após essa data. Você receberá notificações com antecedência sobre a janela exata de manutenção programada, que pode não ser a imediata após a data de início da atualização automática. Você ainda pode aplicar a atualização aos seus clusters sempre que quiser. Se o atributo não estiver definido, a atualização do serviço não estará habilitada para atualização automática e o MemoryDB não atualizará seus clusters automaticamente.

Em Status da atualização do cluster, é possível visualizar uma lista de clusters nos quais a atualização do serviço não foi aplicada ou acabou de ser aplicada recentemente. Para cada cluster, é possível visualizar o seguinte:

- Nome do cluster: o nome do cluster
- Nós atualizados: a proporção de nós individuais dentro de um cluster específico que foram atualizados ou permanecem disponíveis para a atualização de serviço específica.

- Tipo de atualização: o tipo da atualização de serviço, que é security-update (atualização-de-segurança) ou engine-update (atualização-de-mecanismo)
- Status: o status da atualização de serviço no cluster, que é um dos seguintes:
  - disponível: a atualização está disponível para clusters de requisito.
  - em andamento: a atualização está sendo aplicada a esse cluster.
  - programada: a data de atualização foi programada.
  - concluída: a atualização foi aplicada com êxito. O cluster com status completo será exibido por 7 dias após sua conclusão.

  Se você escolheu qualquer um ou todos os clusters com o status disponível ou programado e, em seguida, escolheu Aplicar agora, a atualização começará a ser aplicada nesses clusters.

Aplicação das atualizações de serviços usando a AWS CLI

Depois de receber a notificação de que há atualizações de serviços disponíveis, você poderá inspecioná-las e aplicá-las usando a AWS CLI:

- Para recuperar uma descrição das atualizações de serviços disponíveis, execute o seguinte comando:

  ```
  aws memorydb describe-service-updates --status available
  ```

  Para obter mais informações, consulte describe-service-updates.
- Para aplicar uma atualização de serviço em uma lista de clusters, execute o seguinte comando:

  ```
  aws memorydb batch-update-cluster --service-update
  ServiceUpdateNameToApply=sample-service-update --cluster-names cluster-1
  cluster2
  ```

  Para obter mais informações, consulte batch-update-cluster.

Referência

Os tópicos desta seção abrangem o trabalho com a API do MemoryDB e a seção da AWS CLI do MemoryDB. Também estão incluídas nesta seção mensagens de erro comuns e notificações de serviço.

- Usando a API do MemoryDB
- Referência da API do MemoryDB
- Seção do MemoryDB de referência da AWS CLI

Usando a API do MemoryDB

Esta seção fornece descrições orientadas por tarefas de como usar e implementar as operações do MemoryDB. Para uma descrição completa dessas operações, consulte a Referência da API do MemoryDB.

Tópicos

Como usar a API de consulta

Parâmetros de consulta

As solicitações baseadas em consulta HTTP são solicitações HTTP que usam o verbo HTTP GET ou POST e um parâmetro de consulta chamado `Action`.

Cada solicitação de consulta deve incluir alguns parâmetros comuns para lidar com a autenticação e a seleção de uma ação.

Algumas operações levam listas de parâmetros. Essas listas são especificadas usando a notação `param.`*n*. Os valores de *n* são inteiros a partir de 1.

Autenticação de solicitação de consulta

Só é possível enviar solicitações de consulta por meio de HTTPS, e é preciso incluir uma assinatura em todas as solicitações de consulta. Esta seção descreve como criar a assinatura. O método descrito no procedimento a seguir é conhecido como versão de assinatura 4.

As etapas básicas a seguir são usadas para autenticar as solicitações à AWS. Isso presume que você esteja registrado na AWS e possua um ID de chave de acesso e uma chave de acesso secreta.

Processo de autenticação de consulta

1. O remetente elabora uma solicitação à AWS.
2. O remetente calcula a assinatura da solicitação, um hash codificado para o HMAC (Hash-based Message Authentication Code) com uma função de hash SHA-1, conforme definido na próxima seção deste tópico.

3. O remetente da solicitação envia os dados da solicitação, a assinatura e o ID da chave de acesso (o identificador da chave de acesso secreta usada) à AWS.
4. A AWS usa o ID de chave de acesso para pesquisar pela chave de acesso secreta.
5. A AWS gera uma assinatura a partir dos dados da solicitação e da chave de acesso secreta usando o mesmo algoritmo usado para calcular a assinatura enviada na solicitação.
6. Se as assinaturas coincidirem, a solicitação será considerada autêntica. Se a comparação falhar, a solicitação será descartada e a AWS retornará uma resposta de erro.

> **Note**
>
> Se uma solicitação contiver um parâmetro `Timestamp`, a assinatura calculada para a solicitação expirará 15 minutos após o valor.
> Se uma solicitação contiver um parâmetro `Expires`, a assinatura expirará no horário especificado pelo parâmetro `Expires`.

Para calcular a assinatura da solicitação

1. Crie a query string canonizada de que você precisará posteriormente neste procedimento:
   a. Classifique os componentes query string UTF-8 por nome do parâmetro com o ordenamento natural de bytes. Os parâmetros podem vir do URI GET ou do corpo POST (quando Content-Type for application/x-www-form-urlencoded).
   b. Codificar em URL o nome do parâmetro e os valores de acordo com as seguintes regras:
      i. Não codificar em URL nenhum dos caracteres não reservados que definem o RFC 3986. Esses caracteres não reservados são A–Z, a–z, 0–9, hífen (-), sublinhado (_), ponto (.) e til (~).
      ii. Codificar em percentual todos os outros caracteres com %XY, onde X e Y são caracteres hexadecimais de 0 a 9 e maiúsculas de A a F.
      iii. Codificar em percentual os caracteres UTF-8 estendidos na forma %XY%ZA...
      iv. Codificar em percentual o caractere de espaço como %20 (e não +, como em esquemas de codificação comuns).
   c. Separe os nomes de parâmetro codificados a partir de seus valores codificados com o sinal de igual (=) (caractere ASCII 61), mesmo se o valor do parâmetro estiver vazio.

d. Separe os pares de nome-valor por um "&" (e comercial) (código 38 em ASCII).

2. Crie a string para assinar de acordo com a seguinte pseudogramática ("\n" representa uma nova linha em ASCII).

```
StringToSign = HTTPVerb + "\n" +
ValueOfHostHeaderInLowercase + "\n" +
HTTPRequestURI + "\n" +
CanonicalizedQueryString <from the preceding step>
```

O componente HTTPRequestURI é o componente de caminho absoluto HTTP do URI até a query string (não incluída). Se o HTTPRequestURI estiver vazio, use uma barra (/).

3. Calcule um HMAC compatível com RFC 2104 com a string recém-criada, sua chave de acesso secreta como chave e SHA256 ou SHA1 como algoritmo de hash.

Para obter mais informações, consulte https://www.ietf.org/rfc/rfc2104.txt.

4. Converta o valor resultante para base64.
5. Inclua o valor como o valor do parâmetro `Signature` na solicitação.

Por exemplo, a seguir você encontra um exemplo de solicitação (as quebras de linha foram adicionadas para maior clareza).

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeClusters
    &ClusterName=myCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2021-01-01
```

Quanto à string de consulta anterior, você calcularia a assinatura HMAC na seguinte string.

```
GET\n
    memory-db.amazonaws.com\n
    Action=DescribeClusters
    &ClusterName=myCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2021-01-01
```

```
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Credential=AKIADQKE4SARGYLE%2F20140523%2Fus-east-1%2Fmemorydb%2Faws4_request
    &X-Amz-Date=20210801T223649Z
    &X-Amz-SignedHeaders=content-type%3Bhost%3Buser-agent%3Bx-amz-content-sha256%3Bx-
amz-date
        content-type:
        host:memory-db.us-east-1.amazonaws.com
        user-agent:ServicesAPICommand_Client
    x-amz-content-sha256:
    x-amz-date:
```

O resultado é a seguinte solicitação assinada.

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeClusters
    &ClusterName=myCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Credential=AKIADQKE4SARGYLE/20141201/us-east-1/memorydb/aws4_request
    &X-Amz-Date=20210801T223649Z
    &X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
    &X-Amz-Signature=2877960fced9040b41b4feaca835fd5cfeb9264f768e6a0236c9143f915ffa56
```

Para obter informações detalhadas sobre o processo de assinatura e o cálculo da assinatura da solicitação, consulte o tópico Processo de assinatura do Signature Version 4 e seus subtópicos.

Bibliotecas disponíveis

A AWS fornece kits de desenvolvimento de software (SDKs) para desenvolvedores de software que preferem criar aplicações usando APIs específicas de linguagem em vez da API de consulta. Esses SDKs oferecem as funções básicas (não incluídas nas APIs), como autenticação de solicitação, novas tentativas de solicitação e processamento de erros, para que você possa começar a usar com mais facilidade. SDKs e recursos adicionais estão disponíveis para as seguintes linguagens de programação:

- Java
- Windows and .NET
- PHP

- Python
- Ruby

Para obter informações sobre outras linguagens, consulte Código e bibliotecas de exemplo.

Solução de problemas de aplicações

O MemoryDB fornece erros específicos e descritivos para ajudá-lo a solucionar problemas durante a interação com a API do MemoryDB.

Recuperação de erros

Normalmente, espera-se que o aplicativo verifique se uma solicitação gerou um erro antes que você precise processar os resultados. A maneira mais fácil de descobrir se ocorreu um erro é procurar por um `Error`nó na resposta da API do MemoryDB.

A sintaxe XPath apresenta uma maneira simples de procurar pela presença de um nó `Error`, bem como uma maneira fácil de recuperar o código e a mensagem de erro. O snippet de código a seguir usa Perl e o módulo XML::XPath para determinar se ocorreu um erro durante uma solicitação. Caso tenha ocorrido, o código imprimirá o primeiro código de erro e a mensagem na resposta.

```
use XML::XPath;
my $xp = XML::XPath->new(xml =>$response);
if ( $xp->find("//Error") )
{print "There was an error processing your request:\n", " Error code: ",
$xp->findvalue("//Error[1]/Code"), "\n", " ",
$xp->findvalue("//Error[1]/Message"), "\n\n"; }
```

Dicas de solução de problemas

Recomendamos os seguintes processos para diagnosticar e resolver problemas com a API do MemoryDB.

- Verifique se o MemoryDB está funcionando corretamente.

  Para fazer isso, basta abrir uma janela do navegador e enviar uma solicitação de consulta para o serviço do MemoryDB (como o https://memory-db.us-east-1.amazonaws.com). Uma MissingAuthenticationTokenException ou UnknownOperationException confirma que o serviço está disponível e respondendo a solicitações.

- Verificação da estrutura de sua solicitação.

  Cada operação do MemoryDB tem uma página de referência na Referência da API do MemoryDB. Verifique novamente se você está usando os parâmetros corretamente. Para conceder ideias sobre o que pode estar errado, consulte as amostras de solicitações ou cenários de usuários para ver se esses exemplos estão realizando operações similares.
- Verificação do fórum.

  O MemoryDB tem um fórum de discussão onde você pode procurar soluções para os problemas que outros usuários enfrentaram ao longo do caminho. Para exibir o fórum, consulte

  https://forums.aws.amazon.com/ .

Cotas para o Amazon MemoryDB para Redis

Sua AWS conta tem cotas padrão, anteriormente chamadas de limites, para cada AWS serviço. A menos que especificado de outra forma, cada cota é específica da região . Você pode solicitar o aumento de algumas cotas, porém, algumas delas não podem ser aumentadas.

Para solicitar o aumento da cota, consulte Solicitar um aumento de cota no Guia do usuário do Service Quotas. Se a cota ainda não estiver disponível no Service Quotas, use o formulário de aumento de limite.

Sua AWS conta tem as seguintes cotas relacionadas ao MemoryDB.

Recurso	Padrão
Nós por região	300
Nós por cluster por tipo de instância	90
Nódulos por fragmento	6
Grupos de parâmetros por região	150
Grupos de sub-redes por região	150
Sub-redes por grupo de sub-rede	20
Usuários por grupo de usuários	100
Número total de usuários	1000
Número de grupos de usuários	100

Histórico de documentos do Guia do Usuário do MemoryDB

A tabela a seguir descreve as versões da documentação do MemoryDB.

Alteração	Descrição	Data
O MemoryDB agora oferece suporte à autenticação de usuários usando o IAM	A autenticação IAM permite autenticar uma conexão com o MemoryDB para Redis usando identidades AWS Identity and Access Management. Isso possibilita que você fortaleça seu modelo de segurança e simplifique várias tarefas administrativas de segurança. Para obter mais informações, consulte Autenticação com o IAM.	10 de maio de 2023
O MemoryDB agora oferece suporte ao Redis 7	Esta versão traz vários novos atributos ao MemoryDB para Redis: funções Redis, aprimoramentos de ACL, Pub/Sub em fragmento e multiplex ação de E/S aprimorada. Para obter mais informaçõ es, consulte Versões do mecanismo Redis.	9 de maio de 2023
O MemoryDB agora oferece nós reservados	Os nós reservados fornecem um desconto significativo em comparação com os preços de nós sob demanda. Os nós reservados não são nós físicos, mas um desconto na fatura aplicado na sua conta pelo uso de nós sob demanda.	27 de dezembro de 2022

	Para obter mais informações, consulte Nós reservados do MemoryDB.	
O MemoryDB agora oferece suporte à classificação de dados em níveis	MemoryDB para Redis classifica dados em níveis. Você pode usar a classific ação de dados em níveis como uma maneira de menor custo para escalar seus clusters para até centenas de terabytes de capacidade. Para mais informações, consulte Classificação de dados em níveis.	3 de novembro de 2022
O MemoryDB agora oferece suporte ao formato JavaScript Object Notation (JSON) nativo	O formato JSON nativo é uma forma simples e sem esquema de codificar conjuntos de dados complexos em clusters do Redis. É possível armazenar e acessar dados nativamente usando o formato JSON em clusters do Redis e atualizar dados JSON armazenados nesses clusters sem precisar gerenciar código personalizado para serializá -los e desserializá-los. Para obter mais informações, consulte Conceitos básicos do JSON.	25 de maio de 2022

O MemoryDB agora oferece suporte ao PrivateLink AWS	O PrivateLink da AWS permite que você acesse de forma privada as operações da API do MemoryDB sem um gateway da Internet, dispositivo NAT, conexão VPN ou conexão Direct Connect da AWS. Para obter mais informações, consulte API do MemoryDB e interface de endpoint da VPC (AWS PrivateLink).	24 de janeiro de 2022
Versão inicial	Versão inicial do Guia do Usuário do MemoryDB. Para mais informações, consulte O que é o MemoryDB?	19 de agosto de 2021

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.