



Entwicklerhandbuch

Amazon MemoryDB für Redis



Amazon MemoryDB für Redis: Entwicklerhandbuch

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist MemoryDB for Redis?	1
Funktionen von MemoryDB	1
MemoryDB-Kernkomponenten	2
Cluster	3
Knoten	4
Shards	5
Parametergruppen	5
Subnetzgruppen	5
Zugriffskontrolllisten	6
Benutzer	6
Zugehörige Services	6
Auswählen von Regionen und Availability Zones	7
Lokalisieren Ihrer Knoten	8
Unterstützte Regionen und Endpunkte	9
Zugreifen auf MemoryDB	12
MemoryDB-Sicherheit	13
Erste Schritte mit MemoryDB	14
Einrichtung	14
Erstellen Sie Ihr Konto AWS	15
Erteilen programmgesteuerten Zugriffs	17
Richten Sie Ihre Berechtigungen ein (nur für neue MemoryDB-Benutzer)	18
AWS CLI herunterladen und konfigurieren	19
Schritt 1: Erstellen eines Clusters	21
Einen MemoryDB-Cluster erstellen	21
Authentifizierung einrichten	32
Schritt 2: Autorisieren Sie den Zugriff auf den Cluster	33
Schritt 3: Connect zum Cluster herstellen	35
Finden Sie Ihren Cluster-Endpoint	35
Stellen Sie eine Connect zu einem MemoryDB-Cluster her (Linux)	35
Schritt 4: Löschen eines Clusters	37
Wie geht es weiter?	40
Knoten verwalten	41
MemoryDB-Knoten und Shards	41
Unterstützte Knotentypen	43

Reservierte Knoten	45
Überblick über reservierte Knoten	45
Ersetzen von Knoten	57
Verwalten von Clustern	60
Daten-Tiering	61
Bewährte Methoden	62
Einschränkungen	62
Preise für Daten-Tiering	63
Überwachung	63
Verwenden von Daten-Tiering	63
Wiederherstellen von Daten aus einem Snapshot in Cluster mit aktiviertem Daten-Tiering	65
Vorbereitung eines Clusters	66
Bestimmung Ihrer Anforderungen	67
Erstellen eines Clusters	70
Anzeigen der Details eines Clusters	71
Modifizieren eines Clusters	76
Knoten zu einem Cluster hinzufügen/entfernen	79
Zugriff auf Ihren Cluster	81
Gewähren Sie Zugriff auf Ihren Cluster	81
Von außen auf MemoryDB zugreifenAWS	83
Verbindungsendpunkte ermitteln	89
Shards	92
Auffinden der Namen eines Shards	93
Verwaltung Ihrer MemoryDB-Implementierung	97
Engine-Versionen	97
Redis 7.0 (erweitert)	97
Redis 7.0 (erweitert)	98
Redis 6.2 (erweitert)	99
Upgrade von Engine-Versionen	100
Erste Schritte mit JSON	102
Überblick über den Redis-JSON-Datentyp	103
Unterstützte Befehle	116
Markierung Ihrer MemoryDB-Ressourcen	158
Überwachung von Kosten mit Tags	163
Verwalten von Tags mit dem AWS CLI	165
Verwalten von Tags mithilfe der MemoryDB-API	168

Tutorial: Konfiguration einer Lambda-Funktion für den Zugriff auf MemoryDB in einer Amazon VPC	296
Schritt 1: Erstellen eines Clusters	296
Schritt 2: Erstellen einer Lambda-Funktion	299
Schritt 3: Testen der Lambda -Funktion	303
Schritt 4: Aufräumen (optional)	304
Vektor-Suche	306
Überblick über die Vektorsuche	306
Indizes und Schlüsselräume	307
Indexfeldtypen	308
Vektorindex-Algorithmen	309
Ausdruck für eine Vektor-Suchanfrage	310
Befehl INFO	312
Sicherheit bei der Vektorsuche	315
Funktionen und Einschränkungen der Vektorsuche	316
Verfügbarkeit der Vektorsuche	316
Parametrische Einschränkungen	316
Skalierungsgrenzen	317
Betriebliche Einschränkungen	317
Import/Export von Snapshots und Live-Migration	318
Speicherverbrauch	318
Nicht genügend Speicher beim Auffüllen	318
Transaktionen	318
Anwendungsfälle	319
Retrieval Augmented Generation (RAG)	319
Foundation Model (FM) -Pufferspeicher	319
Betrugserkennung	320
Andere Anwendungsfälle	321
Mit dem AWS Management Console	321
Verwenden von AWS Command Line Interface	322
Befehle für die Vektorsuche	322
FT.CREATE	323
FT.SEARCH	327
FT.AGGREGATE	330
FT.DROPINDEX	331
FT.INFO	332

FT. _LISTE	334
FT.ALIASADD	335
FT.ALIASDEL	335
FT.ALIASUPDATE	335
FT. _ALIASLISTE	336
FT.CONFIG ABRUFEN	336
FT.CONFIG-HILFE	336
FT.CONFIG GESETZT	337
FT.PROFILE	337
FT.ERLÄUTERN	338
FT.EXPLAINCLI	338
Sicherheit	339
Datenschutz	340
Datensicherheit in MemoryDB für Redis	341
Verschlüsselung im Ruhezustand	342
Verschlüsselung während der Übermittlung (TLS)	345
Benutzer mit ACLs authentifizieren	346
Authentifizieren mit IAM	361
Identity and Access Management	369
Zielgruppe	370
Authentifizierung mit Identitäten	370
Verwalten des Zugriffs mit Richtlinien	374
Wie funktioniert MemoryDB for Redis mit IAM	377
Beispiele für identitätsbasierte Richtlinien	388
Fehlerbehebung	391
Zugriffskontrolle	393
Übersicht über die Verwaltung von Zugriffsberechtigungen	395
Protokollierung und Überwachung	425
Überwachung mit CloudWatch	426
Überwachung von Ereignissen	446
Protokollieren von MemoryDB für Redis-API-Aufrufe mit AWS CloudTrail	460
Compliance-Validierung	467
Sicherheit der Infrastruktur	468
Richtlinie für den Datenverkehr zwischen Netzwerken	469
MemoryDB und Amazon VPC	469
Subnetze und Subnetzgruppen	482

VPC-Endpunkte MemoryDB für Redis-API und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)	496
Service-Updates	499
Verwalten der Service-Updates	500
Referenz	503
Verwenden der MemoryDB-API	504
Verwenden der Abfrage-API	504
Verfügbare Bibliotheken	508
Fehlerbehebung bei Anwendungen	508
Kontingente	510
Dokumentverlauf	511
.....	dxiv

Was ist MemoryDB for Redis?

MemoryDB for Redis ist ein langlebiger In-Memory-Datenbankdienst, der ultraschnelle Leistung bietet. Er wurde speziell für moderne Anwendungen mit Microservices-Architekturen entwickelt.

MemoryDB ist mit Redis, einem beliebten Open-Source-Datenspeicher, kompatibel, sodass Sie schnell Anwendungen mit denselben flexiblen und benutzerfreundlichen Redis-Datenstrukturen, APIs und Befehlen erstellen können, die sie bereits heute verwenden. Mit MemoryDB werden alle Ihre Daten im Arbeitsspeicher gespeichert, sodass Sie Leselatenz im Mikrosekundenbereich und Schreibvorgänge im einstelligen Millisekundenbereich sowie einen hohen Durchsatz erreichen können. MemoryDB speichert Daten auch dauerhaft in mehreren Availability Zones (AZs) mithilfe eines Multi-AZ-Transaktionsprotokolls, um ein schnelles Failover, eine Datenbankwiederherstellung und Knotenneustarts zu ermöglichen.

MemoryDB bietet sowohl In-Memory-Leistung als auch Multi-AZ-Beständigkeit und kann als leistungsstarke Primärdatenbank für Ihre Microservices-Anwendungen verwendet werden. Dadurch entfällt die Notwendigkeit, sowohl einen Cache als auch eine dauerhafte Datenbank separat zu verwalten.

Themen

- [Funktionen von MemoryDB](#)
- [MemoryDB-Kernkomponenten](#)
- [Zugehörige Services](#)
- [Auswählen von Regionen und Availability Zones](#)
- [Zugreifen auf MemoryDB](#)
- [MemoryDB-Sicherheit](#)

Funktionen von MemoryDB

MemoryDB for Redis ist ein langlebiger In-Memory-Datenbankdienst, der ultraschnelle Leistung bietet. Zu den Funktionen von MemoryDB gehören:

- Starke Konsistenz für Primärknoten und garantierte letztendliche Konsistenz für Replikatknoten. Weitere Informationen finden Sie unter [Konsistenz](#).
- Lese- und Schreiblatenzen im einstelligen Millisekundenbereich mit bis zu 160 Millionen TPS pro Cluster.

- Flexible und benutzerfreundliche Redis-Datenstrukturen und APIs. Erstellen Sie ganz einfach neue Anwendungen oder migrieren Sie bestehende Redis-Anwendungen fast ohne Änderungen.
- Datenbeständigkeit mithilfe eines Multi-AZ-Transaktionsprotokolls, das eine schnelle Wiederherstellung und einen schnellen Neustart der Datenbank ermöglicht.
- Multi-AZ-Verfügbarkeit mit automatischem Failover sowie Erkennung und Wiederherstellung von Knotenausfällen.
- Skalieren Sie einfach horizontal, indem Sie Knoten hinzufügen und entfernen, oder vertikal, indem Sie zu größeren oder kleineren Knotentypen wechseln. Sie können den Schreibdurchsatz durch Hinzufügen von Shards und den Lesedurchsatz durch Hinzufügen von Replikaten skalieren.
- Read-after-write R-Konsistenz für Primärknoten und garantierte letztendliche Konsistenz für Replikatknoten.
- MemoryDB unterstützt Verschlüsselung bei der Übertragung, Verschlüsselung im Ruhezustand und Authentifizierung von Benutzern über. [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#)
- Automatische Snapshots in Amazon S3 mit Aufbewahrung für bis zu 35 Tage.
- Support für bis zu 500 Knoten und mehr als 100 TB Speicher pro Cluster (mit 1 Replikat pro Shard).
- Verschlüsselung während der Übertragung mit TLS und Verschlüsselung im Ruhezustand mit Schlüsseln. AWS KMS
- Benutzerauthentifizierung und Autorisierung mit Redis. [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#)
- Support für AWS Graviton2-Instanztypen.
- Integration mit anderen AWS Diensten wie CloudWatch Amazon VPC und Amazon SNS für Überwachung, Sicherheit und Benachrichtigungen. CloudTrail
- Vollständig verwaltetes Software-Patching und -Upgrades.
- AWS-Integration von Identity and Access Management (IAM) und tagbasierte Zugriffskontrolle für Management-APIs.

MemoryDB-Kernkomponenten

Nachstehend finden Sie eine Übersicht zu den wichtigsten Komponenten einer MemoryDB-Bereitstellung.

Themen

- [Cluster](#)
- [Knoten](#)
- [Shards](#)
- [Parametergruppen](#)
- [Subnetzgruppen](#)
- [Zugriffskontrolllisten](#)
- [Benutzer](#)

Cluster

Ein Cluster ist eine Sammlung von einem oder mehreren Knoten, die einem einzelnen Dataset dienen. Ein MemoryDB-Dataset ist in Shards partitioniert und jeder Shard verfügt über einen primären Knoten und bis zu 5 optionale Replikatknoten. Ein primärer Knoten bedient Lese- und Schreibanforderungen, während ein Replikat nur Leseanforderungen bereitstellt. Ein primärer Knoten kann auf einen Replikatknoten ausfallen und dieses Replikat auf den neuen primären Knoten für diesen Shard hochstufen. MemoryDB führt Redis als Datenbankmodul aus, und wenn Sie einen Cluster erstellen, geben Sie die Redis-Version für Ihren Cluster an. Sie können einen Cluster erstellen und ändern, indem Sie die AWS CLI, die MemoryDB-API oder die AWS Management Console aus.

Jeder MemoryDB-Cluster führt eine Redis-Engineversion aus. Jede Redis-Engine-Version verfügt über eigene unterstützte Funktionen. Außerdem verfügt jede Redis-Engine-Version über eine Reihe von Parametern in einer Parametergruppe, die das Verhalten der von ihr verwalteten Cluster steuern.

Die Rechen- und Speicherkapazität eines Clusters wird über seinen Knotentyp festgelegt. Sie können den Knotentyp auswählen, der Ihren Anforderungen am besten entspricht. Wenn sich Ihre Anforderungen im Laufe der Zeit ändern, können Sie die Knotentypen wechseln. Weitere Informationen finden Sie unter [Unterstützte Knotentypen](#).

Note

Weitere Informationen zu -Preisen zu MemoryDB-Knotentypen finden Sie unter [Preise für MemoryDB-Preise](#) aus.

Sie führen einen Cluster in einer Virtual Private Cloud (VPC) mit dem Amazon Virtual Private Cloud Cloud-Service (Amazon VPC) aus. Wenn Sie eine VPC verwenden, haben Sie die Kontrolle über

Ihre virtuelle Netzwerkumgebung. Sie können Ihren eigenen IP-Adressbereich auswählen, Subnetze erstellen sowie Routing-Tabellen und Zugriffskontrolllisten konfigurieren. MemoryDB verwaltet Snapshots, Software-Patching, automatische Fehlererkennung und Wiederherstellung. Für den Betrieb Ihres Clusters in einer VPC fallen keine zusätzlichen Preise an. Weitere Informationen zur Verwendung von Amazon VPC mit MemoryDB finden Sie unter [MemoryDB und Amazon VPC](#) aus.

Viele MemoryDB-Operationen sind auf Cluster ausgerichtet:

- Erstellen eines Clusters
- Modifizieren eines Clusters
- Erstellen von Snapshots eines Clusters
- Löschen eines Clusters
- Anzeigen der Elemente in einem Cluster
- Hinzufügen oder Entfernen von Kostenzuordnungs-Tags in einem Cluster

Detailliertere Informationen finden Sie in den folgenden verwandten Themen:

- [Verwalten von Clustern](#) und [Knoten verwalten](#)

Informationen zu Clustern, Knoten und verwandten Operationen

- [Belastbarkeit in MemoryDB für Redis](#)

Informationen zur Verbesserung der Fehlertoleranz von Clustern.

Knoten

Ein Knoten ist der kleinste Baustein einer MemoryDB-Bereitstellung und wird mit einer Amazon EC2-Instance ausgeführt. Jeder Knoten führt die Redis-Version aus, die gewählt wurden, als Sie Ihren Cluster erstellt haben. Ein Knoten gehört zu einem Shard, der zu einem Cluster gehört.

Jeder Knoten führt eine Instance der Engine mit der Version aus, die gewählt wurden, als Sie Ihren Cluster erstellt haben. Bei Bedarf können Sie die Knoten in einem Cluster auf einen anderen Typ nach oben oder unten skalieren. Weitere Informationen finden Sie unter [Skalierung](#).

Jeder Knoten in einem Cluster ist vom selben Knotentyp. Mehrere Knotentypen werden unterstützt, jeder mit unterschiedlich viel Arbeitsspeicher. Eine Liste der unterstützten Knotentypen finden Sie unter [Unterstützte Knotentypen](#).

Weitere Informationen zu Knoten finden Sie unter [Knoten verwalten](#).

Shards

Ein Shard ist eine Gruppierung von einem bis 6 Knoten, wobei einer als primärer Schreibknoten dient und der andere 5 als Read-Replikate dient. Ein MemoryDB-Cluster hat immer mindestens einen Shard.

MemoryDB-Cluster können bis zu 500 Shards haben, wobei Ihre Daten über die Shards verteilt sind. Sie können beispielsweise einen Cluster mit 500 Knoten konfigurieren, der zwischen 83 Shards (ein primärer Knoten und 5 Replikate pro Shard) und 500 Shards (ein primärer Knoten und keine Replikate) umfasst. Stellen Sie sicher, dass für die Erhöhung genügend IP-Adressen verfügbar sind. Häufige Fallstricke sind Subnetze in der Subnetzgruppe, die einen zu kleinen CIDR-Bereich haben, oder Subnetze, die gemeinsam genutzt und von anderen Clustern stark beansprucht werden.

Ein Shard mit mehreren Knoten implementiert die Replikation durch einen Primärknoten mit Lese-/Schreibzugriff und 1-5 Replikationsknoten. Weitere Informationen finden Sie unter [Grundlegendes zur MemoryDB-Replikation](#).

Weitere Informationen zu Shards finden Sie unter [Arbeiten mit Shards](#).

Parametergruppen

Eine Parametergruppen vereinfachen die Verwaltung von Laufzeiteinstellungen für Redis in Ihrem Cluster. Parameter werden zur Steuerung der Speichernutzung, der Artikelgrößen und mehr verwendet. Eine MemoryDB-Parametergruppe ist eine benannte Sammlung Engine-spezifischer Parameter, die Sie einem Cluster zuweisen können, und alle Knoten in diesem Cluster werden genauso konfiguriert.

Detailliertere Informationen zu MemoryDB-Parametergruppen finden Sie unter [Konfiguration von Engine-Parametern unter Verwendung von Parametergruppen](#) aus.

Subnetzgruppen

Eine Subnetzgruppe ist eine Sammlung von Subnetzen (in der Regel private Subnetze), die Sie für Ihre, in einer Amazon Virtual Private Cloud (VPC)-Umgebung ausgeführten, Cluster festlegen können.

Wenn Sie einen Cluster in einer Amazon VPC erstellen, können Sie eine Subnetzgruppe angeben oder die angegebene Standardgruppe verwenden. MemoryDB verwendet diese Subnetzgruppe, um

ein Subnetz und IP-Adressen innerhalb dieses Subnetzes auszuwählen und diese mit Ihrem Knoten zu verknüpfen.

Detailliertere Informationen zu MemoryDB-Subnetzgruppen finden Sie unter [Subnetze und Subnetzgruppen](#) aus.

Zugriffskontrolllisten

Eine Zugriffskontrollliste ist eine Sammlung von einem oder mehreren Benutzern.

Zugriffszeichenfolgen dem Redis [Regeln für Netzwerk-ACLs](#) um den Benutzerzugriff auf Redis-Befehle und Daten zu autorisieren.

Weitere Informationen zu MemoryDB-Zugriffssteuerungslisten finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#) aus.

Benutzer

Ein Benutzer hat einen Benutzernamen und ein Kennwort und wird verwendet, um auf Daten zuzugreifen und Befehle in Ihrem MemoryDB-Cluster auszugeben. Ein Benutzer ist Mitglied einer Zugriffssteuerungsliste (ACL), mit der Sie Berechtigungen für diesen Benutzer auf MemoryDB-Clustern ermitteln können. Weitere Informationen finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#)

Zugehörige Services

[ElastiCache für Redis](#)

Bei der Entscheidung, ob Sie MemoryDB für Redis oder ElastiCache für Redis verwenden möchten, sollten Sie die folgenden Vergleiche berücksichtigen:

- MemoryDB for Redis ist eine robuste In-Memory-Datenbank für Workloads, die eine ultraschnelle Primärdatenbank erfordern. Sie sollten die Verwendung von MemoryDB in Betracht ziehen, wenn Ihr Workload eine dauerhafte Datenbank erfordert, die eine ultraschnelle Leistung bietet (Leselatenz im Mikrosekundenbereich und Schreiblatenz im einstelligen Millisekundenbereich). MemoryDB eignet sich möglicherweise auch gut für Ihren Anwendungsfall, wenn Sie eine Anwendung mithilfe von Redis-Datenstrukturen und -APIs mit einer primären, dauerhaften Datenbank erstellen möchten. Schließlich sollten Sie erwägen, MemoryDB zu verwenden, um Ihre Anwendungsarchitektur zu vereinfachen und die Kosten zu senken, indem Sie die Verwendung einer Datenbank durch einen Cache ersetzen, um Dauerhaftigkeit und Leistung zu gewährleisten.

- ElastiCache for Redis ist ein Dienst, der häufig verwendet wird, um Daten aus anderen Datenbanken und Datenspeichern mithilfe von Redis zwischenspeichern. Sie sollten Redis ElastiCache für das Zwischenspeichern von Workloads in Betracht ziehen, bei denen Sie den Datenzugriff mit Ihrer vorhandenen Primärdatenbank oder Ihrem vorhandenen Datenspeicher beschleunigen möchten (Lese- und Schreibleistung im Mikrosekundenbereich). Sie sollten Redis auch ElastiCache für Anwendungsfälle in Betracht ziehen, in denen Sie die Redis-Datenstrukturen und APIs verwenden möchten, um auf Daten zuzugreifen, die in einer Primärdatenbank oder einem Datenspeicher gespeichert sind.

Auswählen von Regionen und Availability Zones

AWS Cloud Computing-Ressourcen werden in hochverfügbaren Rechenzentren untergebracht. Die Einrichtungen dieser Rechenzentren befinden sich an verschiedenen Standorten, um die Skalierbarkeit und Zuverlässigkeit zu erhöhen. Diese Standorte sind nach Regionen und Availability Zones kategorisiert.

AWS-Regionen sind große und weit verstreute, separate geografische Bereiche. Availability Zones sind isolierte Standorte innerhalb einer AWS-Region. Sie sind so aufgebaut, dass sie von Ausfällen in anderen Availability Zones nicht betroffen sind. Sie bieten kostengünstige Netzwerkkonnektivität mit geringer Latenz zu anderen Availability Zones in der gleichen AWS-Region.

Important

Jede Region ist komplett eigenständig. Jede MemoryDB-Aktivität, die Sie initiieren (z. B. das Erstellen von Clustern), wird nur in Ihrer aktuellen Standardregion ausgeführt.

Wenn Sie einen Cluster in einer bestimmten Region erstellen oder damit arbeiten möchten, müssen Sie den entsprechenden regionalen Service-Endpunkt wählen. Informationen zu Service-Endpunkten finden Sie unter [Unterstützte Regionen und Endpunkte](#).

Lokalisieren Ihrer Knoten

Jeder Cluster, der über mindestens ein Replikat verfügt, muss auf mehrere AZs verteilt sein. Die einzige Möglichkeit, alles innerhalb einer einzigen AZ zu finden, ist ein Cluster, der aus Shards mit einem Knoten besteht.

Durch die Lokalisierung der Knoten in verschiedenen AZs eliminiert MemoryDB die Möglichkeit, dass ein Ausfall, z. B. ein Stromausfall, in einer AZ zu einem Verlust der Verfügbarkeit führt.

- [Einen MemoryDB-Cluster erstellen](#)
- [Einen MemoryDB-Cluster ändern](#)

Unterstützte Regionen und Endpunkte

MemoryDB for Redis ist in mehreren Regionen verfügbar. AWS Das bedeutet, dass Sie MemoryDB-Cluster an Standorten starten können, die Ihren Anforderungen entsprechen. Beispielsweise können Sie Cluster in einer besonders nah am Kunden befindlichen AWS-Region einrichten oder sie zur Erfüllung bestimmter rechtlicher Anforderungen in einer bestimmten AWS-Region platzieren. Da MemoryDB die Verfügbarkeit auf eine neue AWS Region ausweitet, unterstützt MemoryDB außerdem die beiden jeweils neuesten MAJOR.MINOR Versionen für die neue Region. Weitere Informationen zu MemoryDB-Versionen finden Sie unter [Redis-Engine-Versionen](#)

Standardmäßig verweisen die AWS SDKsAWS CLI, die MemoryDB-API und die MemoryDB-Konsole auf die Region USA-Ost (Nord-Virginia). Da MemoryDB die Verfügbarkeit auf neue Regionen ausdehnt, stehen auch neue Endpunkte für diese Regionen zur Verfügung, die Sie in Ihren HTTP-Anfragen, den SDKs und der Konsole verwenden können. AWS AWS CLI

Jede -Region ist darauf ausgelegt, vollständig von den anderen -Regionen getrennt zu sein. Innerhalb jeder Region gibt es mehrere Availability Zones (Verfügbarkeitszonen, AZ). Indem Sie Ihre Knoten in verschiedenen AZs starten, erreichen Sie die größtmögliche Fehlertoleranz. Weitere Informationen zu Regionen und Availability Zones finden Sie [Auswählen von Regionen und Availability Zones](#) am Anfang dieses Themas.

Regionen, in denen MemoryDB unterstützt wird

Regionsname/Region	Endpunkt	Protokoll	
Region USA Ost (Ohio) us-east-2	memory-db.us-east-2.amazonaws.com	HTTPS	
Region USA Ost (Nord-Virginia) us-east-1	memory-db.us-east-1.amazonaws.com	HTTPS	
Region US West (N. California) us-west-1	memory-db.us-west-1.amazonaws.com	HTTPS	

Regionsname/Region	Endpoint	Protokoll	
Region USA West (Oregon) us-west-2	memory-db.us-west-2.amazonaws.com	HTTPS	
Region Kanada (Zentral) ca-central-1	memory-db.ca-central-1.amazonaws.com	HTTPS	
Region Asien-Pazifik (Hongkong) ap-east-1	memory-db.ap-east-1.amazonaws.com	HTTPS	
Region Asien-Pazifik (Mumbai) ap-south-1	memory-db.ap-south-1.amazonaws.com	HTTPS	
Region Asien-Pazifik (Tokio) ap-northeast-1	memory-db.ap-northeast-1.amazonaws.com	HTTPS	
Region Asien-Pazifik (Seoul) ap-northeast-2	memory-db.ap-northeast-2.amazonaws.com	HTTPS	
Region Asien-Pazifik (Singapur) ap-southeast-1	memory-db.ap-southeast-1.amazonaws.com	HTTPS	
Region Asien-Pazifik (Sydney) ap-southeast-2	memory-db.ap-southeast-2.amazonaws.com	HTTPS	

Regionsname/Region	Endpoint	Protokoll	
Region Europa (Frankfurt) eu-central-1	memory-db.eu-central-1.amazonaws.com	HTTPS	
Region Europa (Irland) eu-west-1	memory-db.eu-west-1.amazonaws.com	HTTPS	
Region Europa (London) eu-west-2	memory-db.eu-west-2.amazonaws.com	HTTPS	
Region Europa (Paris) eu-west-3	memory-db.eu-west-3.amazonaws.com	HTTPS	
Region Europa (Stockholm) eu-north-1	memory-db.eu-north-1.amazonaws.com	HTTPS	
Region Europa (Mailand) eu-south-1	memory-db.eu-south-1.amazonaws.com	HTTPS	
Region Südamerika (São Paulo) sa-east-1	memory-db.sa-east-1.amazonaws.com	HTTPS	
Region China (Peking) cn-north-1	memory-db.cn-north-1.amazonaws.com.cn	HTTPS	

Regionsname/Region	Endpoint	Protokoll	
Region China (Ningxia) cn-northwest-1	memory-db.cn- northwest-1.am azonaws.com.cn	HTTPS	

Eine Tabelle mit AWS Produkten und Dienstleistungen nach Regionen finden Sie unter [Produkte und Dienstleistungen nach Regionen](#).

Eine Tabelle der unterstützten Availability Zones innerhalb der Regionen finden Sie unter [Subnetze und Subnetzgruppen](#).

Zugreifen auf MemoryDB

Jeder MemoryDB-Cluster-Endpoint enthält eine Adresse und einen Port. Dieser Cluster-Endpoint unterstützt das Redis-Cluster-Protokoll, sodass Clients die spezifischen Rollen, IP-Adressen und Steckplätze für jeden Knoten im Cluster ermitteln können. Wenn ein primärer Knoten ausfällt und stattdessen ein Replikat heraufgestuft wird, können Sie mithilfe des Redis-Cluster-Protokolls eine Verbindung zum Cluster-Endpoint herstellen, um den neuen Primärknoten zu ermitteln.

Sie müssen mit dem Befehl `cluster nodes` eine Verbindung zum Cluster-Endpoint herstellen, um Knotenendpunkte zu ermitteln. Nachdem Sie den richtigen Knoten für einen Schlüssel gefunden haben, können Sie sich für Lese-/Schreibanforderungen direkt mit dem Knoten verbinden. Ein Redis-Client kann den Cluster-Endpoint verwenden, um automatisch eine Verbindung zum richtigen Knoten herzustellen.

Um Fehler bei bestimmten Knoten in einem Cluster zu beheben, können Sie auch knotenspezifische Endpunkte verwenden, diese sind jedoch für den normalen Gebrauch nicht erforderlich.

Um den Endpoint eines Clusters zu finden, gehen Sie wie folgt vor:

- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(AWSCLI\)](#)
- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(MemoryDB-API\)](#)

Informationen zum Herstellen einer Verbindung zu Knoten oder Clustern finden Sie unter [Mit Redis-Client eine Verbindung zu MemoryDB-Knoten herstellen](#).

MemoryDB-Sicherheit

Die Sicherheit von MemoryDB wird auf drei Ebenen verwaltet:

- Um zu kontrollieren, wer Verwaltungsaktionen auf MemoryDB-Clustern und -Knoten ausführen kann, verwenden Sie AWS Identity and Access Management (IAM). Wenn Sie AWS mithilfe von IAM-Anmeldeinformationen eine Verbindung herstellen, muss Ihr AWS Konto über IAM-Richtlinien verfügen, die die für die Ausführung von Vorgängen erforderlichen Berechtigungen gewähren. Weitere Informationen finden Sie unter [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#)
- Um die Zugriffsebenen für Cluster zu steuern, erstellen Sie Benutzer mit bestimmten Berechtigungen und weisen sie den Access Control Lists (ACL) zu. Die ACL wiederum wird dann einem oder mehreren Clustern zugeordnet. Weitere Informationen finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).
- MemoryDB-Cluster müssen in einer Virtual Private Cloud (VPC) erstellt werden, die auf dem Amazon VPC-Service basiert. Um zu kontrollieren, welche Geräte und Amazon EC2 EC2-Instances Verbindungen zum Endpunkt und Port des Knotens für MemoryDB-Cluster in einer VPC öffnen können, verwenden Sie eine VPC-Sicherheitsgruppe. Diese Endpunkt- und Portverbindungen können mithilfe von Transport Layer Security (TLS)/Secure Sockets Layer (SSL) erstellt werden. Darüber hinaus können Firewallregeln in Ihrem Unternehmen steuern, ob Geräte, die in Ihrem Unternehmen laufen, Verbindungen zu einem MemoryDB-Cluster öffnen können. Weitere Informationen zu VPCs finden Sie unter [MemoryDB und Amazon VPC](#).

Weitere Informationen zur Sicherheitskonfiguration finden Sie unter [Sicherheit in MemoryDB for Redis](#).

Erste Schritte mit MemoryDB

Diese Übung führt Sie durch die Schritte, mit denen Sie mithilfe der MemoryDB Management Console einen MemoryDB-Cluster erstellen, Zugriff darauf gewähren, eine Verbindung zu ihm herstellen und ihn schließlich löschen.

Note

Für die Zwecke dieser Übung empfehlen wir, beim Erstellen eines Clusters die Option Einfache Erstellung zu verwenden und zu den anderen beiden Optionen zurückzukehren, sobald Sie sich mit den Funktionen von MemoryDB näher befasst haben.

Themen

- [Einrichtung](#)
- [Schritt 1: Erstellen eines Clusters](#)
- [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#)
- [Schritt 3: Connect zum Cluster herstellen](#)
- [Schritt 4: Löschen eines Clusters](#)
- [Wie geht es weiter?](#)

Einrichtung

Im Folgenden finden Sie Themen, in denen die einmaligen Aktionen beschrieben werden, die Sie ergreifen müssen, um mit der Nutzung von MemoryDB zu beginnen.

Themen

- [Erstellen Sie Ihr Konto AWS](#)
- [Erteilen programmgesteuerten Zugriffs](#)
- [Richten Sie Ihre Berechtigungen ein \(nur für neue MemoryDB-Benutzer\)](#)
- [AWS CLI herunterladen und konfigurieren](#)

Erstellen Sie Ihr Konto AWS

Melde dich an für ein AWS-Konto

Wenn Sie noch keine haben AWS-Konto, führen Sie die folgenden Schritte aus, um eine zu erstellen.

Um sich für eine anzumelden AWS-Konto

1. Öffnen Sie <https://portal.aws.amazon.com/billing/signup>.
2. Folgen Sie den Online-Anweisungen.

Bei der Anmeldung müssen Sie auch einen Telefonanruf entgegennehmen und einen Verifizierungscode über die Telefontasten eingeben.

Wenn Sie sich für eine anmelden AWS-Konto, Root-Benutzer des AWS-Kontos wird eine erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Aus Sicherheitsgründen sollten Sie einem Benutzer Administratorzugriff zuweisen und nur den Root-Benutzer verwenden, um [Aufgaben auszuführen, für die Root-Benutzerzugriff erforderlich](#) ist.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Sie können jederzeit Ihre aktuelle Kontoaktivität anzeigen und Ihr Konto verwalten. Rufen Sie dazu <https://aws.amazon.com/> auf und klicken Sie auf Mein Konto.

Erstellen Sie einen Benutzer mit Administratorzugriff

Nachdem Sie sich für einen angemeldet haben AWS-Konto, sichern Sie Ihren Root-Benutzer des AWS-Kontos AWS IAM Identity Center, aktivieren und erstellen Sie einen Administratorbenutzer, sodass Sie den Root-Benutzer nicht für alltägliche Aufgaben verwenden.

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

1. Melden Sie sich [AWS Management Console](#) als Kontoinhaber an, indem Sie Root-Benutzer auswählen und Ihre AWS-Konto E-Mail-Adresse eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

Hilfe bei der Anmeldung mit dem Root-Benutzer finden Sie unter [Anmelden als Root-Benutzer](#) im AWS-Anmeldung Benutzerhandbuch zu.

2. Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den Root-Benutzer.

Anweisungen finden Sie unter [Aktivieren eines virtuellen MFA-Geräts für Ihren AWS-Konto Root-Benutzer \(Konsole\)](#) im IAM-Benutzerhandbuch.

Erstellen Sie einen Benutzer mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

Anweisungen finden Sie unter [Aktivieren AWS IAM Identity Center](#) im AWS IAM Identity Center Benutzerhandbuch.

2. Gewähren Sie einem Benutzer in IAM Identity Center Administratorzugriff.

Ein Tutorial zur Verwendung von IAM-Identity-Center-Verzeichnis als Identitätsquelle finden [Sie unter Benutzerzugriff mit der Standardeinstellung konfigurieren IAM-Identity-Center-Verzeichnis](#) im AWS IAM Identity Center Benutzerhandbuch.

Melden Sie sich als Benutzer mit Administratorzugriff an

- Um sich mit Ihrem IAM-Identity-Center-Benutzer anzumelden, verwenden Sie die Anmelde-URL, die an Ihre E-Mail-Adresse gesendet wurde, als Sie den IAM-Identity-Center-Benutzer erstellt haben.

Hilfe bei der Anmeldung mit einem IAM Identity Center-Benutzer finden Sie [im AWS-Anmeldung Benutzerhandbuch unter Anmeldung beim AWS Zugriffsportal](#).

Weisen Sie weiteren Benutzern Zugriff zu

1. Erstellen Sie in IAM Identity Center einen Berechtigungssatz, der der bewährten Methode zur Anwendung von Berechtigungen mit den geringsten Rechten folgt.

Anweisungen finden Sie im Benutzerhandbuch unter [Einen Berechtigungssatz erstellen](#).AWS IAM Identity Center

2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

Anweisungen finden [Sie im AWS IAM Identity Center Benutzerhandbuch unter Gruppen hinzufügen](#).

Erteilen programmgesteuerten Zugriffs

Benutzer benötigen programmatischen Zugriff, wenn sie mit AWS außerhalb des interagieren möchten. AWS Management Console Die Art und Weise, wie programmatischer Zugriff gewährt wird, hängt vom Benutzertyp ab, der zugreift. AWS

Um Benutzern programmgesteuerten Zugriff zu gewähren, wählen Sie eine der folgenden Optionen.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
Mitarbeiteridentität (Benutzer, die in IAM Identity Center verwaltet werden)	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder APIs zu signieren. AWS	<p>Befolgen Sie die Anweisungen für die Schnittstelle, die Sie verwenden möchten.</p> <ul style="list-style-type: none"> Informationen zu den AWS CLI finden Sie unter Konfiguration der AWS CLI zu AWS IAM Identity Center verwendenden im AWS Command Line Interface Benutzerhandbuch. Informationen zu AWS SDKs, Tools und AWS APIs finden Sie unter IAM Identity Center-Authentifizierung im Referenzhandbuch für AWS SDKs und Tools.
IAM	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder APIs zu signieren. AWS	Folgen Sie den Anweisungen unter Verwenden temporärer Anmeldeinformationen mit AWS Ressourcen im IAM-Benutzerhandbuch.
IAM	(Nicht empfohlen) Verwenden Sie langfristige Anmeldeinformationen, um	Befolgen Sie die Anweisungen für die Schnittstelle, die Sie verwenden möchten.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
	<p>programmatische Anfragen an die AWS CLI, AWS SDKs oder APIs zu signieren. AWS</p>	<ul style="list-style-type: none"> • Informationen dazu finden Sie unter Authentifizierung mithilfe von IAM-Benutzeranmeldedaten im Benutzerhandbuch. AWS CLI AWS Command Line Interface • Informationen zu AWS SDKs und Tools finden Sie unter Authentifizieren mit langfristigen Anmeldeinformationen im Referenzhandbuch für AWS SDKs und Tools. • Informationen zu AWS APIs finden Sie unter Verwaltung von Zugriffsschlüsseln für IAM-Benutzer im IAM-Benutzerhandbuch.

Verwandte Themen:

- [Was ist IAM](#) im IAM-Benutzerhandbuch.
- AWS Allgemeine Referenz zu [Sicherheitsanmeldedaten](#).

Richten Sie Ihre Berechtigungen ein (nur für neue MemoryDB-Benutzer)

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in: AWS IAM Identity Center

Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter [Erstellen eines Berechtigungssatzes](#) im AWS IAM Identity Center -Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter [Erstellen einer Rolle für einen externen Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch.

- IAM-Benutzer:
 - Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter [Erstellen einer Rolle für einen IAM-Benutzer](#) im IAM-Benutzerhandbuch.
 - (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter [Hinzufügen von Berechtigungen zu einem Benutzer \(Konsole\)](#) im IAM-Benutzerhandbuch.

MemoryDB for Redis erstellt und verwendet dienstbezogene Rollen, um in Ihrem Namen Ressourcen bereitzustellen und auf andere AWS Ressourcen und Dienste zuzugreifen. Damit MemoryDB eine dienstbezogene Rolle für Sie erstellt, verwenden Sie die verwaltete Richtlinie mit dem Namen `AWSAmazonMemoryDBFullAccess`. Diese Rolle besitzt die vordefinierte Berechtigung, die der Service benötigt, um eine Service-verknüpfte Rolle für Sie zu erstellen.

Sie können sich entscheiden, anstelle der Standardrichtlinie eine benutzerseitig verwaltete Richtlinie zu verwenden. Stellen Sie in diesem Fall sicher, dass Sie entweder über die erforderlichen Aufrufberechtigungen verfügen `iam:createServiceLinkedRole` oder dass Sie die serviceverknüpfte MemoryDB-Rolle erstellt haben.

Weitere Informationen finden Sie hier:

- [Erstellen einer neuen Richtlinie](#)(IAM)
- [AWS-verwaltete \(vordefinierte\) Richtlinien für MemoryDB for Redis](#)
- [Verwenden von serviceverknüpften Rollen für Amazon MemoryDB for Redis](#)

AWS CLI herunterladen und konfigurieren

Das AWS CLI ist unter <http://aws.amazon.com/cli> verfügbar. Sie läuft unter Windows, MacOS und Linux. Gehen Sie nach dem Herunterladen wie folgt vor AWS CLI, um es zu installieren und zu konfigurieren:

1. Gehen Sie zum [AWS -Benutzerhandbuch zu Command Line Interface](#).
2. Folgen Sie den Anweisungen zur [Installation der AWS CLI](#) und [zur Konfiguration der AWS CLI](#).

Schritt 1: Erstellen eines Clusters

Bevor Sie einen Cluster für die Produktion erstellen, müssen Sie natürlich überlegen, wie Sie den Cluster entsprechend Ihren geschäftlichen Anforderungen konfigurieren. Diese Probleme werden im [Vorbereitung eines Clusters](#)-Abschnitt adressiert. Für die Zwecke dieser Übung „Erste Schritte“ können Sie die Standardkonfigurationswerte akzeptieren, sofern sie zutreffen.

Der Cluster, den Sie gleich starten werden, wird live sein und nicht in einer Sandbox ausgeführt. Es fallen die standardmäßigen MemoryDB-Nutzungsgebühren für die Instance an, bis Sie sie löschen. Die Gesamtkosten werden minimal sein (meistens geringer als ein Dollar), wenn Sie diese Übung in einer Sitzung durchlaufen und den Cluster löschen, sobald Sie die Übung abgeschlossen haben. [Weitere Informationen zu den MemoryDB-Nutzungsraten finden Sie unter MemoryDB.](#)

Ihr Cluster wird, basierend auf dem Amazon-VPC-Service, in einer Virtual Private Cloud (VPC) gestartet.

Einen MemoryDB-Cluster erstellen

Die folgenden Beispiele zeigen, wie ein Cluster mithilfe der MemoryDB-API AWS CLI und der AWS Management Console MemoryDB-API erstellt wird.

Einen Cluster erstellen (Konsole)

Um einen Cluster mit der MemoryDB-Konsole zu erstellen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Clusters und dann Create aus.

Easy create

1. Füllen Sie den Bereich Configuration (Konfiguration) aus. Dadurch werden der Knotentyp und die Standardkonfiguration Ihres Clusters konfiguriert. Wählen Sie aus den folgenden Optionen die entsprechende Speichergröße und Netzwerkleistung aus, die Sie benötigen:
 - Produktion
 - Entwicklung/Test
 - Demo

2. Füllen Sie den Abschnitt Cluster-Informationen aus.
 - a. Geben Sie in das Feld Name einen Namen für Ihr Cluster ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
- Er muss mit einem Buchstaben beginnen.
- Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
- Er darf nicht mit einem Bindestrich enden.

- b. Im Feld Description (Beschreibung) geben Sie eine Beschreibung für den Cluster ein.

3. Füllen Sie den Abschnitt Subnetzgruppen aus:

- Erstellen Sie für Subnetzgruppen eine neue Subnetzgruppe oder wählen Sie eine vorhandene aus der verfügbaren Liste aus, die Sie auf diesen Cluster anwenden möchten. Wenn Sie eine neue erstellen:
 - Geben Sie einen Namen ein
 - Geben Sie eine Beschreibung ein
 - Wenn Sie Multi-AZ aktiviert haben, muss die Subnetzgruppe mindestens zwei Subnetze enthalten, die sich in verschiedenen Availability Zones befinden. Weitere Informationen finden Sie unter [Subnetze und Subnetzgruppen](#).
 - Wenn Sie eine neue Subnetzgruppe erstellen und noch keine VPC haben, werden Sie aufgefordert, eine VPC zu erstellen. Weitere Informationen finden Sie unter [Was ist Amazon VPC?](#) im Amazon VPC-Benutzerhandbuch.

4. Für die Vektorsuche können Sie die Vektorsuchfunktion aktivieren, um Vektoreinbettungen zu speichern und Vektorsuchen durchzuführen. Beachten Sie, dass dadurch die Werte für Redis-Versionskompatibilität, Parametergruppen und Shards korrigiert werden. Weitere Informationen finden Sie unter [Vektor-Suche](#).

5. Standardeinstellungen anzeigen:

Bei Verwendung von Easy Create sind die übrigen Clustereinstellungen standardmäßig festgelegt. Beachten Sie, dass einige dieser Einstellungen nach der Erstellung geändert werden können, wie unter Nach der Erstellung bearbeitbar angegeben.

6. Bei Tags können Sie optional Tags anwenden, um Ihre Cluster zu durchsuchen und zu filtern oder Ihre AWS Kosten zu verfolgen.

- Überprüfen Sie alle Ihre Einträge und ausgewählten Optionen und machen Sie dann evtl. erforderliche Korrekturen. Wenn Sie bereit sind, wählen Sie Erstellen, um Ihren Cluster zu starten, oder Abbrechen, um den Vorgang abzubrechen.

Sobald als Status des Clusters available erscheint, können Sie EC2 Zugriff darauf erteilen, eine Verbindung mit ihm herstellen und ihn verwenden. Weitere Informationen finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#).

⚠ Important

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Create new cluster

- Füllen Sie den Abschnitt Cluster-Informationen aus.
 - Geben Sie in das Feld Name einen Namen für Ihr Cluster ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

 - Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
 - Im Feld Description (Beschreibung) geben Sie eine Beschreibung für den Cluster ein.
- Füllen Sie den Abschnitt Subnetzgruppen aus:
 - Erstellen Sie für Subnetzgruppen eine neue Subnetzgruppe oder wählen Sie eine vorhandene aus der verfügbaren Liste aus, die Sie auf diesen Cluster anwenden möchten. Wenn Sie eine neue erstellen:
 - Geben Sie einen Namen ein
 - Geben Sie eine Beschreibung ein

- Wenn Sie Multi-AZ aktiviert haben, muss die Subnetzgruppe mindestens zwei Subnetze enthalten, die sich in verschiedenen Availability Zones befinden. Weitere Informationen finden Sie unter [Subnetze und Subnetzgruppen](#).
- Wenn Sie eine neue Subnetzgruppe erstellen und noch keine VPC haben, werden Sie aufgefordert, eine VPC zu erstellen. Weitere Informationen finden Sie unter [Was ist Amazon VPC?](#) im Amazon VPC-Benutzerhandbuch.

3. Füllen Sie den Abschnitt Clustereinstellungen aus:

- a. Für die Funktion „Vektorsuche aktivieren“ können Sie diese Option aktivieren, um Vektoreinbettungen zu speichern und Vektorsuchen durchzuführen. Beachten Sie, dass dadurch die Werte für Redis-Versionskompatibilität, Parametergruppen und Shards korrigiert werden. Weitere Informationen finden Sie unter [Vektor-Suche](#).
- b. Akzeptieren Sie aus Gründen der Redis-Versionskompatibilität die Standardeinstellung. 6.2
- c. Akzeptieren Sie für Port den Redis-Standardport 6379 oder geben Sie die Portnummer ein, falls Sie einen anderen Port verwenden möchten.
- d. Wenn Sie die Vektorsuche aktiviert haben, verwenden Sie für Parametergruppe. `default.memorydb-redis7.search.preview` Andernfalls akzeptieren Sie die `default.memorydb-redis7` Parametergruppe.

Parametergruppen steuern die Laufzeitparameter Ihres Clusters. Weitere Informationen zu Parametergruppen finden Sie unter [Redis-spezifische Parameter](#).

- e. Wählen Sie unter Knotentyp einen Wert für den gewünschten Knotentyp (zusammen mit der zugehörigen Speichergröße) aus.

Wenn Sie einen Knotentyp aus der R6gd-Familie wählen, wird Daten-Tiering automatisch aktiviert. Hierbei wird der Datenspeicher zwischen Speicher und SSD aufteilt. Weitere Informationen finden Sie unter [Daten-Tiering](#).

- f. Wählen Sie unter Anzahl der Shards die Anzahl der Shards aus, die Sie für diesen Cluster benötigen. Für eine höhere Verfügbarkeit Ihrer Cluster empfehlen wir, dass Sie mindestens 2 Shards hinzufügen.

Sie können die Anzahl der Shards in Ihrem Cluster dynamisch ändern. Weitere Informationen finden Sie unter [Skalieren von MemoryDB-Clustern](#).

- g. Wählen Sie für Replicas per shard (Replikate pro Shard) die Anzahl der Read Replica-Knoten aus, die sich in jedem Shard befinden sollen.

Es bestehen die folgenden Einschränkungen:

- Wenn Sie Multi-AZ aktiviert haben, stellen Sie sicher, dass mindestens ein Replikat pro Shard vorhanden ist.
 - Die Anzahl der Replikate ist für jeden Shard gleich, wenn der Cluster mithilfe der Konsole erstellt wird.
- h. Wählen Sie Weiter
 - i. Füllen Sie den Abschnitt Erweiterte Einstellungen aus:
 - i. Wählen Sie für Security groups (Sicherheitsgruppen) die gewünschten Sicherheitsgruppen für diesen Cluster aus. Eine security group (Sicherheitsgruppe) fungiert als Firewall, um den Netzwerkzugriff auf Ihren Cluster zu steuern. Sie können die Standardsicherheitsgruppe für Ihre VPC verwenden oder eine neue erstellen.

Weitere Informationen zu Sicherheitsgruppen finden Sie unter [Sicherheitsgruppen für Ihre VPC](#) im Benutzerhandbuch zu Amazon VPC.

- ii. Zur Verschlüsselung Ihrer Daten haben Sie die folgenden Optionen:
 - Verschlüsselung im Ruhezustand – Ermöglicht die Verschlüsselung von Daten, die auf der Festplatte gespeichert sind. Weitere Informationen finden Sie unter [Verschlüsselung im Ruhezustand](#).

 Note

Sie haben die Möglichkeit, einen anderen Verschlüsselungsschlüssel als den Standardschlüssel anzugeben, indem Sie den vom Kunden verwalteten AWS KMS-Schlüssel wählen und dann den Schlüssel auswählen.

- Verschlüsselung während der Übertragung – Ermöglicht die Verschlüsselung von Daten während der Übertragung. Wenn Sie keine Verschlüsselung auswählen, wird eine offene Zugriffskontrollliste namens „Open Access“ mit einem Standardbenutzer erstellt. Weitere Informationen finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).

- iii. Geben Sie für Snapshot optional einen Aufbewahrungszeitraum für Snapshots und ein Snapshot-Fenster an. Standardmäßig ist Automatische Snapshots aktivieren vorausgewählt.
- iv. Geben Sie für das Wartungsfenster optional ein Wartungsfenster an. Das Wartungsfenster ist die Zeit, in der Regel eine Stunde, jede Woche, zu der MemoryDB die Systemwartung für Ihren Cluster plant. Sie können MemoryDB erlauben, den Tag und die Uhrzeit für Ihr Wartungsfenster auszuwählen (keine Präferenz), oder Sie können Tag, Uhrzeit und Dauer selbst wählen (Wartungsfenster angeben). Treffen Sie bei Wahl von Specify maintenance window eine Auswahl in den Listen Start day, Start time und Duration (in Stunden) für Ihr Wartungsfenster. Alle Uhrzeiten sind in UCT angegeben.

Weitere Informationen finden Sie unter [Verwaltung der Wartung](#).
- v. Wählen Sie für Benachrichtigungen ein bestehendes Amazon Simple Notification Service (Amazon SNS)-Thema oder wählen Sie Manuelle ARN-Eingabe und geben Sie den Amazon-Ressourcennamen (ARN) des Themas ein. Mit Amazon SNS können Sie Push-Benachrichtigungen an mit dem Internet verbundene Smart-Geräte senden. Standardmäßig sind Benachrichtigungen deaktiviert. [Weitere Informationen finden Sie unter <https://aws.amazon.com/sns/>](#).
- vi. Bei Tags können Sie optional Tags anwenden, um Ihre Cluster zu durchsuchen und zu filtern oder Ihre AWS Kosten nachzuverfolgen.
- j. Überprüfen Sie alle Ihre Einträge und ausgewählten Optionen und machen Sie dann evtl. erforderliche Korrekturen. Wenn Sie bereit sind, wählen Sie Erstellen, um Ihren Cluster zu starten, oder Abbrechen, um den Vorgang abubrechen.

Sobald als Status des Clusters available erscheint, können Sie EC2 Zugriff darauf erteilen, eine Verbindung mit ihm herstellen und ihn verwenden. Weitere Informationen finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#).

 **Important**

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Restore from snapshots

Wählen Sie unter Snapshot-Quelle den Quell-Snapshot aus, aus dem Daten migriert werden sollen. Weitere Informationen finden Sie unter [Snapshot und Wiederherstellung](#).

Note

Wenn Sie möchten, dass in Ihrem neuen Cluster die Vektorsuche aktiviert ist, muss für den Quell-Snapshot auch die Vektorsuche aktiviert sein.

Der Zielcluster verwendet standardmäßig die Einstellungen des Quell-Clusters. Optional können Sie die folgenden Einstellungen auf dem Zielcluster ändern:

1. Cluster-Informationen

- a. Geben Sie in das Feld Name einen Namen für Ihr Cluster ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
- Er muss mit einem Buchstaben beginnen.
- Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
- Er darf nicht mit einem Bindestrich enden.

- b. Im Feld Description (Beschreibung) geben Sie eine Beschreibung für den Cluster ein.

2. Subnetzgruppen

- Erstellen Sie für Subnetzgruppen eine neue Subnetzgruppe oder wählen Sie eine vorhandene aus der verfügbaren Liste aus, die Sie auf diesen Cluster anwenden möchten. Wenn Sie eine neue erstellen:
 - Geben Sie einen Namen ein
 - Geben Sie eine Beschreibung ein
 - Wenn Sie Multi-AZ aktiviert haben, muss die Subnetzgruppe mindestens zwei Subnetze enthalten, die sich in verschiedenen Availability Zones befinden. Weitere Informationen finden Sie unter [Subnetze und Subnetzgruppen](#).

- Wenn Sie eine neue Subnetzgruppe erstellen und noch keine VPC haben, werden Sie aufgefordert, eine VPC zu erstellen. Weitere Informationen finden Sie unter [Was ist Amazon VPC?](#) im Amazon VPC-Benutzerhandbuch.

3. Cluster-Einstellungen

- Für die Funktion „Vektorsuche aktivieren“ können Sie diese Option aktivieren, um Vektoreinbettungen zu speichern und Vektorsuchen durchzuführen. Beachten Sie, dass dadurch die Werte für Redis-Versionskompatibilität, Parametergruppen und Shards korrigiert werden. Weitere Informationen finden Sie unter [Vektor-Suche](#).
- Akzeptieren Sie aus Gründen der Redis-Versionskompatibilität die Standardeinstellung. 6.2
- Akzeptieren Sie für Port den Redis-Standardport 6379 oder geben Sie die Portnummer ein, falls Sie einen anderen Port verwenden möchten.
- Wenn Sie die Vektorsuche aktiviert haben, verwenden Sie für Parametergruppe. `default.memorydb-redis7.search.preview` Andernfalls akzeptieren Sie die `default.memorydb-redis7` Parametergruppe.

Parametergruppen steuern die Laufzeitparameter Ihres Clusters. Weitere Informationen zu Parametergruppen finden Sie unter [Redis-spezifische Parameter](#).

- Wählen Sie unter Knotentyp einen Wert für den gewünschten Knotentyp (zusammen mit der zugehörigen Speichergröße) aus.

Wenn Sie einen Knotentyp aus der R6gd-Familie wählen, wird Daten-Tiering automatisch aktiviert. Hierbei wird der Datenspeicher zwischen Speicher und SSD aufteilt. Weitere Informationen finden Sie unter [Daten-Tiering](#).

- Wählen Sie unter Anzahl der Shards die Anzahl der Shards aus, die Sie für diesen Cluster benötigen. Für eine höhere Verfügbarkeit Ihrer Cluster empfehlen wir, dass Sie mindestens 2 Shards hinzufügen.

Sie können die Anzahl der Shards in Ihrem Cluster dynamisch ändern. Weitere Informationen finden Sie unter [Skalieren von MemoryDB-Clustern](#).

- Wählen Sie für Replicas per shard (Replikate pro Shard) die Anzahl der Read Replica-Knoten aus, die sich in jedem Shard befinden sollen.

Es bestehen die folgenden Einschränkungen:

- Wenn Sie Multi-AZ aktiviert haben, stellen Sie sicher, dass mindestens ein Replikat pro Shard vorhanden ist.
 - Die Anzahl der Replikate ist für jeden Shard gleich, wenn der Cluster mithilfe der Konsole erstellt wird.
- h. Wählen Sie Weiter
- i. Erweiterte Einstellungen
- i. Wählen Sie für Security groups (Sicherheitsgruppen) die gewünschten Sicherheitsgruppen für diesen Cluster aus. Eine security group (Sicherheitsgruppe) fungiert als Firewall, um den Netzwerkzugriff auf Ihren Cluster zu steuern. Sie können die Standardsicherheitsgruppe für Ihre VPC verwenden oder eine neue erstellen.

Weitere Informationen zu Sicherheitsgruppen finden Sie unter [Sicherheitsgruppen für Ihre VPC](#) im Benutzerhandbuch zu Amazon VPC.

- ii. Zur Verschlüsselung Ihrer Daten haben Sie die folgenden Optionen:
- Verschlüsselung im Ruhezustand – Ermöglicht die Verschlüsselung von Daten, die auf der Festplatte gespeichert sind. Weitere Informationen finden Sie unter [Verschlüsselung im Ruhezustand](#).

 Note

Sie haben die Möglichkeit, einen anderen Verschlüsselungsschlüssel als den Standardschlüssel anzugeben, indem Sie den vom Kunden verwalteten AWS KMS-Schlüssel wählen und dann den Schlüssel auswählen.

- Verschlüsselung während der Übertragung – Ermöglicht die Verschlüsselung von Daten während der Übertragung. Wenn Sie keine Verschlüsselung auswählen, wird eine offene Zugriffskontrollliste namens „Open Access“ mit einem Standardbenutzer erstellt. Weitere Informationen finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).
- iii. Geben Sie für Snapshot optional einen Aufbewahrungszeitraum für Snapshots und ein Snapshot-Fenster an. Standardmäßig ist Automatische Snapshots aktivieren vorausgewählt.

- iv. Geben Sie für das Wartungsfenster optional ein Wartungsfenster an. Das Wartungsfenster ist die Zeit, in der Regel eine Stunde, jede Woche, zu der MemoryDB die Systemwartung für Ihren Cluster plant. Sie können MemoryDB erlauben, den Tag und die Uhrzeit für Ihr Wartungsfenster auszuwählen (keine Präferenz), oder Sie können Tag, Uhrzeit und Dauer selbst wählen (Wartungsfenster angeben). Treffen Sie bei Wahl von Specify maintenance window eine Auswahl in den Listen Start day, Start time und Duration (in Stunden) für Ihr Wartungsfenster. Alle Uhrzeiten sind in UCT angegeben.

Weitere Informationen finden Sie unter [Verwaltung der Wartung](#).

- v. Wählen Sie für Benachrichtigungen ein bestehendes Amazon Simple Notification Service (Amazon SNS)-Thema oder wählen Sie Manuelle ARN-Eingabe und geben Sie den Amazon-Ressourcennamen (ARN) des Themas ein. Mit Amazon SNS können Sie Push-Benachrichtigungen an mit dem Internet verbundene Smart-Geräte senden. Standardmäßig sind Benachrichtigungen deaktiviert. [Weitere Informationen finden Sie unter https://aws.amazon.com/sns/](#).
- vi. Bei Tags können Sie optional Tags anwenden, um Ihre Cluster zu durchsuchen und zu filtern oder Ihre AWS Kosten nachzuverfolgen.
- j. Überprüfen Sie alle Ihre Einträge und ausgewählten Optionen und machen Sie dann evtl. erforderliche Korrekturen. Wenn Sie bereit sind, wählen Sie Erstellen, um Ihren Cluster zu starten, oder Abbrechen, um den Vorgang abubrechen.

Sobald als Status des Clusters available erscheint, können Sie EC2 Zugriff darauf erteilen, eine Verbindung mit ihm herstellen und ihn verwenden. Weitere Informationen finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#).

 **Important**

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Einen Cluster erstellen (AWS CLI)

Informationen zum Erstellen eines Clusters mit dem AWS CLI finden Sie unter [create-cluster](#). Im Folgenden wird ein Beispiel gezeigt:

Für Linux, macOS oder Unix:

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

Für Windows:

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

Sie sollten die folgende JSON-Antwort erhalten:

```
{  
  "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:xxxxxxxxxxxx:cluster/my-cluster",  
    "SnapshotRetentionLimit": 0,  
    "MaintenanceWindow": "wed:03:00-wed:04:00",  
  }  
}
```

```
"SnapshotWindow": "04:30-05:30",
"ACLName": "my-acl",
"DataTiering": "false",
"AutoMinorVersionUpgrade": true
}
}
```

Sie können mit der Verwendung des Clusters beginnen, sobald sich sein Status auf geändert hat `available`.

Important

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Einen Cluster erstellen (MemoryDB-API)

Verwenden Sie die Aktion, um einen Cluster mithilfe der MemoryDB-API zu erstellen. [CreateCluster](#)

Important

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Authentifizierung einrichten

Informationen zum Einrichten der Authentifizierung für Ihren Cluster finden Sie unter [Authentifizieren mit IAM](#) und [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).

Schritt 2: Autorisieren Sie den Zugriff auf den Cluster

In diesem Abschnitt wird davon ausgegangen, dass Sie mit dem Starten und dem Herstellen einer Verbindung zu Amazon-EC2-Instances vertraut sind. Weitere Informationen dazu finden Sie im [Amazon-EC2-Erste-Schritte-Leitfaden](#).

MemoryDB-Cluster sind für den Zugriff von einer Amazon EC2 EC2-Instance aus konzipiert. Sie können auch von containerisierten oder serverlosen Anwendungen abgerufen werden, die in Amazon Elastic Container Service oder ausgeführt werden. AWS Lambda Das gängigste Szenario ist der Zugriff auf einen MemoryDB-Cluster von einer Amazon EC2 EC2-Instance in derselben Amazon Virtual Private Cloud (Amazon VPC), was bei dieser Übung der Fall sein wird.

Bevor Sie von einer EC2-Instance eine Verbindung zu einem Cluster herstellen können, müssen Sie der EC2-Instance Zugriff auf den Cluster gewähren.

Der gängigste Anwendungsfall ist, wenn eine auf einer EC2-Instance bereitgestellte Anwendung eine Verbindung mit einem Cluster in der gleichen VPC herstellen muss. Nachfolgend finden Sie den leichtesten Weg für die Verwaltung des Zugriffs zwischen EC2-Instances und DB-Instances in derselben VPC:

1. Erstellen Sie eine VPC-Sicherheitsgruppe für Ihren Cluster. Diese Sicherheitsgruppe kann verwendet werden, um den Zugriff auf die Cluster einzuschränken. Sie können für diese Sicherheitsgruppe beispielsweise eine benutzerdefinierte Regel erstellen, die TCP-Zugriff über den Port, den Sie dem Cluster bei seiner Erstellung zugeordnet haben, und eine IP-Adresse gewährt, mit der Sie auf den Cluster zugreifen.

Der Standardport für MemoryDB-Cluster ist. 6379

2. Erstellen Sie eine VPC-Sicherheitsgruppe für Ihre EC2-Instances (Web- und Anwendungsserver). Mithilfe dieser Sicherheitsgruppe können Sie bei Bedarf den Internetzugriff auf die EC2-Instance über die Routing-Tabelle der VPC zulassen. Sie können beispielsweise Regeln für diese Sicherheitsgruppe festlegen, damit der TCP-Zugriff auf die EC2-Instance über Port 22 möglich ist.
3. Erstellen Sie in der Sicherheitsgruppe für Ihren Cluster benutzerdefinierte Regeln, die Verbindungen von der Sicherheitsgruppe aus zulassen, die Sie für Ihre EC2-Instances erstellt haben. Damit wird jedem Mitglied der Sicherheitsgruppe der Zugriff auf die DB-Instances gestattet.

So erstellen Sie eine Regel in einer VPC-Sicherheitsgruppe, die Verbindungen über eine andere Sicherheitsgruppe zulässt

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon VPC-Konsole unter <https://console.aws.amazon.com/vpc>.
2. Klicken Sie im linken Navigationsbereich auf Security Groups.
3. Wählen oder erstellen Sie eine Sicherheitsgruppe, die Sie für Ihre Cluster verwenden werden. Wählen Sie unter Inbound Rules (Eingangsregeln) die Option Edit Inbound Rules (Eingangsregeln bearbeiten) und dann Add Rule (Regeln hinzufügen). Diese Sicherheitsgruppe gewährt Mitgliedern einer anderen Sicherheitsgruppe Zugriff.
4. Wählen Sie für Type die Option Custom TCP Rule aus.
 - a. Geben Sie für Port Range den Port an, den Sie beim Erstellen des Clusters verwendet haben.

Der Standardport für MemoryDB-Cluster ist. 6379
 - b. Geben Sie in das Feld Source die ersten Zeichen der ID der Sicherheitsgruppe ein. Wählen Sie aus der Liste die Sicherheitsgruppe aus, die Sie für Ihre Amazon-EC2-Instances verwenden werden.
5. Wählen Sie Save, wenn Sie fertig sind.

Sobald Sie den Zugriff aktiviert haben, können Sie nun, wie im nächsten Abschnitt beschrieben, eine Verbindung zum Cluster herstellen.

Informationen zum Zugriff auf Ihren MemoryDB-Cluster von einer anderen Amazon VPC, einer anderen AWS Region oder sogar Ihrem Unternehmensnetzwerk aus finden Sie im Folgenden:

- [Zugriffsmuster zum Zugreifen auf einen MemoryDB-Cluster in einer Amazon VPC](#)
- [Zugriff auf MemoryDB-Ressourcen von außerhalbAWS](#)

Schritt 3: Connect zum Cluster herstellen

Bevor Sie fortfahren, müssen Sie [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#) abschließen.

In diesem Abschnitt wird davon ausgegangen, dass Sie eine Amazon-EC2-Instance erstellt haben und eine Verbindung mit ihr möglich ist. Anweisungen dazu finden Sie im [Amazon-EC2-Erste-Schritte-Leitfaden](#).

Eine Amazon EC2 EC2-Instance kann nur dann eine Verbindung zu einem Cluster herstellen, wenn Sie sie dazu autorisiert haben.

Finden Sie Ihren Cluster-Endpunkt

Sobald der Cluster den Status verfügbar hat und Sie autorisierten Zugriff haben, können Sie sich bei einer Amazon-EC2-Instance anmelden und eine Verbindung zum Cluster herstellen. Hierzu müssen Sie zuerst den Endpunkt bestimmen.

Weitere Informationen zum Auffinden Ihrer Endgeräte finden Sie im Folgenden:

- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(AWS Management Console\)](#)
- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(AWSCLI\)](#)
- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(MemoryDB-API\)](#)

Stellen Sie eine Connect zu einem MemoryDB-Cluster her (Linux)

Da Sie nun den benötigten Endpunkt haben, können Sie sich bei einer EC2-Instance anmelden und eine Verbindung zum Cluster herstellen. Im folgenden Beispiel verwenden Sie das CLI-Hilfsprogramm, um mithilfe von Ubuntu 22 eine Verbindung zu einem Cluster herzustellen. Die neueste Version von CLI unterstützt auch SSL/TLS für die Verbindung von Clustern mit aktivierter Verschlüsselung/Authentifizierung.

Mit Redis-Cli eine Verbindung zu MemoryDB-Knoten herstellen

Für den Zugriff auf Daten von MemoryDB-Knoten verwenden Sie Clients, die mit Secure Socket Layer (SSL) arbeiten. Sie können redis-cli auch mit TLS/SSL in Amazon Linux und Amazon Linux 2 verwenden.

So verwenden Sie `redis-cli`, um eine Verbindung zu einem MemoryDB-Cluster auf Amazon Linux 2 oder Amazon Linux herzustellen

1. Laden Sie das `redis-cli`-Dienstprogramm herunter und kompilieren Sie es. Dieses Dienstprogramm ist Teil der Redis-Softwaredistribution.
2. Geben Sie an der Eingabeaufforderung Ihrer EC2-Instance die entsprechenden Befehle für die von Ihnen verwendete Linux-Version ein.

Amazon Linux 2023

Wenn Sie Amazon Linux 2023 verwenden, geben Sie Folgendes ein:

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

Geben Sie dann den folgenden Befehl ein und ersetzen Sie den in diesem Beispiel gezeigten Befehl durch den Endpunkt Ihres Clusters und den Port.

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

Weitere Informationen zum Auffinden des Endpunkts finden Sie unter [Finden Sie Ihre Knotenendpunkte](#).

Amazon Linux 2

Wenn Sie Amazon Linux 2 verwenden, geben Sie Folgendes ein:

```
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

Wenn Sie Amazon Linux verwenden, geben Sie Folgendes ein:

```
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/
```

Unter Amazon Linux müssen Sie möglicherweise auch die folgenden zusätzlichen Schritte ausführen:

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

3. Nachdem Sie das Hilfsprogramm `redis-cli` heruntergeladen und installiert haben, wird empfohlen, den optionalen Befehl auszuführen. `make-test`
4. Um eine Verbindung zu einem Cluster mit aktivierter Verschlüsselung und Authentifizierung herzustellen, geben Sie diesen Befehl ein:

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

Note

Wenn Sie `redis6` auf Amazon Linux 2023 installieren, können Sie jetzt den folgenden Befehl verwenden `redis6-cli`: `redis-cli`

```
redis6-cli -h Primary or Configuration Endpoint --tls -p 6379
```

Schritt 4: Löschen eines Clusters

Solange sich ein Cluster im Zustand `available` befindet, werden Ihnen dafür Gebühren berechnet. Dabei spielt es keine Rolle, ob Sie ihn aktiv nutzen oder ob nicht. Löschen Sie den Cluster, damit keine Gebühren mehr anfallen.

Warning

Wenn Sie einen MemoryDB-Cluster löschen, bleiben Ihre manuellen Snapshots erhalten. Sie können auch einen letzten Snapshot erstellen, bevor der Cluster gelöscht wird. Automatische

Snapshots werden nicht aufbewahrt. Weitere Informationen finden Sie unter [Snapshot und Wiederherstellung](#).

Mit dem AWS Management Console

Mit dem folgenden Verfahren wird ein einzelner Cluster aus Ihrer Bereitstellung gelöscht. Um mehrere Cache-Cluster zu löschen, wiederholen Sie das Verfahren für jeden Cluster, den Sie löschen möchten. Sie brauchen nicht zu warten, bis ein Cluster fertig gelöscht ist, bevor Sie den Vorgang zum Löschen eines anderen Clusters starten.

Löschen eines Clusters

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Um den zu löschenden Cluster auszuwählen, klicken Sie in der Clusterliste auf das Optionsfeld neben dem Namen des Clusters. In diesem Fall der Name des von Ihnen unter [Schritt 1: Erstellen eines Clusters](#) erstellten Clusters.
3. Klicken Sie bei Actions auf Delete.
4. Wählen Sie zunächst aus, ob vor dem Löschen ein Snapshot des Clusters erstellt werden soll, und geben Sie dann `delete` in das Bestätigungsfeld Löschen ein, um den Cluster zu löschen, oder wählen Sie Abbrechen, um den Cluster beizubehalten.

Wenn Sie Delete auswählen, ändert sich der Status des Clusters zu deleting.

Sobald dieser Cluster nicht mehr in der Cluster-Liste erscheint, fallen dafür keine Gebühren mehr an.

Verwenden Sie den AWS CLI

Der folgende Code löscht den Cluster `my-cluster`. In diesem Fall ersetzen Sie `my-cluster` mit dem Namen des Clusters, den Sie unter [Schritt 1: Erstellen eines Clusters](#) erstellt haben.

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

Der `delete-cluster` CLI-Vorgang löscht nur einen Cluster. Um mehrere Cluster zu löschen, rufen Sie `delete-cluster` für jeden Cluster auf, den Sie löschen möchten. Sie müssen nicht warten, bis ein Cluster vollständig gelöscht ist, bevor Sie einen anderen löschen.

Für Linux, macOS oder Unix:

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

Für Windows:

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

Weitere Informationen finden Sie unter [delete-cluster](#).

Verwenden der MemoryDB-API

Der folgende Code löscht den Cluster `my-cluster`. In diesem Fall ersetzen Sie `my-cluster` mit dem Namen des Clusters, den Sie unter [Schritt 1: Erstellen eines Clusters](#) erstellt haben.

```
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>
```

Der `DeleteCluster` API-Vorgang löscht nur einen Cluster. Um mehrere Cluster zu löschen, rufen Sie `DeleteCluster` für jeden Cluster auf, den Sie löschen möchten. Sie müssen nicht warten, bis ein Cluster vollständig gelöscht ist, bevor Sie einen anderen löschen.

Weitere Informationen finden Sie unter [DeleteCluster](#).

Wie geht es weiter?

Nachdem Sie die Übung Erste Schritte ausprobiert haben, können Sie in den folgenden Abschnitten mehr über MemoryDB und die verfügbaren Tools erfahren:

- [Erste Schritte mit AWS](#)
- [Tools für Amazon Web Services](#)
- [AWS -Befehlszeilenschnittstelle](#)
- [API-Referenz für MemoryDB für Redis.](#)

Knoten verwalten

Ein Knoten ist der kleinste Baustein einer MemoryDB-für-Redis-Bereitstellung. Ein Knoten gehört zu einem Shard, der zu einem Cluster gehört. Auf jedem Knoten wird die Engine-Version ausgeführt, die beim Erstellen oder letzten Ändern des Clusters ausgewählt wurde. Jeder Knoten besitzt einen eigenen Domain Name Service (DNS)-Namen und Port. Es werden mehrere Arten von MemoryDB-Knoten unterstützt, die jeweils unterschiedliche Mengen an zugehörigem Speicher und Rechenleistung aufweisen.

Themen

- [MemoryDB-Knoten und Shards](#)
- [Unterstützte Knotentypen](#)
- [Reservierte MemoryDB-Knoten](#)
- [Ersetzen von Knoten](#)

Einige wichtige Operationen, die Knoten umfassen, sind folgende:

- [Knoten zu einem Cluster hinzufügen/entfernen](#)
- [Skalierung](#)
- [Verbindungsendpunkte ermitteln](#)

MemoryDB-Knoten und Shards

Ein Shard ist eine hierarchische Anordnung von Knoten, die jeweils in einen Cluster verpackt sind. Shards unterstützen die Replikation. Innerhalb eines Shards dient ein Knoten als Primärknoten mit Lese-/Schreibzugriff. Alle anderen Knoten in einem Shard dienen als schreibgeschützte Replikatate des Primärknotens. MemoryDB unterstützt mehrere Shards innerhalb eines Clusters. Diese Unterstützung ermöglicht die Partitionierung Ihrer Daten in einem MemoryDB-Cluster.

MemoryDB unterstützt die Replikation über Shards. Die API-Operation [DescribeClusters](#) listet die Shards mit den Mitgliedsknoten, den Knotennamen, Endpunkten und anderen Informationen auf.

Nachdem ein MemoryDB-Cluster erstellt wurde, kann er geändert (auf- oder abskaliert) werden. Weitere Informationen finden Sie unter [Skalierung](#) und [Ersetzen von Knoten](#).

Wenn Sie einen neuen Cluster erstellen, können Sie ihn mit Daten aus dem alten Cluster bestücken, damit er nicht von Anfang an leer ist. Dies kann hilfreich sein, wenn Sie Ihren Knotentyp oder Ihre Engine-Version ändern oder von Amazon ElastiCache für Redis migrieren müssen. Weitere Informationen finden Sie unter [Manuelle Snapshots erstellen](#) und [Wiederherstellung aus einem Snapshot](#).

Unterstützte Knotentypen

MemoryDB unterstützt die folgenden Knotentypen.

RAM-optimiert

Instance-Typ	Baseline-Bandbreite (Gbit/s)	Maximale Bandbreite (Gbit/s)	Verbessertes E/A-Multiplexing (Redis 7.0.4+)	Engine-Mindestversion
db.r7g.large	0,937	12,5	Nein	6.2
db.r7g.xlarge	1.876	12,5	Nein	6.2
db.r7g.2xlarge	3,75	15	Ja	6.2
db.r7g.4xlarge	7,5	15	Ja	6.2
db.r7g.8xlarge	15	N/A	Ja	6.2
db.r7g.12xlarge	22,5	N/A	Ja	6.2
db.r7g.16xlarge	30	N/A	Ja	6.2
db.r6g.large	0.75	10.0	Nein	6.2
db.r6g.xlarge	1.25	10.0	Nein	6.2
db.r6g.2xlarge	2.5	10.0	Ja	6.2
db.r6g.4xlarge	5.0	10.0	Ja	6.2
db.r6g.8xlarge	12	N/A	Ja	6.2
db.r6g.12xlarge	20	N/A	Ja	6.2
db.r6g.16xlarge	25	N/A	Ja	6.2

Mit Daten-Tiering optimierter Speicher

Instance-Typ	Baseline-Bandbreite (Gbit/s)	Maximale Bandbreite (Gbit/s)	Verbessertes E/A-Multiplexing (Redis 7.0.4+)	Engine-Mindestversion
db.r6gd.xlarge	1.25	10	Nein	6.2
db.r6gd.2xlarge	2.5	10	Nein	6.2
db.r6gd.4xlarge	5.0	10	Nein	6.2
db.r6gd.8xlarge	12	N/A	Nein	6.2

Knoten für allgemeine Zwecke

Instance-Typ	Baseline-Bandbreite (Gbit/s)	Maximale Bandbreite (Gbit/s)	Verbessertes E/A-Multiplexing (Redis 7.0.4+)	Engine-Mindestversion
db.t4g.klein	0,128	5.0	Nein	6.2
db.t4g.medium	0,256	5.0	Nein	6.2

Informationen zur Verfügbarkeit von AWS Regionen finden Sie unter [MemoryDB für Redis – Preise](#)

Alle Knotentypen werden in einer Virtual Private Cloud (VPC) erstellt.

Reservierte MemoryDB-Knoten

Reservierte Knoten bieten Ihnen einen erheblichen discount im Vergleich zu On-Demand-Node-Preisen. Reservierte Knoten sind keine physischen Knoten, sondern ein Abrechnungsrabatt, der für die Nutzung von On-Demand-Knoten in Ihrem Konto gewährt wird. Rabatte für reservierte Knoten sind an den Knotentyp und die AWS Region gebunden.

Das allgemeine Verfahren für die Arbeit mit reservierten Knoten ist wie folgt:

- Informieren Sie sich über die verfügbaren Angebote für reservierte Knoten
- Erwerben Sie ein Angebot für reservierte Knoten mithilfe des AWS Management Console SDK AWS Command Line Interface oder
- Sehen Sie sich die Informationen zu Ihren vorhandenen reservierten Knoten an

Themen

- [Überblick über reservierte Knoten](#)

Überblick über reservierte Knoten

Wenn Sie einen reservierten MemoryDB-Node erwerben, verpflichten Sie sich, für die Dauer des reservierten Knotens einen vergünstigten Preis für einen bestimmten Knotentyp zu erhalten. Um einen reservierten MemoryDB-Node zu verwenden, erstellen Sie einen neuen Node, genau wie bei einem On-Demand-Knoten. Der neue Knoten, den Sie erstellen, muss den Spezifikationen des reservierten Knotens entsprechen. Wenn die Spezifikationen des neuen Knotens mit einem vorhandenen reservierten Knoten für Ihr Konto übereinstimmen, wird Ihnen der für den reservierten Knoten angebotene vergünstigte Tarif in Rechnung gestellt. Andernfalls wird der Knoten zu einem On-Demand-Tarif in Rechnung gestellt. Sie können die AWS Management Console, oder die MemoryDB-API verwenden AWS CLI, um verfügbare Angebote für reservierte Knoten aufzulisten und zu erwerben.

MemoryDB bietet reservierte Knoten für die speicheroptimierten Knoten R7g, R6g und R6gd (mit Datenklassierung). [Preisinformationen finden Sie unter Preise für MemoryDB for Redis.](#)

Angebotstypen

Reservierte Knoten sind in drei Varianten erhältlich — No Upfront, Partial Upfront und All Upfront —, mit denen Sie Ihre Kosten für MemoryDB for Redis auf der Grundlage Ihrer erwarteten Nutzung optimieren können.

Keine Vorauszahlung — Diese Option ermöglicht den Zugriff auf einen reservierten Knoten, ohne dass eine Vorauszahlung erforderlich ist. Ihr reservierter Node ohne Vorauszahlung berechnet für jede Stunde innerhalb der Laufzeit einen vergünstigten Stundensatz, unabhängig von der Nutzung, und es ist keine Vorauszahlung erforderlich.

Teilweise Vorauszahlung — Bei dieser Option muss ein Teil des reservierten Knotens im Voraus bezahlt werden. Die innerhalb der Laufzeit verbleibenden Stunden werden unabhängig von der Nutzung zu einem vergünstigten Stundensatz berechnet.

Vollständige Vorauszahlung — Die vollständige Zahlung erfolgt zu Beginn der Laufzeit, ohne dass für den Rest der Laufzeit weitere Kosten anfallen, unabhängig von der Anzahl der genutzten Stunden.

Alle drei Angebotstypen sind mit Laufzeiten von einem Jahr und drei Jahren erhältlich.

Größe: flexible reservierte Knoten

Wenn Sie einen reservierten Knoten kaufen, geben Sie unter anderem den Knotentyp an, z. B. `db.r6g.xlarge`. [Weitere Informationen zu Knotentypen finden Sie unter MemoryDB for Redis Pricing.](#)

Wenn Sie einen Knoten haben und ihn auf eine größere Kapazität skalieren müssen, wird Ihr reservierter Knoten automatisch auf Ihren skalierten Knoten angewendet. Das heißt, Ihre reservierten Knoten werden automatisch für die Nutzung beliebiger Größe in derselben Knotenfamilie verwendet. Größenflexible reservierte Knoten sind für Knoten mit derselben AWS Region verfügbar. Größenflexible reservierte Knoten können nur innerhalb ihrer Knotenfamilien skaliert werden. Beispielsweise kann ein reservierter Knoten für eine `db.r6g.xlarge` für eine `db.r6g.2xlarge` gelten, aber nicht für eine `db.r6gd.large`, da `db.r6g` und `db.r6gd` unterschiedliche Knotenfamilien sind.

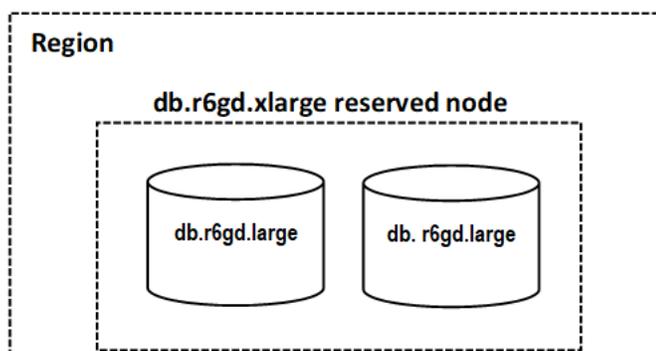
Größenflexibilität bedeutet, dass Sie sich frei zwischen Konfigurationen innerhalb derselben Knotenfamilie bewegen können. Sie können beispielsweise ohne zusätzliche Kosten von einem reservierten `r6g.xlarge`-Knoten (8 normalisierte Einheiten) zu zwei reservierten `r6g.large`-Knoten (8 normalisierte Einheiten) ($2 \times 4 = 8$ normalisierte Einheiten) in derselben Region wechseln. AWS

Sie können die Nutzung für verschiedene reservierte Knotengrößen vergleichen, indem Sie normalisierte Einheiten verwenden. Beispielsweise entspricht eine Nutzungseinheit auf zwei

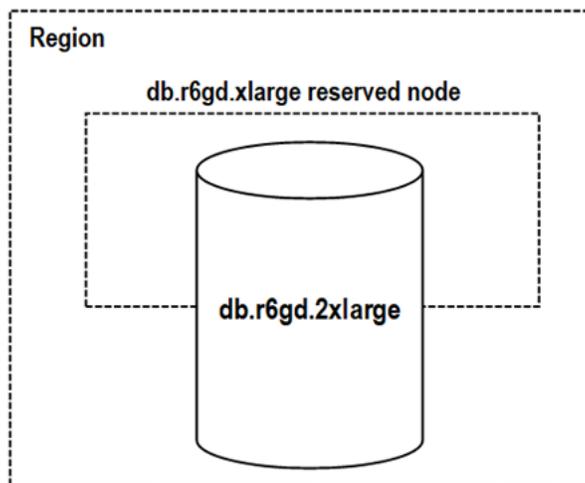
db.r6g.4xlarge-Knoten 16 normalisierten Nutzungseinheiten auf einem db.r6g.large. Die folgende Tabelle zeigt die Anzahl der normalisierten Einheiten für jede Knotengröße:

Knotengröße	Normalisierte Einheiten
small	1
Medium	2
large	4
xlarge	8
2xlarge	16
4xlarge	32
6xlarge	48
8xlarge	64
10xlarge	80
12xlarge	96
16xlarge	128

Beispiel: Sie kaufen einen reservierten Knoten db.r6gd.xlarge und Sie haben zwei reservierte Knoten vom Typ db.r6gd.large in Ihrem Konto in derselben Region. AWS In diesem Fall wird der Abrechnungsvorteil vollständig auf beide Knoten angewendet.



Wenn in Ihrem Konto in derselben AWS Region eine db.r6gd.2xlarge-Instance ausgeführt wird, wird der Abrechnungsvorteil alternativ auf 50 Prozent der Nutzung des reservierten Knotens angerechnet.



Löschen eines reservierten Knotens

Die Bedingungen für einen reservierten Knoten beinhalten eine Laufzeit von einem oder drei Jahren. Sie können einen reservierten Knoten nicht kündigen. Sie können jedoch einen Knoten löschen, für den ein discount für reservierte Knoten gilt. Der Vorgang zum Löschen eines Knotens, für den ein discount für reservierte Knoten gilt, ist derselbe wie für jeden anderen Knoten.

Wenn Sie einen Knoten löschen, für den ein discount für reservierte Knoten gilt, können Sie einen anderen Knoten mit kompatiblen Spezifikationen starten. In diesem Fall erhalten Sie den Rabatt während des Reservierungszeitraums (ein Jahr oder drei Jahre).

Mit reservierten Knoten arbeiten

Sie können die AWS Management Console, und die MemoryDB-API verwenden AWS Command Line Interface, um mit reservierten Knoten zu arbeiten.

Konsole

Um Preise und Informationen zu verfügbaren Angeboten für reservierte Knoten zu erhalten

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im Navigationsbereich Reserved Nodes aus.
3. Wählen Sie Reservierte Knoten kaufen aus.
4. Wählen Sie unter Knotentyp den Knotentyp aus, den Sie bereitstellen möchten.

5. Wählen Sie unter Menge die Anzahl der Knoten aus, die Sie bereitstellen möchten.
6. Wählen Sie unter Laufzeit den Zeitraum aus, für den der Datenbankknoten reserviert werden soll.
7. Wählen Sie für Angebotstyp den Angebotstyp aus.

Nachdem Sie diese Auswahl getroffen haben, können Sie die Preisinformationen unter Reservierungsübersicht einsehen.

 **Wichtig**

Wählen Sie Stornieren, um zu vermeiden, dass für den Kauf dieser reservierten Knoten Gebühren anfallen.

Nachdem Sie Informationen zu den verfügbaren Angeboten für reservierte Knoten erhalten haben, können Sie diese Informationen verwenden, um ein Angebot zu erwerben, wie im folgenden Verfahren beschrieben:

So kaufen Sie einen reservierten Knoten:

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im Navigationsbereich Reserved Nodes aus.
3. Wählen Sie Reservierte Knoten kaufen aus.
4. Wählen Sie unter Knotentyp den Knotentyp aus, den Sie bereitstellen möchten.
5. Wählen Sie unter Menge die Anzahl der Knoten aus, die Sie bereitstellen möchten.
6. Wählen Sie unter Laufzeit den Zeitraum aus, für den der Datenbankknoten reserviert werden soll.
7. Wählen Sie für Angebotstyp den Angebotstyp aus.
8. (Optional) Sie können den reservierten Knoten, die Sie erwerben, Ihre eigene Kennung zuweisen, um sie leichter nachverfolgen zu können. Geben Sie unter Reservierungs-ID eine Kennung für Ihren reservierten Knoten ein.

Nachdem Sie diese Auswahl getroffen haben, können Sie die Preisinformationen unter Reservierungsübersicht einsehen.

9. Wählen Sie Reservierte Knoten kaufen aus.
10. Ihre reservierten Knoten werden gekauft und dann in der Liste Reservierte Knoten angezeigt.

Um Informationen über reservierte Knoten für Ihr AWS Konto zu erhalten

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im Navigationsbereich Reserved Nodes aus.
3. Die reservierten Knoten für Ihr Konto werden angezeigt. Um detaillierte Informationen zu einem bestimmten reservierten Knoten zu erhalten, wählen Sie diesen Knoten in der Liste aus. Sie können dann detaillierte Informationen zu diesem Knoten im Detail sehen.

AWS Command Line Interface

Im folgenden `describe-reserved-nodes-offerings` Beispiel werden Details zu Angeboten für reservierte Knoten zurückgegeben.

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

Dadurch wird eine Ausgabe erzeugt, die der folgenden ähnelt:

```
{
  "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"
        }
      ]
    }
  ]
}
```

```
}
```

Sie können auch die folgenden Parameter übergeben, um den Umfang der zurückgegebenen Daten einzuschränken:

- `--reserved-nodes-offering-id` – Die ID des Angebots, das Sie erwerben möchten
- `--node-type`— Der Knotentyp-Filterwert. Verwenden Sie diesen Parameter, um nur die Reservierungen anzuzeigen, die dem angegebenen Knotentyp entsprechen.
- `--duration`— Der Wert des Dauerfilters, angegeben in Jahren oder Sekunden. Verwenden Sie diesen Parameter, um nur Reservierungen für diesen Zeitraum anzuzeigen.
- `--offering-type`— Verwenden Sie diesen Parameter, um nur die verfügbaren Angebote anzuzeigen, die dem angegebenen Angebotstyp entsprechen.

Sobald Sie Informationen über die verfügbaren Angebote für reservierte Knoten haben, können Sie diese Informationen verwenden, um ein Angebot zu erwerben.

Im folgenden `purchase-reserved-nodes-offering` Beispiel werden neue reservierte Knoten gekauft

Für Linux, macOS oder Unix:

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

Für Windows:

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

- `--reserved-nodes-offering-id` steht für den Namen der reservierten Knoten, die zum Kauf angeboten werden.
- `--reservation-id` ist eine vom Kunden angegebene Kennung zur Nachverfolgung dieser Reservierung.

Note

Die Reservierungs-ID ist eine eindeutige, vom Kunden angegebene Kennung zur Nachverfolgung dieser Reservierung. Wenn dieser Parameter nicht angegeben ist, generiert MemoryDB automatisch eine Kennung für die Reservierung.

- `--node-count` ist die Anzahl der zu reservierenden Knoten. Sie ist standardmäßig auf 1 eingestellt.

Dadurch wird eine Ausgabe erzeugt, die der folgenden ähnelt:

```
{
  "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:xxxxxxx:reservednode/reservation"
  }
}
```

Nachdem Sie reservierte Knoten gekauft haben, können Sie Informationen über Ihre reservierten Knoten abrufen.

Das folgende `describe-reserved-nodes` Beispiel gibt Informationen über reservierte Knoten für dieses Konto zurück.

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

Dadurch wird eine Ausgabe erzeugt, die der folgenden ähnelt:

```
{
  "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:xxxxxxx:reservednode/ri-2022-12-16-00-28-40-600"
    }
  ]
}
```

Sie können auch die folgenden Parameter übergeben, um den Umfang der zurückgegebenen Daten einzuschränken:

- `--reservation-id`— Sie können den reservierten Knoten, die Sie kaufen, Ihre eigene Kennung zuweisen, um sie besser verfolgen zu können.
- `--reserved-nodes-offering-id`— Der Filterwert für die Angebots-ID. Verwenden Sie diesen Parameter, um nur gekaufte Reservierungen anzuzeigen, die der angegebenen Angebots-ID entsprechen.
- `--node-type`— Der Knotentyp-Filterwert. Verwenden Sie diesen Parameter, um nur die Reservierungen anzuzeigen, die dem angegebenen Knotentyp entsprechen.
- `--duration`— Der Wert des Dauerfilters, angegeben in Jahren oder Sekunden. Verwenden Sie diesen Parameter, um nur Reservierungen für diesen Zeitraum anzuzeigen.

- `--offering-type`— Verwenden Sie diesen Parameter, um nur die verfügbaren Angebote anzuzeigen, die dem angegebenen Angebotstyp entsprechen.

MemoryDB-API

Die folgenden Beispiele zeigen, wie die [MemoryDB-Abfrage-API](#) für reservierte Knoten verwendet wird:

DescribeReservedNodesOfferings

Gibt Details zu Angeboten für reservierte Knoten zurück.

```
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>
```

Die folgenden Parameter schränken den Umfang der zurückgegebenen Daten ein:

- `ReservedNodesOfferingId` steht für den Namen der reservierten Knoten, die zum Kauf angeboten werden.
- `Duration`— Der Wert des Dauerfilters, angegeben in Jahren oder Sekunden. Verwenden Sie diesen Parameter, um nur Reservierungen für diesen Zeitraum anzuzeigen.
- `NodeType`— Der Knotentyp-Filterwert. Verwenden Sie diesen Parameter, um nur die Angebote anzuzeigen, die dem angegebenen Knotentyp entsprechen.
- `OfferingType`— Verwenden Sie diesen Parameter, um nur die verfügbaren Angebote anzuzeigen, die dem angegebenen Angebotstyp entsprechen.

Sobald Sie Informationen über die verfügbaren Angebote für reservierte Knoten haben, können Sie diese Informationen verwenden, um ein Angebot zu erwerben.

PurchaseReservedNodesOffering

Ermöglicht Ihnen den Kauf eines Angebots für reservierte Knoten.

```
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` steht für den Namen der reservierten Knoten, die zum Kauf angeboten werden.
- `ReservationID` ist eine vom Kunden angegebene Kennung zur Nachverfolgung dieser Reservierung.

Note

Die Reservierungs-ID ist eine eindeutige, vom Kunden angegebene Kennung zur Nachverfolgung dieser Reservierung. Wenn dieser Parameter nicht angegeben ist, generiert MemoryDB automatisch eine Kennung für die Reservierung.

- `NodeCount` ist die Anzahl der zu reservierenden Knoten. Sie ist standardmäßig auf 1 eingestellt.

Nachdem Sie reservierte Knoten gekauft haben, können Sie Informationen zu Ihren reservierten Knoten abrufen.

DescribeReservedNodes

Gibt Informationen über reservierte Knoten für dieses Konto zurück.

```
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>
```

Die folgenden Parameter schränken den Umfang der zurückgegebenen Daten ein:

- `ReservedNodesOfferingId` steht für den Namen des reservierten Knotens.
- `ReservationID`— Sie können den reservierten Knoten, die Sie kaufen, Ihre eigene Kennung zuweisen, um sie leichter verfolgen zu können.
- `NodeType`— Der Knotentyp-Filterwert. Verwenden Sie diesen Parameter, um nur die Reservierungen anzuzeigen, die dem angegebenen Knotentyp entsprechen.
- `Duration`— Der Wert des Dauerfilters, angegeben in Jahren oder Sekunden. Verwenden Sie diesen Parameter, um nur Reservierungen für diesen Zeitraum anzuzeigen.
- `OfferingType`— Verwenden Sie diesen Parameter, um nur die verfügbaren Angebote anzuzeigen, die dem angegebenen Angebotstyp entsprechen.

Die Abrechnung für Ihre reservierten Knoten anzeigen

Sie können die Abrechnung für Ihre reservierten Knoten im Abrechnungs-Dashboard unter einsehen AWS Management Console.

Um die Abrechnung für reservierte Knoten einzusehen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie oben in der Konsole über die Schaltfläche Suchen die Option Abrechnung aus.

3. Wählen Sie auf der linken Seite des Dashboards Rechnungen aus.
4. Erweitern Sie unter AWS Servicegebühren die Option MemoryDB.
5. Erweitern Sie die AWS Region, in der sich Ihre reservierten Knoten befinden, z. B. USA Ost (Nord-Virginia).

Ihre reservierten Knoten und ihre Stundengebühren für den aktuellen Monat werden unter Amazon MemoryDB CreateCluster Reserved Instances angezeigt.

Instance Type	Hours	Price
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	...

Ersetzen von Knoten

MemoryDB aktualisiert seine Flotte häufig mit Patches und Upgrades, normalerweise nahtlos. Von Zeit zu Zeit müssen wir jedoch Ihre MemoryDB-Knoten neu starten, um obligatorische Betriebssystemaktualisierungen auf den zugrunde liegenden Host anzuwenden. Diese Ersetzungen sind erforderlich, um Upgrades anzuwenden, die die Sicherheit, Zuverlässigkeit und Betriebsleistung erhöhen.

Sie haben die Möglichkeit, diese Ersetzungen vor dem geplanten Knotenersetzungsfenster jederzeit selbst zu verwalten. Wenn Sie eine Ersetzung selbst verwalten, erhält Ihre Instance das Betriebssystem-Update, wenn Sie den Knoten neu starten, und der geplante Austausch des Knotens wird abgebrochen. Sie erhalten möglicherweise weiterhin Benachrichtigungen darüber, dass der Knotenaustausch stattfinden wird. Wenn Sie die erforderliche Wartung bereits manuell vorgenommen haben, können Sie diese Benachrichtigungen ignorieren.

Note

Von MemoryDB für Redis automatisch generierte Ersatzknoten können unterschiedliche IP-Adressen haben. Sie sind dafür verantwortlich, Ihre Anwendungskonfiguration zu überprüfen, um sicherzustellen, dass Ihre Knoten den entsprechenden IP-Adressen zugeordnet sind.

In der folgenden Liste sind die Aktionen aufgeführt, die Sie ausführen können, wenn MemoryDB einen Ihrer Knoten für den Ersatz plant:

Optionen zum Ersetzen von MemoryDB-Knoten

- Nichts tun – Wenn Sie nichts tun, ersetzt MemoryDB den Knoten wie geplant.

Wenn der Knoten Mitglied eines Multi-AZ-Clusters ist, bietet MemoryDB eine verbesserte Verfügbarkeit beim Patchen, Aktualisieren und anderen wartungsbezogenen Knotenaustauschen.

Der Austausch wird abgeschlossen, während der Cluster eingehende Schreibanforderungen bedient.

- Ändern Ihres Wartungsfensters – Bei geplanten Wartungsereignissen erhalten Sie eine E-Mail oder ein Benachrichtigungsereignis von MemoryDB. Wenn Sie in diesen Fällen Ihr Wartungsfenster vor der geplanten Ersatzzeit ändern, wird Ihr Knoten zur neuen Uhrzeit ersetzt. Weitere Informationen finden Sie unter [Einen MemoryDB-Cluster ändern](#).

Note

Die Möglichkeit, Ihr Ersatzfenster durch Verschieben Ihres Wartungsfensters zu ändern, ist nur verfügbar, wenn die MemoryDB-Benachrichtigung ein Wartungsfenster enthält. Wenn die Benachrichtigung kein Wartungsfenster enthält, können Sie Ihr Ersatzfenster nicht ändern.

Es ist beispielsweise Donnerstag, der 9. November, um 15:00 Uhr und das nächste Wartungsfenster ist am Freitag, dem 10. November, um 17:00 Uhr. Es folgen drei Szenarien mit den jeweiligen Ergebnissen:

- Sie ändern Ihr Wartungsfenster auf Freitag, 16:00 Uhr, nach den aktuellen Datums- und Uhrzeitangaben, aber vor dem nächsten geplanten Wartungsfenster. Dieser Knoten wird am Freitag, dem 10. November, um 16:00 Uhr ersetzt.
- Sie ändern Ihr Wartungsfenster auf Samstag, 16:00 Uhr, nach den aktuellen Datums- und Uhrzeitangaben und nach dem nächsten geplanten Wartungsfenster. Dieser Knoten wird am Samstag, dem 11. November, um 16:00 Uhr ersetzt.
- Sie ändern Ihr Wartungsfenster auf Mittwoch um 16:00 Uhr, früher in der Woche als das aktuelle Datum und die aktuelle Uhrzeit. Dieser Knoten wird am kommenden Mittwoch, dem 15. November, um 16:00 ersetzt.

Detaillierte Anweisungen finden Sie unter [Verwaltung der Wartung](#).

Verwalten von Clustern

Die meisten MemoryDB-Operationen werden auf Clusterebene ausgeführt. Sie können einen Cluster mit einer bestimmten Anzahl von Knoten und einer Parametergruppe einrichten, die die Eigenschaften für jeden Knoten steuert. Alle Knoten innerhalb eines Clusters gehören demselben Knotentyp an und verfügen über die gleichen Einstellungen für Parameter und Sicherheitsgruppen.

Jeder Cluster muss über eine Cluster-Kennung verfügen. Die Cluster-Kennung ist ein vom Kunden angegebener Name für den Cluster. Diese Kennung gibt einen bestimmten Cluster an, wenn mit der MemoryDB-API und den MemoryDB-Befehlen interagiert wird. AWS CLI Die Cluster-Kennung muss für diesen Kunden in einer AWS-Region eindeutig sein.

MemoryDB-Cluster sind für den Zugriff über eine Amazon EC2 EC2-Instance konzipiert. Sie können Ihren MemoryDB-Cluster nur in einer Virtual Private Cloud (VPC) starten, die auf dem Amazon VPC-Service basiert, aber Sie können von außen darauf zugreifen. AWS Weitere Informationen finden Sie unter [Zugriff auf MemoryDB-Ressourcen von außerhalbAWS](#).

Daten-Tiering

Die Daten von Clustern, die einen Knotentyp aus der R6gd-Familie verwenden, sind zwischen Speicher und lokalem SSD-Speicher (Solid-State-Laufwerke) gestaffelt. Daten-Tiering bietet eine neue Option mit gutem Preis-Leistungs-Verhältnis für Redis-Workloads, indem Daten nicht nur im Speicher gespeichert werden, sondern zusätzlich auch kostengünstigere Solid-State-Laufwerke (SSDs) in jedem Clusterknoten verwendet werden. Ähnlich wie bei anderen Knotentypen werden die auf r6gd-Knoten geschriebenen Daten dauerhaft in einem Multi-AZ-Transaktionsprotokoll gespeichert. Daten-Tiering ist ideal für Workloads, die regelmäßig auf bis zu 20 Prozent ihres gesamten Datensatzes zugreifen, und für Anwendungen, die beim Zugriff auf Daten auf SSD zusätzliche Latenz tolerieren können.

In Clustern mit Daten-Tiering überwacht MemoryDB die letzte Zugriffszeit jedes gespeicherten Elements. Wenn der verfügbare Speicher (DRAM) vollständig verbraucht ist, verwendet MemoryDB einen LRU-Algorithmus (Least Recently Used), um selten aufgerufene Elemente automatisch aus dem Speicher auf das SSD zu verschieben. Wenn anschließend auf Daten auf dem SSD zugegriffen wird, verschiebt MemoryDB sie automatisch und asynchron zurück in den Speicher, bevor die Anforderung verarbeitet wird. Wenn Sie eine Workload haben, die regelmäßig nur auf eine Teilmenge ihrer Daten zugreift, ist Daten-Tiering eine optimale Möglichkeit, Ihre Kapazität kostengünstig zu skalieren.

Beachten Sie, dass bei der Verwendung von Daten-Tiering die Schlüssel selbst immer im Speicher verbleiben, während die LRU die Platzierung von Werten in dem Speicher im Vergleich zur Festplatte regelt. Im Allgemeinen empfehlen wir, dass Ihre Schlüsselgrößen kleiner als Ihre Wertgrößen sind, wenn Sie Daten-Tiering verwenden.

Das Daten-Tiering ist so konzipiert, dass es minimale Auswirkungen auf die Leistung von Anwendungs-Workloads hat. Bei Annahme von 500-Byte-String-Werten können Sie beispielsweise normalerweise zusätzliche 450 Mikrosekunden Latenzzeit für Leseanforderungen an auf SSD gespeicherte Daten im Vergleich zu Leseanforderungen an Daten im Speicher erwarten.

Mit der größten Daten-Tiering-Knotengröße (db.r6gd.8xlarge) können Sie bis zu ~500 TB in einem einzigen Cluster mit 500 Knoten speichern (250 TB bei Verwendung von 1 Lesereplikat). Für die Datenklassifizierung reserviert MemoryDB 19% des (DRAM-) Speichers pro Knoten für die Nichtdatennutzung. Daten-Tiering ist mit allen Redis-Befehlen und -Datenstrukturen kompatibel, die in MemoryDB unterstützt werden. Um diese Funktion nutzen zu können, sind keine clientseitigen Änderungen erforderlich.

Themen

- [Bewährte Methoden](#)
- [Einschränkungen](#)
- [Preise für Daten-Tiering](#)
- [Überwachung](#)
- [Verwenden von Daten-Tiering](#)
- [Wiederherstellen von Daten aus einem Snapshot in Cluster mit aktiviertem Daten-Tiering](#)

Bewährte Methoden

Wir empfehlen Ihnen, die folgenden bewährten Methoden:

- Daten-Tiering ist ideal für Workloads, die regelmäßig auf bis zu 20 Prozent ihres gesamten Datensatzes zugreifen, und für Anwendungen, die beim Zugriff auf Daten auf SSD zusätzliche Latenz tolerieren können.
- Bei Verwendung von SSD-Kapazität, die auf Daten-Tiering-Knoten verfügbar ist, empfehlen wir, dass die Wertgröße größer als die Schlüsselgröße ist. Die Wertgröße darf nicht größer als 128 MB sein. Andernfalls wird er nicht auf die Festplatte verschoben. Wenn Elemente zwischen DRAM und SSD verschoben werden, bleiben die Schlüssel immer im Speicher und nur die Werte werden in die SSD-Ebene verschoben.

Einschränkungen

Für Daten-Tiering gelten die folgenden Beschränkungen:

- Der verwendete Knotentyp muss aus der r6gd-Familie stammen, die in den folgenden Regionen verfügbar ist: 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 und sa-east-1.
- Sie können einen Snapshot eines R6gd-Clusters nur dann in einem anderen Cluster wiederherstellen, wenn dieses ebenfalls R6gd verwendet.
- Sie können einen Snapshot für Cluster mit Daten-Tiering nicht nach Amazon S3 exportieren.
- Das unvergabelte Speichern wird nicht unterstützt.
- Die Skalierung von einem Cluster mit Daten-Tiering (z. B. ein Cluster, der einen R6gd-Knotentyp verwendet) zu einem Cluster ohne Daten-Tiering (z. B. ein Cluster, der einen R6g-Knotentyp verwendet) wird nicht unterstützt.

- Daten-Tiering unterstützt nur die maxmemory-Richtlinien `volatile-lru`, `allkeys-lru` und `noeviction`.
- Elemente, die größer als 128 MiB sind, werden nicht auf SSD verschoben.

Preise für Daten-Tiering

R6gd-Knoten haben eine 5-mal höhere Gesamtkapazität (Speicher + SSD). Bei maximaler Auslastung können Sie durch ihren Einsatz Einsparungen von über 60 Prozent im Vergleich zu R6g-Knoten (nur Speicher) erzielen. Weitere Informationen finden Sie [Sie finden Sie finden Sie finden Sie finden Sie](#) finden [Sie](#) finden Sie finden Sie finden Sie finden Sie

Überwachung

MemoryDB bietet Metriken, die speziell zur Überwachung der Leistungs-Cluster entwickelt wurden, die Daten-Tiering verwenden. Um das Verhältnis von Elementen in DRAM im Vergleich zu SSD zu überwachen, können Sie die `currItems` Metrik verwenden [Metriken für MemoryDB](#). Sie können den Prozentsatz berechnen als: $(\text{currItems with Dimension: Tier} = \text{Memory} * 100) / (\text{currItems with no dimension filter})$. Wenn der Prozentsatz der Elemente im Speicher sinkt, empfehlen wir Ihnen, dies in Betracht zu ziehen [Skalieren von MemoryDB-Clustern](#).

Weitere Informationen finden Sie [Sie finden Sie finden Sie](#) finden Sie [Metriken für MemoryDB](#)

Verwenden von Daten-Tiering

Verwenden von Daten-Tiering mit der AWS Management Console

Beim Erstellen eines Clusters verwenden Sie Daten-Tiering, indem Sie einen Knotentyp aus der R6gd-Familie auswählen, z. B. `db.r6gd.xlarge`. Bei Auswahl dieses Knotentyps wird das Daten-Tiering automatisch aktiviert.

Weitere Informationen zum Erstellen von Clustern finden Sie unter [Schritt 1: Erstellen eines Clusters](#).

Aktivieren von Daten-Tiering mit der AWS CLI

Beim Erstellen eines Clusters mit der verwenden Sie Daten-Tiering AWS CLI, indem Sie einen Knotentyp aus der R6gd-Familie auswählen, z. B. `db.r6gd.xlarge`, und den `--data-tiering` Parameter festlegen.

Sie können sich das Daten-Tiering nicht abwählen, wenn Sie einen Knotentyp aus der R6gd-Familie auswählen. Wenn Sie den Parameter `--no-data-tiering` festlegen, schlägt die Operation fehl.

Für Linux, macOS oder Unix:

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

Für Windows:

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

Nach dem Ausführen dieses Vorgangs sehen Sie eine Antwort ähnlich dem folgenden:

```
{  
  "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:xxxxxxxxxxxx:cluster/my-cluster",  
    "SnapshotRetentionLimit": 0,  
    "MaintenanceWindow": "wed:03:00-wed:04:00",  
    "SnapshotWindow": "04:30-05:30",  
  }  
}
```

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

Wiederherstellen von Daten aus einem Snapshot in Cluster mit aktiviertem Daten-Tiering

Sie können einen Snapshot in einem neuen Cluster mit aktiviertem Daten-Tiering wiederherstellen. AWS Wenn Sie einen Cluster mit Knotentypen in der R6gd-Familie erstellen, ist Daten-Tiering aktiviert.

Wiederherstellen von Daten aus einem Snapshot in Cluster mit aktiviertem Daten-Tiering (Konsole)

Um einen Snapshot in einem neuen Cluster mit aktiviertem Daten-Tiering wiederherzustellen (Konsole) [Wiederherstellung aus einem Snapshot \(Konsole\)](#)

Beachten Sie, dass Sie einen Knotentyp aus der R6gd-Familie auswählen müssen.

Wiederherstellen von Daten aus einem Snapshot in Cluster mit aktiviertem Daten-Tiering (AWSCLI)

Beim Erstellen eines Clusters mit der wird Daten-Tiering verwendet AWS CLI, indem Sie einen Knotentyp aus der R6gd-Familie auswählen, z. B. db.r6gd.xlarge, und den `--data-tiering` Parameter festlegen.

Sie können sich das Daten-Tiering nicht abwählen, wenn Sie einen Knotentyp aus der R6gd-Familie auswählen. Wenn Sie den Parameter `--no-data-tiering` festlegen, schlägt die Operation fehl.

Für Linux, macOS oder 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
```

Für Linux, macOS oder 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
```

Nach dem Ausführen dieses Vorgangs sehen Sie eine Antwort ähnlich dem folgenden:

```
{
  "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:xxxxxxxxxxxx:cluster/my-cluster",
    "SnapshotRetentionLimit": 0,
    "MaintenanceWindow": "wed:03:00-wed:04:00",
    "SnapshotWindow": "04:30-05:30",
    "ACLName": "my-acl",
    "DataTiering": "true"
  }
}
```

Vorbereitung eines Clusters

Im Folgenden finden Sie Anweisungen zum Erstellen eines Clusters mithilfe der MemoryDB-Konsole, der oder der MemoryDB-API. AWS CLI

Bei der Erstellung eines Clusters empfiehlt es sich, einige Vorarbeiten zu erledigen, damit Sie nicht sofort ein Upgrade durchführen oder Änderungen vornehmen müssen.

Themen

- [Bestimmung Ihrer Anforderungen](#)

Bestimmung Ihrer Anforderungen

Vorbereitung

Wenn Sie die Antworten auf die folgenden Fragen kennen, können Sie die Erstellung Ihres Clusters vereinfachen:

- Stellen Sie sicher, dass Sie eine Subnetzgruppe in derselben VPC erstellen, bevor Sie mit der Erstellung eines Clusters beginnen. Alternativ können Sie die bereitgestellte Standard-Subnetzgruppe verwenden. Weitere Informationen finden Sie unter [Subnetze und Subnetzgruppen](#).

MemoryDB ist so konzipiert, dass auf Amazon EC2 von innen AWS zugegriffen werden kann. Wenn Sie jedoch in einer VPC starten, die auf Amazon VPC basiert, können Sie Zugriff von außen gewähren. AWS Weitere Informationen finden Sie unter [Zugriff auf MemoryDB-Ressourcen von außerhalbAWS](#).

- Müssen Sie irgendwelche Parameterwerte anpassen?

Erstellen Sie in diesem Fall eine benutzerdefinierte Parametergruppe. Weitere Informationen finden Sie unter [Erstellen einer Parametergruppe](#).

- Müssen Sie eine VPC-Sicherheitsgruppe erstellen?

Weitere Informationen finden Sie unter [Sicherheit in Ihrer VPC](#).

- Haben Sie vor, Fehlertoleranz zu implementieren?

Weitere Informationen finden Sie unter [Minimieren von Ausfällen](#).

Themen

- [Anforderungen an Speicher und Prozessor](#)
- [MemoryDB-Cluster-Konfiguration](#)
- [Verbessertes I/O-Multiplexing](#)
- [Voraussetzungen für Skalierung](#)
- [Zugriffsvoraussetzungen](#)
- [Region und Verfügbarkeitszonen](#)

Anforderungen an Speicher und Prozessor

Der Grundbaustein von MemoryDB for Redis ist der Knoten. Knoten werden in Shards konfiguriert, um Cluster zu bilden. Berücksichtigen Sie bei der Bestimmung des für den Cluster zu verwendenden Knotentyps die Knotenkonfiguration des Clusters und die zu speichernde Datenmenge.

MemoryDB-Cluster-Konfiguration

MemoryDB-Cluster bestehen aus 1 bis 500 Shards. Die Daten in einem MemoryDB-Cluster sind auf die Shards im Cluster verteilt. Ihre Anwendung stellt über eine Netzwerkadresse, die als Endpunkt bezeichnet wird, eine Verbindung zu einem MemoryDB-Cluster her. Zusätzlich zu den Knotenendpunkten hat der MemoryDB-Cluster selbst einen Endpunkt, den Cluster-Endpunkt. Ihre Anwendung kann diesen Endpunkt verwenden, um aus dem Cluster zu lesen oder in ihn zu schreiben, wobei die Entscheidung, von welchem Knoten gelesen oder auf welchen geschrieben werden soll, MemoryDB überlassen bleibt.

Verbessertes I/O-Multiplexing

Wenn Sie Redis Version 7.0 oder höher verwenden, erhalten Sie zusätzliche Beschleunigung durch erweitertes I/O-Multiplexing, bei dem jeder dedizierte Netzwerk-I/O-Thread Befehle von mehreren Clients an die Redis-Engine weiterleitet, wodurch die Fähigkeit von Redis genutzt wird, Befehle effizient stapelweise zu verarbeiten. [Weitere Informationen finden Sie unter *Extrem schnelle Leistung und the section called "Unterstützte Knotentypen"*](#)

Voraussetzungen für Skalierung

Alle Cluster können auf einen größeren Knotentyp hochskaliert werden. Wenn Sie einen MemoryDB-Cluster hochskalieren, können Sie dies online tun, sodass der Cluster verfügbar bleibt, oder Sie können einen neuen Cluster anhand eines Snapshots erstellen und vermeiden, dass der neue Cluster leer anfängt.

Weitere Informationen finden Sie unter [Skalierung](#) in diesem Handbuch.

Zugriffsvoraussetzungen

MemoryDB-Cluster werden standardmäßig von Amazon EC2 EC2-Instances aus aufgerufen. Der Netzwerkzugriff auf einen MemoryDB-Cluster ist auf das Konto beschränkt, mit dem der Cluster erstellt wurde. Bevor Sie von einer Amazon EC2 EC2-Instance aus auf einen Cluster zugreifen können, müssen Sie daher den Zugang zum Cluster autorisieren. Detaillierte Anweisungen finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#) in diesem Handbuch.

Region und Verfügbarkeitszonen

Indem Sie Ihre MemoryDB-Cluster in einer AWS Region in der Nähe Ihrer Anwendung platzieren, können Sie die Latenz reduzieren. Bei Clustern mit mehreren Knoten lassen sich die Auswirkungen von Ausfällen auf Ihren Cluster reduzieren, indem Sie Ihre Knoten in verschiedenen Availability Zones platzieren.

Weitere Informationen finden Sie hier:

- [Auswählen von Regionen und Availability Zones](#)
- [Minimieren von Ausfällen](#)

Erstellen eines Clusters

MemoryDB for Redis bietet drei Möglichkeiten, einen Cluster zu erstellen. Weitere Informationen finden Sie unter [Schritt 1: Erstellen eines Clusters](#).

Anzeigen der Details eines Clusters

Sie können detaillierte Informationen zu einem oder mehreren Clustern mithilfe der MemoryDB-Konsole oder der MemoryDB-API anzeigen. AWS CLI

Details für einen MemoryDB-Cluster anzeigen (Konsole)

Das folgende Verfahren beschreibt, wie Sie die Details eines MemoryDB-Clusters mithilfe der MemoryDB-Konsole anzeigen.

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Um Details zu einem Cluster anzuzeigen, wählen Sie das Optionsfeld links neben dem Namen des Clusters und dann Details anzeigen aus. Sie können auch direkt auf den Cluster klicken, um die Seite mit den Cluster-Details aufzurufen.

Auf der Seite mit den Cluster-Details werden Details zum Cluster angezeigt, einschließlich des Cluster-Endpunkts. Sie können weitere Details mithilfe der verschiedenen Registerkarten auf der Seite mit den Cluster-Details anzeigen.

3. Wählen Sie den Tab Shards and Nodes, um eine Liste der Shards des Clusters und die Anzahl der Knoten in jedem Shard zu sehen.
4. Um spezifische Informationen zu einem Knoten anzuzeigen, erweitern Sie den Shard in der Tabelle unten. Alternativ können Sie auch über das Suchfeld nach dem Shard suchen.

Dadurch werden Informationen zu jedem Knoten angezeigt, einschließlich seiner Availability Zone, seiner Slots/Keyspaces und seines Status.

5. Wählen Sie die Registerkarte Metriken, um die jeweiligen Prozesse zu überwachen, z. B. die CPU-Auslastung und die Engine-CPU-Auslastung. Weitere Informationen finden Sie unter [Metriken für MemoryDB](#).
6. Wählen Sie die Registerkarte Netzwerk und Sicherheit, um Details zu der Subnetzgruppe und den Sicherheitsgruppen anzuzeigen.
 - a. Unter Subnetzgruppe sehen Sie den Namen der Subnetzgruppe, einen Link zu der VPC, zu der das Subnetz gehört, und den Amazon-Ressourcennamen (ARN) der Subnetzgruppe.
 - b. Unter Sicherheitsgruppen können Sie die ID, den Namen und die Beschreibung der Sicherheitsgruppe sehen.

7. Wählen Sie die Registerkarte **Wartung und Snapshot**, um Details zu den Snapshot-Einstellungen zu sehen.
 - a. In **Snapshot** können Sie sehen, ob automatische Snapshots aktiviert sind, wie lange Snapshots aufbewahrt werden und welches Snapshot-Fenster geöffnet ist.
 - b. Unter **Snapshots** sehen Sie eine Liste aller Snapshots dieses Clusters, einschließlich des Snapshot-Namens, der Größe, der Anzahl der Shards und des Status.

Weitere Informationen finden Sie unter [Snapshot und Wiederherstellung](#).

8. Wählen Sie die Registerkarte **Wartung und Snapshot**, um Details zum Wartungsfenster sowie alle ausstehenden ACL-, Resharding- oder Service-Updates anzuzeigen. Weitere Informationen finden Sie unter [Verwaltung der Wartung](#).
9. Wählen Sie die Registerkarte **Service Updates**, um Details zu allen Service-Updates zu sehen, die für diesen Cluster gelten. Weitere Informationen finden Sie unter [Service-Updates in MemoryDB für Redis](#).
10. Wählen Sie die Registerkarte **„Tags“**, um Details zu allen Ressourcen- oder Kostenzuweisungs-Tags anzuzeigen, die mit diesem Cluster verknüpft sind. Weitere Informationen finden Sie unter [Schnappschüsse taggen](#).

Details eines Clusters anzeigen (AWSCLI)

Sie können die Details eines Clusters mit dem AWS CLI-Befehl `describe-clusters` abrufen. Wenn der Parameter `--cluster-name` weggelassen wird, werden Details für mehrere Cluster, bis zu `--max-results`, zurückgegeben. Wenn der Parameter `--cluster-name` eingeschlossen wird, werden Details für den angegebenen Cluster zurückgegeben. Sie können die Anzahl der mit dem Parameter `--max-results` zurückgegebenen Datensätze begrenzen.

Der folgende Code listet die Details für `my-cluster` auf.

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

Der folgende Code listet die Details für bis zu 25 Cluster auf.

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

Example

Für Linux, macOS oder Unix:

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

Für Windows:

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

Die folgende JSON-Ausgabe zeigt die Antwort:

```
{  
  "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-1:0000000000:cluster/my-cluster",
"SnapshotRetentionLimit": 0,
"MaintenanceWindow": "sat:06:30-sat:07:30",
"SnapshotWindow": "04:00-05:00",
"ACLName": "open-access",
"DataTiering": "false",
"AutoMinorVersionUpgrade": true,
}

```

Weitere Informationen finden Sie im Thema AWS CLI für MemoryDB. [describe-clusters](#)

Details eines Clusters anzeigen (MemoryDB-API)

Sie können die Details für einen Cluster mithilfe der MemoryDB-API-Aktion anzeigen.

DescribeClusters Wenn der Parameter `ClusterName` eingeschlossen wird, werden Details für den angegebenen Cluster zurückgegeben. Wenn der Parameter `ClusterName` weggelassen wird, werden Details für bis zu `MaxResults` (Standard 100) Cluster zurückgegeben. Der Wert für `MaxResults` darf nicht kleiner als 20 oder größer als 100 sein.

Der folgende Code listet die Details für `my-cluster` auf.

```
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>
```

Der folgende Code listet die Details für bis zu 25 Cluster auf.

```
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>
```

Weitere Informationen finden Sie im Referenzthema MemoryDB-API. [DescribeClusters](#)

Einen MemoryDB-Cluster ändern

Neben dem Hinzufügen oder Entfernen von Knoten aus einem Cluster kann es vorkommen, dass Sie weitere Änderungen an einem vorhandenen Cluster vornehmen müssen, z. B. das Hinzufügen einer Sicherheitsgruppe, das Ändern des Wartungsfensters oder einer Parametergruppe.

Es wird empfohlen, dass das Wartungsfenster in den Zeitraum mit der geringsten Nutzung fällt. Dies muss folglich zeitweise korrigiert werden.

Wenn Sie die Parameter eines Clusters ändern, wird die Änderung sofort auf den Cluster angewendet. Dies gilt unabhängig davon, ob Sie die Parametergruppe des Clusters oder einen Parameterwert innerhalb der Parametergruppe des Clusters ändern.

Sie können auch die Engine-Version Ihrer Cluster aktualisieren. Sie können beispielsweise eine neue Engine-Nebenversion auswählen und MemoryDB beginnt sofort mit der Aktualisierung Ihres Clusters.

Verwendung von AWS Management Console

So modifizieren Sie einen Cluster:

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie aus der Liste in der oberen rechten Ecke die AWS-Region aus, in der sich der Cluster befindet, den Sie ändern möchten.
3. Gehen Sie in der linken Navigationsleiste zu Clusters. Wählen Sie unter Clusterdetails den Cluster mithilfe der Optionsschaltfläche aus und gehen Sie zu Aktionen und dann zu Ändern.
4. Die Seite Ändern wird angezeigt.
5. Nehmen Sie im Fenster Ändern die gewünschten Änderungen vor. Zu den Optionen gehören:
 - Beschreibung
 - Subnetzgruppen
 - VPC Security group(s) (VPC-Sicherheitsgruppe(n))
 - Knotentyp

Note

Wenn der Cluster einen Knotentyp aus der R6gd-Familie verwendet, können Sie nur eine andere Knotengröße aus dieser Familie auswählen. Wenn Sie einen Knotentyp

aus der R6gd-Familie wählen, wird Daten-Tiering automatisch aktiviert. Weitere Informationen finden Sie unter [Daten-Tiering](#).

- Redis-Versionskompatibilität
- Aktivieren Sie automatische Schnappschüsse
- Aufbewahrungszeitraum für Snapshots
- Snapshot-Fenster
- Wartungsfenster
- Thema für SNS-Benachrichtigung

6. Wählen Sie Änderungen speichern aus.

Sie können auch auf der Seite mit den Cluster-Details auf Ändern klicken, um Änderungen am Cluster vorzunehmen. Wenn Sie bestimmte Abschnitte des Clusters ändern möchten, können Sie auf der Seite mit den Cluster-Details auf die entsprechende Registerkarte wechseln und auf Ändern klicken.

Verwenden der AWS CLI

Sie können einen vorhandenen Cluster mithilfe der AWS CLI-Operation `update-cluster` ändern. Um den Konfigurationswert eines Clusters zu ändern, geben Sie die ID des Clusters, den zu ändernden Parameter und den neuen Wert des Parameters ein. Das folgende Beispiel ändert das Wartungsfenster für einen Cluster namens `my-cluster` und übernimmt die Änderung umgehend.

Für Linux, macOS oder Unix:

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

Für Windows:

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

Weitere Informationen finden Sie unter [update-cluster](#) in der AWS CLI Befehlsreferenz.

Verwenden der MemoryDB-API

Sie können einen vorhandenen Cluster mithilfe der MemoryDB-API-Operation ändern. [UpdateCluster](#)
Um den Konfigurationswert eines Clusters zu ändern, geben Sie die ID des Clusters, den zu ändernden Parameter und den neuen Wert des Parameters ein. Das folgende Beispiel ändert das Wartungsfenster für einen Cluster namens `my-cluster` und übernimmt die Änderung umgehend.

```
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>
```

Knoten zu einem Cluster hinzufügen/entfernen

Sie können Knoten mit der AWS Management Console, der oder der MemoryDB-API zu einem Cluster hinzufügen oder daraus entfernen. AWS CLI

Verwenden der AWS Management Console

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie aus der Clusterliste den Clusternamen aus, zu dem Sie einen Knoten hinzufügen oder entfernen möchten.
3. Wählen Sie auf der Registerkarte Shards and Nodes die Option Knoten hinzufügen/löschen
4. Geben Sie im Feld Neue Anzahl von Knoten die gewünschte Anzahl von Knoten ein.
5. Wählen Sie Bestätigen aus.

Important

Wenn Sie die Anzahl der Knoten auf 1 setzen, ist Multi-AZ für Sie nicht mehr aktiviert. Sie können sich auch dafür entscheiden, Auto Failover zu aktivieren.

Verwenden der AWS CLI

1. Identifizieren Sie die Namen der Knoten, die Sie entfernen möchten. Weitere Informationen finden Sie unter [Anzeigen der Details eines Clusters](#).
2. Listen Sie mit der CLI-Operation `update-cluster` wie im folgenden Beispiel die zu entfernenden Knoten auf.

Um über die Befehlszeilenschnittstelle Knoten aus einem Cluster zu entfernen, verwenden Sie den Befehl `update-cluster` mit den folgenden Parametern:

- `--cluster-name` Die ID des Clusters, aus dem Sie Knoten entfernen möchten.
- `--replica-configuration`— Ermöglicht es Ihnen, die Anzahl der Replikate festzulegen:
 - `ReplicaCount`— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Replikatknoten anzugeben.
- `--region` Gibt die AWS-Region des Clusters an, aus der Sie Knoten entfernen möchten.

Für Linux, macOS oder Unix:

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

Für Windows:

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

Weitere Informationen finden Sie in den AWS CLI-Themen [update-cluster](#).

Verwenden der MemoryDB-API

Um Knoten mithilfe der MemoryDB-API zu entfernen, rufen Sie den UpdateCluster API-Vorgang mit dem Clusternamen und einer Liste der zu entfernenden Knoten auf, wie hier gezeigt:

- **ClusterName**— Die ID des Clusters, aus dem Sie Knoten entfernen möchten.
- **ReplicaConfiguration**— Ermöglicht es Ihnen, die Anzahl der Replikate festzulegen:
 - **ReplicaCount**— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Replikatknoten anzugeben.
- **Region**— Gibt die AWS-Region des Clusters an, aus dem Sie einen Knoten entfernen möchten.

Weitere Informationen finden Sie unter [UpdateCluster](#).

Zugriff auf Ihren Cluster

Auf Ihre MemoryDB for Redis-Instances kann über eine Amazon EC2 EC2-Instance zugegriffen werden.

Sie können von einer Amazon EC2-Instance in derselben Amazon VPC auf Ihren MemoryDB-Knoten zugreifen. Oder indem Sie VPC-Peering verwenden, können Sie von einem Amazon EC2 in einer anderen Amazon VPC von einem Amazon EC2 aus auf Ihren MemoryDB-Knoten zugreifen.

Themen

- [Gewähren Sie Zugriff auf Ihren Cluster](#)
- [Zugriff auf MemoryDB-Ressourcen von außerhalbAWS](#)

Gewähren Sie Zugriff auf Ihren Cluster

Sie können nur über eine Amazon EC2-Instance, die in derselben Amazon VPC ausgeführt wird, eine Verbindung zu Ihrem MemoryDB-Cluster herstellen. In diesem Fall müssen Sie Netzwerkzugang zum Cluster gewähren.

So gewähren Sie einem Cluster den Netzwerkeingang aus einer Amazon-VPC-Sicherheitsgruppe

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon-EC2-Konsole unter <https://console.aws.amazon.com/EC2/>.
2. Im linken Navigationsbereich unter Netzwerk & Sicherheit, wählen Sicherheitsgruppen aus.
3. Wählen Sie aus der Liste der Sicherheitsgruppen die Sicherheitsgruppe Ihrer Amazon VPC aus. Sofern Sie keine Sicherheitsgruppe für die Verwendung von MemoryDB erstellt haben, wird diese Sicherheitsgruppe genannt Standard aus.
4. Wählen Sie die Registerkarte Inbound und verfahren Sie dann wie folgt:
 - a. Wählen Sie Edit aus.
 - b. Wählen Sie Add rule.
 - c. Wählen Sie in der Spalte Type die Option Custom TCP Rule aus.
 - d. Geben Sie in das Feld Port Range die Portnummer Ihres Clusterknotens ein. Diese Nummer muss mit der Nummer übereinstimmen, die Sie beim Starten des Clusters angegeben haben. Der Standardport für Redis ist **6379** aus.

- e. In der `Source`-Feld, wählen Sie `Überall`. Da sie den Portbereich (0.0.0.0/0) umfasst, kann sich jede Amazon EC2 Instance, die Sie innerhalb Ihrer Amazon VPC starten, mit Ihren MemoryDB-Knoten verbinden.

 **Important**

Öffnen des MemoryDB-Clusters für 0.0.0.0/0 macht den Cluster nicht dem Internet zugänglich, da für ihn keine öffentliche IP-Adresse vorhanden ist und daher von außerhalb der VPC nicht auf ihn zugegriffen werden kann. Die Standardsicherheitsgruppe kann jedoch auf andere Amazon-EC2-Instances im Kundenkonto angewendet werden und diese Instances können eine öffentliche IP-Adresse haben. Wenn diese Instances zufälligerweise eine Aktion auf dem Standardport ausführen, dann könnte dieser Service unbeabsichtigt zugänglich gemacht werden. Daher wird empfohlen, eine VPC-Sicherheitsgruppe einzurichten, die ausschließlich von MemoryDB genutzt wird. Weitere Informationen finden Sie unter [Benutzerdefinierte Sicherheitsgruppen](#).

- f. Wählen Sie `Save` (Speichern) aus.

Wenn Sie eine Amazon EC2 Instance in Ihrer Amazon VPC starten, kann diese Instance eine Verbindung zu Ihrem MemoryDB-Cluster herstellen.

Zugriff auf MemoryDB-Ressourcen von außerhalbAWS

MemoryDB ist ein Service, der für die interne Verwendung in Ihrer VPC konzipiert ist. Von einem externen Zugriff wird wegen der Latenz des Internetverkehrs und aufgrund von Sicherheitsbedenken abgeraten. Sollte jedoch ein externer Zugriff auf MemoryDB zu Test- oder Entwicklungszwecken erforderlich sein, kann dies über ein VPN erfolgen.

Verwendung derAWSClient VPN ermöglichen Sie den externen Zugriff auf Ihre MemoryDB-Knoten. Dadurch bieten sich folgende Vorteile:

- Eingeschränkter Zugriff auf zugelassene Benutzer oder Authentifizierungsschlüssel;
- Verschlüsselter Verkehr zwischen dem VPN-Client und dem AWS-VPN-Endpunkt;
- beschränkter Zugriff auf bestimmte Subnetze oder Knoten,
- einfacher Widerruf des Zugriffs von Benutzern oder Authentifizierungsschlüsseln,
- Prüfung von Verbindungen.

Die folgenden Verfahren veranschaulichen, wie Sie:

Themen

- [Anlegen einer Zertifizierungsstelle](#)
- [Konfigurieren von AWS-Client-VPN-Komponenten](#)
- [Konfigurieren des VPN-Clients](#)

Anlegen einer Zertifizierungsstelle

Es ist möglich, eine Zertifizierungsstelle (Certificate Authority, CA) mit verschiedenen Methoden oder Tools zu erstellen. Wir schlagen dazu das Dienstprogramm `easy-rsa` vor, das vom [OpenVPN](#)-Projekt bereitgestellt wird. Unabhängig von der Option, die Sie wählen, achten Sie darauf, die Schlüssel sicher zu halten. Über das folgende Verfahren werden die `easy-rsa`-Skripte heruntergeladen, die Zertifizierungsstelle und die Schlüssel zur Authentifizierung des ersten VPN-Clients erstellt:

- Um die ersten Zertifikate zu erstellen, öffnen Sie ein Terminal und gehen Sie folgendermaßen vor:
 - `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`

Es wird ein pki-Unterverzeichnis mit den Zertifikaten unter `easy-rsa` erstellt.

- Senden Sie das Serverzertifikat an den AWS-Zertifikatsmanager (ACM):
 - Wählen Sie in der ACM-Konsole die Option Certificate Manager (Zertifikatmanager) aus.
 - Wählen Sie Import Certificate (Zertifikat importieren) aus.
 - Geben Sie das in der Datei `easy-rsa/pki/issued/server.crt` zur Verfügung stehende Zertifikat des öffentlichen Schlüssels in das Feld Certificate body (Zertifikattext) ein.
 - Fügen Sie den unter `easy-rsa/pki/private/server.key` zur Verfügung stehenden privaten Schlüssel in das Feld Private Certificate Key (Privater Zertifikatsschlüssel) ein. Achten Sie darauf, dass Sie alle Zeilen zwischen BEGIN AND END PRIVATE KEY (einschließlich der Zeilen BEGIN und END) auswählen.
 - Fügen Sie den in der Datei `easy-rsa/pki/ca.crt` verfügbaren öffentlichen CA-Schlüssel in das Feld Certificate chain (Zertifikatskette) ein.
 - Wählen Sie die Option Review and import (Überprüfen und Importieren) aus.
 - Klicken Sie auf Import (Importieren).

Um die Zertifikate des Servers über den AWS-CLI an ACM zu übermitteln, führen Sie den folgenden Befehl aus: `aws acm import-certificate --certificate file://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`

Notieren Sie sich den Zertifikats-ARN für eine spätere Verwendung.

Konfigurieren von AWS-Client-VPN-Komponenten

Unter Verwendung der AWS-Konsole

Wählen Sie in der AWS-Konsole Dienste und dann VPC.

Wählen Sie unter Virtual Private Network (Virtuelles privates Netzwerk) die Option Client VPN Endpoints (Client-VPN-Endpunkte) aus und führen Sie die folgenden Schritte aus:

Konfigurieren von AWS-Client-VPN-Komponenten

- Wählen Sie **Create Client VPN Endpoint (Client-VPN-Endpunkt erstellen)** aus.
- Folgende Optionen stehen Ihnen zur Verfügung:
 - **Client IPv4 CIDR:** Verwenden Sie ein privates Netzwerk mit einer Netzwerkmaske in einem Bereich von mindestens /22. Stellen Sie sicher, dass das ausgewählte Subnetz nicht mit den Adressen der VPC-Netzwerke in Konflikt steht. Beispiel: 10.0.0.0/22.
 - Wählen Sie unter **Server certificate ARN (Serverzertifikat-ARN)** den ARN des zuvor importierten Zertifikats aus.
 - Wählen Sie die Option **Use mutual authentication (Gegenseitige Authentifizierung verwenden)** aus.
 - Wählen Sie unter **Client certificate ARN (Client-Zertifikats-ARN)** den ARN des zuvor importierten Zertifikats aus.
 - Wählen Sie **Create Client VPN Endpoint (Client-VPN-Endpunkt erstellen)** aus.

Verwendung der AWS CLI

Führen Sie den Befehl aus:

```
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
```

Beispielausgabe:

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

Zuordnen der Zielnetzwerke zum VPN-Endpunkt

- Wählen Sie den neuen VPN-Endpunkt und dann die Registerkarte **Associations (Zuordnungen)** aus.
- Wählen Sie **Associate (Zuordnen)** aus und nutzen Sie die folgenden Optionen:
 - **VPC:** Wählen Sie die VPC des MemoryDB-Clusters aus.

- Wählen Sie eines der Netzwerke des MemoryDB-Clusters aus. Überprüfen Sie im Zweifelsfall die Netzwerke im Subnetzgruppenim MemoryDB-Dashboard.
- Wählen Sie Associate (Zuordnen) aus. Wiederholen Sie ggf. die Schritte für die verbleibenden Netzwerke.

Verwendung der AWS CLI

Führen Sie den Befehl aus:

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

Beispielausgabe:

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

Überprüfen der VPN-Sicherheitsgruppe

Der VPN-Endpunkt übernimmt automatisch die Standard-Sicherheitsgruppe der VPC. Überprüfen Sie die Regeln für ein- und ausgehenden Datenverkehr und bestätigen Sie, ob die Sicherheitsgruppe den Datenverkehr vom VPN-Netzwerk (definiert in den Einstellungen für den VPN-Endpunkt) zu den MemoryDB-Netzwerken an den Service-Ports (standardmäßig 6379 für Redis) zulässt.

Wenn Sie die Sicherheitsgruppe ändern müssen, die dem VPN-Endpunkt zugewiesen ist, gehen Sie wie folgt vor:

- Wählen Sie die aktuelle Sicherheitsgruppe aus.
- Klicken Sie auf Apply Security Group (Sicherheitsgruppe anwenden).
- Wählen Sie die neue Sicherheitsgruppe aus.

Verwendung der AWS CLI

Führen Sie den Befehl aus:

```
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
```

Beispielausgabe:

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

Note

Die MemoryDB-Sicherheitsgruppe muss auch Datenverkehr von den VPN-Clients zulassen. Die Adressen der Clients werden entsprechend dem VPC-Netzwerk mit der VPN-Endpunktadresse maskiert. Berücksichtigen Sie daher das VPC-Netzwerk (nicht das Netzwerk der VPN-Clients), wenn Sie die Regel für eingehenden Datenverkehr für die MemoryDB-Sicherheitsgruppe erstellen.

Autorisieren des VPN-Zugriffs auf die Zielnetzwerke

Wählen Sie auf der Registerkarte Authorization (Autorisierung) die Option Authorize Ingress (Autorisierung eingehender Daten) aus und geben Sie Folgendes an:

- Zielnetzwerk zum Aktivieren des Zugriffs: Verwenden Sie entweder 0.0.0.0/0, um den Zugriff auf ein beliebiges Netzwerk (einschließlich des Internets) zu ermöglichen, oder beschränken Sie die MemoryDB-Netzwerke/-Hosts.
- Wählen Sie unter Grant access to: (Zugriff gewähren für:) die Option Allow access to all users (Zugriff für alle Benutzer zulassen) aus.
- Wählen Sie Add Authorization Rules (Autorisierungsregeln hinzufügen) aus.

Verwendung der AWS CLI

Führen Sie den Befehl aus:

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

Beispielausgabe:

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

Ermöglichen des Zugriffs auf das Internet über VPN-Clients

Wenn Sie über das VPN im Internet surfen müssen, müssen Sie eine zusätzliche Route erstellen. Wählen Sie die Registerkarte Route Table (Routentabelle) und dann die Option Create Route (Route erstellen) aus:

- Ziel der Route: 0.0.0.0/0
- Ziel-VPN-Subnetz-ID: Wählen Sie eines der zugeordneten Subnetze mit Zugang zum Internet aus.
- Klicken Sie auf Create Route (Route erstellen).

Verwendung der AWS CLI

Führen Sie den Befehl aus:

```
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-0123456789abcdef
```

Beispielausgabe:

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

Konfigurieren des VPN-Clients

Wählen Sie im AWS-Client-VPN-Dashboard den kürzlich erstellten VPN-Endpunkt aus und wählen Sie Client-Konfiguration herunterladen. Kopieren Sie die Konfigurationsdatei sowie die Dateien `easy-rsa/pki/issued/client1.domain.tld.crt` und `easy-rsa/pki/private/client1.domain.tld.key`. Bearbeiten Sie die Konfigurationsdatei und ändern oder fügen Sie die folgenden Parameter hinzu:

- `cert`: Fügen Sie eine neue Zeile hinzu, wobei der Parameter „`cert`“ auf die Datei `client1.domain.tld.crt` verweist. Verwenden Sie den vollständigen Pfad zu der Datei.
Beispiel: `cert /home/user/.cert/client1.domain.tld.crt`
- `cert: key`: Fügen Sie eine neue Zeile hinzu, wobei der Parameter „`key`“ auf die Datei `client1.domain.tld.key` verweist. Verwenden Sie den vollständigen Pfad zu der Datei.
Beispiel: `key /home/user/.cert/client1.domain.tld.key`

Stellen Sie die VPN-Verbindung mit folgendem Befehl her: `sudo openvpn --config downloaded-client-config.ovpn`

Aufheben des Zugriffs

Soll die Gültigkeit des Zugriffs eines bestimmten Clientschlüssels aufgehoben werden, muss der Schlüssel in der Zertifizierungsstelle widerrufen werden. Übermitteln Sie dann die Sperrliste an AWS-Client-VPN.

Widerrufen des Schlüssels mit easy-rsa:

- `cd easy-rsa`
- `./easyrsa3/easyrsa revoke client1.domain.tld`
- Geben Sie „yes“ (Ja) ein, um fortzufahren, oder nehmen Sie zum Abbrechen eine andere Eingabe vor.

```
Continue with revocation: `yes` ... * `./easyrsa3/easyrsa gen-crl
```

- Es wurde eine aktualisierte CRL erstellt. CRL-Datei: `/home/user/easy-rsa/pki/crl.pem`

Importieren der Sperrliste in das AWS-Client-VPN:

- Wählen Sie auf der AWS Management Console die Option Services und anschließend VPC aus.
- Wählen Sie Client VPN Endpoints (Client-VPN-Endpunkte) aus.
- Wählen Sie den Client-VPN-Endpoint aus und klicken Sie dann auf Actions (Aktionen) -> Import Client Certificate CRL (Client-Zertifikats-CRL importieren).
- Fügen Sie den Inhalt der Datei `crl.pem`-Datei ein:

Verwendung der AWS CLI

Führen Sie den Befehl aus:

```
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
```

Beispielausgabe:

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

Verbindungsendpunkte ermitteln

Ihre Anwendung stellt über den Endpunkt eine Verbindung zu Ihrem Cluster her. Ein Endpunkt ist die eindeutige Adresse eines Clusters. Verwenden Sie die des Clusters `Cluster-Endpoint` für alle - Operationen.

Die folgenden Abschnitte führen Sie durch die Erkennung des Endpunkts, den Sie benötigen.

Ermitteln des Endpunkts für einen MemoryDB-Cluster (AWS Management Console)

So suchen Sie den Endpunkt eines MemoryDB-Clusters

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die MemoryDB for Redis-Konsole unter <https://console.aws.amazon.com/memorydb/> aus.
2. Wählen Sie im Navigationsbereich Cluster (Cluster) aus.

Der Bildschirm Cluster wird mit einer Liste von Clustern angezeigt. Wählen Sie den Cluster aus, mit dem Sie sich verbinden möchten.

3. Um den Endpunkt des Clusters zu suchen, wählen Sie den Namen des Clusters (nicht das Optionsfeld) aus.
4. Der Cluster-Endpunkt wird angezeigt unter Cluster-Details aus. Um sie zu kopieren, wählen Sie den Kopie-Symbol links vom Endpunkt.

Ermitteln des Endpunkts für einen MemoryDB-Cluster (AWSCLI)

Sie können das `describe-clusters`-Befehl zum Ermitteln des Endpunkts für einen Cluster. Mit dem Befehl lässt sich der Endpunkt des Clusters zurückgeben

Der folgende Vorgang ruft den Endpunkt ab, der in diesem Beispiel als *Probe*, für den Cluster `mycluster` aus.

Sie gibt die folgende JSON-Antwort zurück:

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

Für 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
  }
]
}
```

Weitere Informationen finden Sie unter [describe-clusters](#) aus.

Ermitteln des Endpunkts für einen MemoryDB-Cluster (MemoryDB-API)

Sie können mit der MemoryDB for Redis-API den Endpunkt eines Clusters ermitteln.

Ermitteln des Endpunkts für einen MemoryDB-Cluster (MemoryDB-API)

Sie können die MemoryDB-API verwenden, um den Endpunkt für einen -Cluster mit dem `DescribeClusters` Aktion. Die Aktion gibt den Endpunkt des Clusters zurück.

Mit dem folgenden Vorgang wird der Cluster-Endpunkt für den -Cluster abgerufen `mycluster` aus.

```
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>
```

Weitere Informationen finden Sie unter [DescribeClusters](#) aus.

Arbeiten mit Shards

Ein Shard ist eine Sammlung von ein bis 6 Knoten. Sie können einen Cluster mit einer höheren Anzahl an Shards und einer geringeren Anzahl an Replikaten mit bis zu 500 Knoten pro Cluster erstellen. Diese Cluster-Konfiguration reicht von 500 Shards und 0 Replikaten bis hin zu 100 Shards und 4 Replikaten, was dem Höchstwert für die Anzahl erlaubter Replikat entspricht. Die Daten des Clusters werden über die Shards des Clusters hinweg partitioniert. Falls sich mehr als ein Knoten in einem Shard befindet, implementiert der Shard die Replikation auf dem primären Knoten für Lese- und Schreibvorgänge. Die anderen sind schreibgeschützte Replikatknoten.

Wenn Sie einen MemoryDB-Cluster erstellen AWS Management Console, geben Sie die Anzahl der Shards im Cluster und die Anzahl der Knoten in den Shards an. Weitere Informationen finden Sie unter [Einen MemoryDB-Cluster erstellen](#).

Für jeden Knoten in einem Shard gelten dieselben Rechner-, Arbeitsspeicher- und Festspeicherspezifikationen. Mit der MemoryDB-API können Sie cluster-weite Attribute wie die Anzahl der Knoten, Sicherheitseinstellungen und Systemwartungsfenster steuern.

Weitere Informationen erhalten Sie unter [Offline-Resharding und Shard-Rebalancing für MemoryDB](#) und [Online-Resharding und Shard-Rebalancing für MemoryDB](#).

Auffinden der Namen eines Shards

Sie können mithilfe der, der AWS CLI oder der AWS Management Console MemoryDB-API nach.

Verwendung der AWS Management Console

Das folgende Verfahren verwendet den AWS Management Console, um die Shard-Namen eines MemoryDB-Clusters zu finden.

1. [Melden Sie sich bei der Konsole von MemoryDB for Redis an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](#)
2. Wählen Sie im Navigationsbereich die Option Cluster auf.
3. Wählen Sie unter Name den Cluster aus, dessen Shard-Namen Sie finden möchten.
4. Sehen Sie sich auf der Registerkarte Shards und Nodes die Liste der Shards unter Name an. Sie können die einzelnen Knoten auch erweitern, um Details zu ihren Knoten anzuzeigen.

Verwendung der AWS CLI

Um Shard-Namen (Shard) für MemoryDB-Cluster zu finden, verwenden Sie die AWS CLI Operation `describe-clusters` mit dem folgenden optionalen Parameter.

- **--cluster-name**—Ein optionaler Parameter, der bei Verwendung die Ausgabe auf die Details des angegebenen Clusters beschränkt. Wenn dieser Parameter weggelassen wird, werden die Details von bis zu 100 Clustern zurückgegeben.
- **--show-shard-details**— Gibt Details zu den Shards zurück, einschließlich ihrer Namen.

Dieser Befehl gibt die Details für `my-cluster` zurück.

Für Linux, macOS oder Unix:

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

Für Windows:

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

Sie gibt die folgende JSONAntwort zurück:

Die Zeilenumbrüche dienen der besseren Lesbarkeit.

```
{  
  "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
}
]
}

```

Verwenden der der API

Verwenden Sie die API-Operation `DescribeClusters` mit dem folgenden optionalen Parameter, um Shard-IDs für MemoryDB-Cluster zu finden.

- **ClusterName**—Ein optionaler Parameter, der bei Verwendung die Ausgabe auf die Details des angegebenen Clusters beschränkt. Wenn dieser Parameter weggelassen wird, werden die Details von bis zu 100 Clustern zurückgegeben.
- **ShowShardDetails**— Gibt Details zu den Shards zurück, einschließlich ihrer Namen.

Example

Dieser Befehl gibt die Details für `my-cluster` zurück.

Für Linux, macOS oder 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>
```

Verwaltung Ihrer MemoryDB-Implementierung

In diesem Abschnitt finden Sie Einzelheiten zur Verwaltung der verschiedenen Komponenten Ihrer MemoryDB-Implementierung.

Themen

- [Redis-Engine-Versionen](#)
- [Erste Schritte mit JSON](#)
- [Markierung Ihrer MemoryDB-Ressourcen](#)
- [Verwaltung der Wartung](#)
- [Bewährte Methoden](#)
- [Grundlegendes zur MemoryDB-Replikation](#)
- [Snapshot und Wiederherstellung](#)
- [Skalierung](#)
- [Konfiguration von Engine-Parametern unter Verwendung von Parametergruppen](#)
- [Tutorial: Konfiguration einer Lambda-Funktion für den Zugriff auf MemoryDB in einer Amazon VPC](#)

Redis-Engine-Versionen

Dieser Abschnitt behandelt die unterstützten Redis-Engine-Versionen.

Themen

- [MemoryDB für Redis Version 7.1 \(erweitert\)](#)
- [MemoryDB für Redis Version 7.0 \(erweitert\)](#)
- [MemoryDB für Redis Version 6.2 \(erweitert\)](#)
- [Upgrade von Engine-Versionen](#)

MemoryDB für Redis Version 7.1 (erweitert)

MemoryDB for Redis Version 7.1 bietet Unterstützung für Vektorsuchfunktionen in der Vorschauversion für ausgewählte Regionen sowie wichtige Bugfixes und Leistungsverbesserungen.

- **Vektorsuchfunktion: Die** Vektorsuche kann mit der vorhandenen MemoryDB-Funktionalität verwendet werden. Anwendungen, die die Vektorsuche nicht verwenden, sind von ihrer Präsenz nicht betroffen. Die Vorschau der Vektorsuche ist ab MemoryDB for Redis Version 7.1 in den folgenden Regionen verfügbar: USA Ost (Nord-Virginia und Ohio), USA West (Oregon), EU (Irland) und Asien-Pazifik (Tokio). In der Dokumentation [hier](#) finden Sie Informationen zur Aktivierung der Vektorsuchvorschau und der zugehörigen Funktionen.

Note

MemoryDB for Redis Version 7.1 ist mit OSS Redis v7.0 kompatibel. Weitere Informationen zur Version Redis 7.0 finden Sie in den Versionshinweisen zu Redis 7.0 unter [Redis](#) am GitHub

MemoryDB für Redis Version 7.0 (erweitert)

MemoryDB for Redis 7.0 bietet eine Reihe von Verbesserungen und Unterstützung für neue Funktionen:

- **Redis-Funktionen:** MemoryDB für Redis 7 bietet Unterstützung für Redis-Funktionen und bietet eine verwaltete Oberfläche, mit der Entwickler [LUA-Skripts mit auf dem MemoryDB-Cluster gespeicherter Anwendungslogik ausführen können, ohne dass Clients die Skripts](#) bei jeder Verbindung erneut an den Server senden müssen.
- **ACL-Verbesserungen:** MemoryDB für Redis 7 bietet Unterstützung für die nächste Version von Redis Access Control Lists (ACLs). Mit MemoryDB für Redis 7 können Kunden jetzt mehrere Berechtigungssätze für bestimmte Schlüssel oder Schlüsselräume in Redis angeben.
- **Sharded Pub/Sub:** MemoryDB für Redis 7 bietet Unterstützung für die gemeinsame Ausführung der Redis Pub/Sub-Funktionalität, wenn MemoryDB im Cluster Mode Enabled (CME) ausgeführt wird. Die Pub/Sub-Funktionen von Redis ermöglichen es Herausgebern, Nachrichten an eine beliebige Anzahl von Abonnenten eines Kanals zu senden. Mit Amazon MemoryDB for Redis 7 sind Kanäle an einen Shard im MemoryDB-Cluster gebunden, sodass Kanalinformationen nicht mehr zwischen Shards weitergegeben werden müssen. Dies führt zu einer verbesserten Skalierbarkeit.
- **Verbessertes I/O-Multiplexing:** MemoryDB for Redis Version 7 bietet erweitertes I/O-Multiplexing, das einen höheren Durchsatz und eine geringere Latenz für Workloads mit hohem Durchsatz und vielen gleichzeitigen Client-Verbindungen zu einem MemoryDB-Cluster bietet. Wenn Sie beispielsweise einen Cluster von r6g.4xlarge-Knoten verwenden und 5200 Clients gleichzeitig

ausführen, können Sie im Vergleich zu MemoryDB for Redis Version 6 einen um bis zu 46% höheren Durchsatz (Lese- und Schreibvorgänge pro Sekunde) und eine um bis zu 21% verringerte P99-Latenz erzielen.

[Weitere Informationen zur Version Redis 7.0 finden Sie in den Versionshinweisen zu Redis 7.0 unter Redis on. GitHub](#)

MemoryDB für Redis Version 6.2 (erweitert)

MemoryDB stellt die nächste Version der Redis-Engine vor, die automatische Unterstützung für Versionsupgrades [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#), clientseitiges Caching und erhebliche betriebliche Verbesserungen umfasst.

Die Redis-Engine-Version 6.2.6 bietet auch Unterstützung für das native JSON-Format (JavaScript Object Notation), eine einfache, schemalose Methode zur Kodierung komplexer Datensätze innerhalb von Redis-Clustern. Mit der JSON-Unterstützung können Sie die Leistung und die Redis-APIs für Anwendungen nutzen, die über JSON betrieben werden. Weitere Informationen finden Sie unter [Erste Schritte mit JSON](#). Ebenfalls enthalten ist eine JSON-bezogene Metrik, die in die Überwachung der Verwendung `JsonBasedCmds` dieses Datentyps integriert ist. CloudWatch Weitere Informationen finden Sie unter [Metriken für MemoryDB](#).

Mit Redis 6 wird MemoryDB eine einzige Version für jede Redis OSS-Nebenversion anbieten, anstatt mehrere Patch-Versionen anzubieten. Dadurch sollen Verwirrung und Unklarheiten vermieden werden, wenn Sie aus mehreren Nebenversionen wählen müssen. MemoryDB verwaltet außerdem automatisch die Minor- und Patch-Version Ihrer laufenden Cluster und sorgt so für eine verbesserte Leistung und erhöhte Sicherheit. Dies wird über die üblichen Kanäle zur Kundenbenachrichtigung im Rahmen einer Service-Update-Kampagne abgewickelt. Weitere Informationen finden Sie unter [Service-Updates in MemoryDB für Redis](#).

Wenn Sie die Engine-Version bei der Erstellung nicht angeben, wählt MemoryDB automatisch die bevorzugte Redis-Version für Sie aus. Wenn Sie andererseits die Engine-Version mithilfe von angeben, ruft MemoryDB automatisch die bevorzugte Patch-Version von Redis 6.2 auf 6.2, die verfügbar ist.

Wenn Sie beispielsweise einen Cluster erstellen, setzen Sie den Parameter auf `--engine-version 6.2`. Der Cluster wird zum Zeitpunkt der Erstellung mit der aktuell verfügbaren bevorzugten Patch-Version gestartet. Jede Anfrage mit einem Wert für die vollständige Engine-Version wird abgelehnt, es wird eine Ausnahme ausgelöst und der Vorgang schlägt fehl.

Beim Aufrufen der `DescribeEngineVersions` API wird der `EngineVersion` Parameterwert auf 6.2 gesetzt und die tatsächliche Vollversion der Engine wird im `EnginePatchVersion` Feld zurückgegeben.

Weitere Informationen zur Version Redis 6.2 finden Sie in den [Versionshinweisen zu Redis 6.2](#) unter [Redis on GitHub](#)

Upgrade von Engine-Versionen

MemoryDB verwaltet die Patch-Version Ihrer laufenden Cluster standardmäßig automatisch über Service-Updates. Sie können sich auch vom auto Upgrade der Nebenversion abmelden, wenn Sie die `AutoMinorVersionUpgrade` Eigenschaft Ihrer Cluster auf `False` setzen. Sie können sich jedoch nicht vom auto Patch-Versionsupgrade abmelden.

Sie können steuern, ob und wann die protokollkonforme Software, die Ihren Cluster unterstützt, auf neue Versionen aktualisiert wird, die von MemoryDB unterstützt werden, bevor das auto Upgrade gestartet wird. Mit diesem Maß an Kontrolle können Sie die Kompatibilität mit bestimmten Versionen aufrechterhalten, neue Versionen mit Ihrer Anwendung testen, bevor Sie sie für die Produktion bereitstellen, und Versions-Upgrades nach Ihren eigenen Vorgaben und Zeitplänen durchführen lassen.

Sie können Engine-Versions-Upgrades für Ihren Cluster auf folgende Weise initiieren:

- Indem Sie es aktualisieren und eine neue Engine-Version angeben. Weitere Informationen finden Sie unter [Einen MemoryDB-Cluster ändern](#).
- Anwenden des Service-Updates für die entsprechende Engine-Version. Weitere Informationen finden Sie unter [Service-Updates in MemoryDB für Redis](#).

Beachten Sie Folgendes:

- Sie können zwar auf eine neue Engine-Version upgraden, jedoch kein Downgrade auf eine ältere Engine-Version ausführen. Wenn Sie eine ältere Engine-Version verwenden möchten, müssen Sie den vorhandenen Cluster löschen und mit der älteren Engine-Version neu erstellen.
- Wir empfehlen, regelmäßig auf die neueste Hauptversion zu aktualisieren, da die meisten wichtigen Verbesserungen nicht auf ältere Versionen zurückportiert werden. Da MemoryDB die Verfügbarkeit auf eine neue AWS Region ausdehnt, unterstützt MemoryDB die beiden jeweils neuesten MAJOR.MINOR Versionen für die neue Region. Wenn beispielsweise eine neue AWS Region eingeführt wird und die neuesten Versionen von MAJOR.MINOR MemoryDB für Redis 7.0

und 6.2 sind, unterstützt MemoryDB for Redis die Versionen 7.0 und 6.2 in der neuen Region. AWS Sobald neuere MAJOR.MINOR Versionen von MemoryDB for Redis veröffentlicht werden, wird MemoryDB weiterhin Unterstützung für die neu veröffentlichten Versionen von MemoryDB for Redis hinzufügen. Weitere Informationen zur Auswahl von Regionen für MemoryDB finden Sie unter [Unterstützte Regionen und Endpunkte](#)

- Versionsverwaltung der Engine ist so entwickelt, dass Sie so viel Kontrolle wie möglich darüber haben, wie Patches erfolgt. MemoryDB behält sich jedoch das Recht vor, Ihren Cluster in Ihrem Namen zu patchen, falls es zu einer kritischen Sicherheitslücke im System oder in der Software kommen sollte.
- MemoryDB wird für jede Redis OSS-Nebenversion eine einzige Version anbieten, anstatt mehrere Patch-Versionen anzubieten. Dadurch sollen Verwirrung und Unklarheiten vermieden werden, wenn Sie aus mehreren Versionen wählen müssen. MemoryDB verwaltet außerdem automatisch die Minor- und Patch-Version Ihrer laufenden Cluster und sorgt so für eine verbesserte Leistung und erhöhte Sicherheit. Dies wird über die üblichen Kanäle zur Kundenbenachrichtigung im Rahmen einer Service-Update-Kampagne abgewickelt. Weitere Informationen finden Sie unter [Service-Updates in MemoryDB für Redis](#).
- Sie können Ihre Cluster-Version mit minimaler Ausfallzeit aktualisieren. Der Cluster kann während des gesamten Upgrades gelesen und in der Regel auch beschrieben werden, ausgenommen während der Failover-Operation, der nur einige Sekunden dauert.
- Wir empfehlen, Engine-Upgrades in Zeiten mit geringem eingehendem Schreibverkehr durchzuführen.

Cluster mit mehreren Shards werden wie folgt verarbeitet und gepatcht:

- Pro Shard wird jeweils nur ein Upgrade-Vorgang durchgeführt.
- In jedem Shard werden alle Replicas verarbeitet, bevor der Primärknoten verarbeitet wird. Wenn es in einem Shard weniger Replicas gibt, kann der Primärknoten in diesem Shard verarbeitet werden, bevor die Verarbeitung der Replicas in anderen Shards abgeschlossen wird.
- Die Primärknoten für alle Shards werden seriell verarbeitet. Es erfolgt jeweils nur ein Upgrade für einen Primärknoten gleichzeitig.

Themen

- [So führen Sie ein Upgrade von Engine-Versionen aus](#)
- [Beheben blockierter Redis-Engine-Upgrades](#)

So führen Sie ein Upgrade von Engine-Versionen aus

Sie initiieren Versionsupgrades für Ihren Cluster, indem Sie ihn mithilfe der MemoryDB-Konsole, der AWS CLI, oder der MemoryDB-API ändern und eine neuere Engine-Version angeben. Weitere Informationen finden Sie unter den folgenden Themen.

- [Verwendung von AWS Management Console](#)
- [Verwenden der AWS CLI](#)
- [Verwenden der MemoryDB-API](#)

Beheben blockierter Redis-Engine-Upgrades

Wie in der folgenden Tabelle dargestellt, wird das Upgrade Ihrer Redis-Engine blockiert, wenn eine Skalierungsoperation aussteht.

Ausstehende Vorgänge	Blockierte Vorgänge
Aufwärtsskalierung	Unmittelbares Engine-Upgrade
Engine-Upgrade	Unmittelbares Aufwärtsskalieren
Aufwärtsskalierung und Engine-Upgrade	Unmittelbares Aufwärtsskalieren
	Unmittelbares Engine-Upgrade

Erste Schritte mit JSON

MemoryDB unterstützt das native JSON-Format (JavaScriptObject Notation), welches eine einfache schemalose Methode zum Codieren komplexer Datensätze innerhalb von Redis-Clustern ist. Sie können Daten nativ mithilfe des JSON-Formats (JavaScriptObject Notation) in Redis-Clustern speichern, und darauf zugreifen. Sie können die in diesen Clustern gespeicherten JSON-Daten aktualisieren, ohne benutzerdefinierten Code verwalten zu müssen, um ihn zu serialisieren und zu deserialisieren.

Zusätzlich zur Nutzung von Redis-APIs für Anwendungen, die über JSON arbeiten, können Sie jetzt bestimmte Teile eines JSON-Dokuments effizient abrufen und aktualisieren, ohne das gesamte

Objekt bearbeiten zu müssen, was die Leistung verbessern und die Kosten senken kann. Sie können den Inhalt Ihres JSON-Dokuments auch mit der [Goessner](#)-Abfrage abfragen.

Nach dem Erstellen eines Clusters mit einer unterstützten Engine-Version sind der JSON-Datentyp und die zugehörigen Befehle automatisch verfügbar. Dies ist API-kompatibel und RDB-kompatibel mit Version 2 des RedisJSON-Moduls, sodass Sie bestehende JSON-basierte Redis-Anwendungen problemlos nach MemoryDB migrieren können. Weitere Informationen zu den unterstützten Redis-Befehlen finden Sie unter [Unterstützte Befehle](#).

JSON-bezogene Metriken `JsonBasedCmds` sind integriert `CloudWatch`, um die Verwendung dieses Datentyps zu überwachen. Weitere Informationen finden Sie unter [Metriken für MemoryDB](#).

Note

Um JSON verwenden zu können, müssen Sie die Redis-Engine-Version 6.2.6 oder höher ausführen.

Themen

- [Überblick über den Redis-JSON-Datentyp](#)
- [Unterstützte Befehle](#)

Überblick über den Redis-JSON-Datentyp

MemoryDB unterstützt eine Reihe von Redis-Befehlen für die Arbeit mit dem JSON-Datentyp. Im Folgenden finden Sie eine Übersicht über den JSON-Datentyp und eine detaillierte Liste der unterstützten Redis-Befehle.

Terminologie

Begriff	Beschreibung
JSON-Dokument	bezieht sich auf den Wert eines Redis-JSON-Schlüssels
JSON-Wert	bezieht sich auf eine Teilmenge eines JSON-Dokuments, einschließlich des Stammverz

Begriff	Beschreibung
	eichnisses, das das gesamte Dokument darstellt. Ein Wert kann ein Container oder ein Eintrag innerhalb eines Containers sein.
JSON-Element	entspricht dem JSON-Wert

Unterstützter JSON-Standard

Das JSON-Format ist mit [RFC 7159](#) und dem [ECMA-404](#)-JSON-Datenaustauschstandard konform. UTF-8 [Unicode](#) wird im JSON-Text unterstützt.

Stammelement

Das Stammelement kann von jedem JSON-Datentyp stammen. Beachten Sie, dass in früheren RFC 4627 nur Objekte oder Arrays als Stammwerte zugelassen waren. Seit dem Update auf RFC 7159 kann das Stammverzeichnis eines JSON-Dokuments einen beliebigen JSON-Datentyp haben.

Begrenzung der Dokumentgröße

JSON-Dokumente werden intern in einem Format gespeichert, das für schnellen Zugriff und Änderung optimiert ist. Dieses Format verbraucht in der Regel etwas mehr Speicherplatz als die entsprechende serialisierte Darstellung desselben Dokuments. Der Speicherverbrauch eines einzelnen JSON-Dokuments ist auf 64 MB begrenzt, was der Größe der In-Memory-Datenstruktur, nicht der JSON-Zeichenfolge entspricht. Die von einem JSON-Dokument verbrauchte Speichermenge kann mithilfe des `JSON.DEBUG MEMORY` Befehls überprüft werden.

JSON-ACLs

- Der JSON-Datentyp ist vollständig in die Funktion [Redis Access Control Lists \(ACL\)](#) integriert. Ähnlich wie bei den bestehenden Kategorien pro Datentyp (`@string`, `@hash` usw.) wurde eine neue Kategorie `@json` hinzugefügt, um die Verwaltung des Zugriffs auf JSON-Befehle und -Daten zu vereinfachen. Keine anderen vorhandenen Redis-Befehle sind Mitglieder der Kategorie `@json`. Alle JSON-Befehle erzwingen alle Keyspace- oder Befehlseinschränkungen und -berechtigungen.
- Es gibt fünf bestehende Redis-ACL-Kategorien, die um die neuen JSON-Befehle ergänzt wurden: `@read`, `@write`, `@fast`, `@slow` und `@admin`. Die folgende Tabelle zeigt die Zuordnung von JSON-Befehlen zu den entsprechenden Kategorien.

ACL

JSON-Befehl	@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		

JSON-Befehl	@read	@write	@fast	@slow	@admin
JSON.NUMM ULTBY		y	y		
JSON.OBJK EYS	y		y		
JSON.OBJL EN	y		y		
JSON.RESP	y		y		
JSON.SET		y		y	
JSON.STRA PPEND		y	y		
JSON.STRL EN	y		y		
JSON.STRL EN	y		y		
JSON.TOGG LE		y	y		
JSON.TYPE	y		y		
JSON.NUMI NCRBY		y	y		

Begrenzung der Verschachtelungstiefe

Wenn ein JSON-Objekt oder Array ein Element hat, das selbst ein anderes JSON-Objekt oder Array ist, wird gesagt, dass dieses innere Objekt oder Array innerhalb des äußeren Objekts oder Arrays „verschachtelt“ wird. Die maximale Verschachtelungstiefe ist 128. Jeder Versuch, ein Dokument zu erstellen, das eine Verschachtelungstiefe von mehr als 128 enthält, wird mit einem Fehler abgelehnt.

Befehlssyntax

Die meisten Befehle benötigen einen Redis-Schlüsselnamen als erstes Argument. Einige Befehle haben auch ein Pfadargument. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Notation:

- Erforderliche Argumente sind in spitzen Klammern eingeschlossen, z. B. <key>
- Optionale Argumente sind in eckigen Klammern eingeschlossen, z. B. [path]
- Zusätzliche optionale Argumente werden durch... gekennzeichnet, z. B. [json...]

Pfadsyntax

JSON-Redis unterstützt zwei Arten von Pfadsyntaxen:

- Verbesserte Syntax — Folgt der von [Goessner](#) beschriebenen JSONPath-Syntax, wie in der folgenden Tabelle dargestellt. Wir haben die Beschreibungen in der Tabelle zur besseren Übersicht neu angeordnet und geändert.
- Beschränkte Syntax – Hat begrenzte Abfragemöglichkeiten.

Note

Die Ergebnisse einiger Befehle hängen davon ab, welche Art von Pfadsyntax verwendet wird.

Wenn ein Abfragepfad mit „\$“ beginnt, verwendet er die erweiterte Syntax. Andernfalls wird eine eingeschränkte Syntax verwendet.

Verbesserte Syntax

Symbol/Ausdruck	Beschreibung
\$	das Root-Element
. oder []	Kinderbetreuer
..	rekursiver Abstieg

Symbol/Ausdruck	Beschreibung
*	platzhalter. Alle Elemente in einem Objekt oder Array.
[]	Subskript-Operator für Arrays. Der Index basiert auf 0.
[.]	gewerkschaftlicher Betreiber
[start:end:step]	Array-Slice-Operator
?()	wendet einen Filterausdruck (Skript) auf das aktuelle Array oder Objekt an
()	Filterausdruck
@	wird in Filterausdrücken verwendet, die sich auf den aktuellen Knoten beziehen, der gerade verarbeitet wird
==	entspricht, wird in Filterausdrücken verwendet.
!=	nicht gleich, wird in Filterausdrücken verwendet .
>	größer als, wird in Filterausdrücken verwendet.
>=	größer als oder gleich, gilt der Filterausdruck für den Filterausdrücken.
<	kleiner als, wird in Filterausdrücken verwendet.
<=	kleiner als oder gleich, wird in Filterausdrücken verwendet.
&&	logisches UND, wird verwendet, um mehrere Filterausdrücke zu kombinieren.

Symbol/Ausdruck	Beschreibung
	logisches ODER, wird verwendet, um mehrere Filterausdrücke zu kombinieren.

Beispiele

Die folgenden Beispiele basieren auf [Goessners](#) Beispiel-XML-Daten, die wir durch Hinzufügen zusätzlicher Felder geändert haben.

```
{ "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
    }
  }
}

```

Pfad	Beschreibung
<code>\$.store.book[*].author</code>	die Autoren aller Bücher im Laden
<code>\$.author</code>	alle Autoren
<code>\$.store.*</code>	alle Mitglieder des Ladens
<code>\$["store"].*</code>	alle Mitglieder des Ladens
<code>\$.store..price</code>	der Preis von allem im Laden
<code>\$.*</code>	alle rekursiven Mitglieder der JSON-Struktur
<code>\$.book[*]</code>	alle Bücher
<code>\$.book[0]</code>	das erste Buch
<code>\$.book[-1]</code>	das letzte Buch
<code>\$.book[0:2]</code>	Die ersten beiden Bücher
<code>\$.book[0,1]</code>	Die ersten beiden Bücher
<code>\$.book[0:4]</code>	Bücher von Index 0 bis 3 (der Endindex schließt nicht mit ein)
<code>\$.book[0:4:2]</code>	Bücher im Index 0, 2
<code>\$.book[?(@.isbn)]</code>	alle Bücher mit ISBN-Nummer
<code>\$.book[?(@.price<10)]</code>	alle Bücher billiger als 10\$

Pfad	Beschreibung
'\$..book[?(@.price < 10)]'	alle Bücher sind billiger als 10\$. (Der Pfad muss in Anführungszeichen gesetzt werden, wenn er Leerzeichen enthält)
'\$..book[?(@["price"] < 10)]'	alle Bücher billiger als 10\$
'\$..book[?(@["price"] < 10)]'	alle Bücher billiger als 10\$
\$.book[?(@.price>=10&&@.price<=100)]	alle Bücher im Preisbereich von 10 bis 100\$, einschließlich
'\$..book[?(@.price>=10 && @.price<=100)]'	alle Bücher im Preisbereich von 10 bis 100\$, einschließlich. (Der Pfad muss in Anführungszeichen gesetzt werden, wenn er Leerzeichen enthält)
\$.book[?(@.sold==true @.in-stock==false)]	alle Bücher verkauft oder vergriffen
'\$..book[?(@.sold == true @.in-stock == false)]'	alle Bücher verkauft oder vergriffen. (Der Pfad muss in Anführungszeichen gesetzt werden, wenn er Leerzeichen enthält)
'\$.store.book[?(@["category"] == "fiction")]'	alle Bücher der Kategorie Belletristik
'\$.store.book[?(@["category"] != "fiction")]'	alle Bücher der Sachbuch-Kategorien

Weitere Beispiele für Filterausdrücke:

```
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]"
```

Beschränkte Syntax

Symbol/Ausdruck	Beschreibung
. oder []	Kinderbetreuer
[]	Subskript-Operator für Arrays. Der Index basiert auf 0.

Beispiele

Pfad	Beschreibung
.store.book[0].author	der Autor des ersten Buches
.store.book[-1].author	der Autor des letzten Buches
.address.city	Name der Stadt
["store"]["book"][0]["title"]	der Titel des ersten Buches

Pfad	Beschreibung
<code>["store"]["book"][-1]["title"]</code>	der Titel des letzten Buches

Note

Alle [Goessner](#)-Inhalte, die in dieser Dokumentation erwähnt werden, unterliegen der [Creative-Commons-Lizenz](#).

Häufige Fehlerpräfixe

Jede Fehlermeldung hat ein Präfix. Im Folgenden finden Sie eine Liste der häufigsten Fehlerpräfixe:

Präfix	Beschreibung
ERR	ein allgemeiner Fehler
LIMIT	Fehler bei Überschreitung der Größenbegrenzung. Beispiel: Die Dokumentgrößenbegrenzung oder die Verschachtelungstiefenbegrenzung wurde überschritten
NONEXISTENT	ein Schlüssel oder Pfad existiert nicht
OUTOFBOUNDARIES	Array-Index außerhalb der Grenzen
SYNTAXERR	syntax-Fehler
WRONGTYPE	falscher Werttyp

JSON-bezogene Metriken

Die folgenden JSON-Infometriken werden bereitgestellt:

Informationen	Beschreibung
<code>json_total_memory_bytes</code>	Gesamt-Speicherzuteilung für JSON-Objekte
<code>json_num_documents</code>	Gesamtzahl der Dokumente in Redis

Führen Sie den Befehl Redis aus, um Kernmetriken abzufragen:

```
info json_core_metrics
```

Wie MemoryDB mit JSON interagiert

Im Folgenden wird veranschaulicht, wie MemoryDB mit dem JSON-Datentyp interagiert.

Rangfolge der Operatoren

Bei der Bewertung bedingter Ausdrücke zum Filtern, haben `&&s` zuerst Vorrang, und dann werden `||s` ausgewertet, wie es in den meisten Sprachen üblich ist. Operationen innerhalb von Klammern werden zuerst ausgeführt.

Verhalten der maximalen Verschachtelungsbeschränkung

Die maximale Pfadverschachtelungstiefe von MemoryDB ist 128. Ein Wert wie `$.a.b.c.d...` kann also nur 128 Level erreichen.

Umgang mit numerischen Werten

JSON hat keine separaten Datentypen für Ganzzahlen und Fließkommazahlen. Sie werden alle Zahlen genannt.

Wenn eine JSON-Nummer empfangen wird, wird sie in einem von zwei Formaten gespeichert. Wenn die Zahl in eine 64-Bit-Ganzzahl mit Vorzeichen passt, wird sie in dieses Format konvertiert. Andernfalls wird sie als Zeichenfolge gespeichert. Arithmetische Operationen an zwei JSON-Zahlen (z. B. `JSON.NUMINCRBY` und `JSON.NUMMULTBY`) versuchen, so viel Genauigkeit wie möglich beizubehalten. Wenn die beiden Operanden und der resultierende Wert in eine 64-Bit-Ganzzahl mit Vorzeichen passen, wird eine Ganzzahlarithmetik durchgeführt. Andernfalls werden die Eingabeoperanden in 64-Bit-IEEE-Fließkommazahlen mit doppelter Genauigkeit umgewandelt, die arithmetische Operation wird ausgeführt und das Ergebnis wird wieder in eine Zeichenfolge umgewandelt.

Arithmetische Befehle NUMINCRBY und NUMMULTBY:

- Wenn beide Zahlen Ganzzahlen sind und das Ergebnis außerhalb des Bereichs von `int64` liegt, wird es automatisch zu einer Gleitkommazahl mit doppelter Genauigkeit.
- Wenn mindestens eine der Zahlen eine Fließkommazahl ist, ist das Ergebnis eine Fließkommazahl mit doppelter Genauigkeit.
- Wenn das Ergebnis den Bereich von `double` überschreitet, gibt der Befehl einen `OVERFLOW` Fehler zurück.

Note

Vor der Redis-Engine-Version 6.2.6.R2, wenn eine JSON-Nummer bei der Eingabe empfangen wurde, wurde sie in eine der beiden internen Binärdarstellungen umgewandelt: eine 64-Bit-Ganzzahl mit Vorzeichen oder eine 64-Bit-IEEE-Gleitkommazahl mit doppelter Genauigkeit. Die Ursprüngliche Zeichenfolge und alle ihre Formatierungen werden nicht beibehalten. Wenn also eine Zahl als Teil einer JSON-Antwort ausgegeben wird, wird sie von der internen Binärdarstellung in eine druckbare Zeichenfolge konvertiert, die generische Formatierungsregeln verwendet. Diese Regeln könnten dazu führen, dass eine andere Zeichenfolge generiert wird als empfangen wurde.

- Wenn beide Zahlen außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt `int64`, wird das Ergebnis außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt.
- Wenn mindestens eine der Zahlen eine Gleitkommazahl ist, ergibt sich daraus eine doppelt genaue 64-Bit-Gleitkommazahl.
- Wenn das Ergebnis der doppelt genauen 64-Bit-Gleitkommazahl liegt, gibt der Befehl einen `OVERFLOW` Fehler zurück.

Eine detaillierte Liste der verfügbaren Befehle finden Sie unter [Unterstützte Befehle](#).

Strikte Syntaxbewertung

MemoryDB erlaubt keine JSON-Pfade mit ungültiger Syntax, selbst wenn eine Teilmenge des Pfads einen gültigen Pfad enthält. Dies soll für unsere Kunden ein korrektes Verhalten sicherstellen.

Unterstützte Befehle

Die folgenden Redis-JSON-Befehle werden unterstützt:

Themen

- [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

Hängen Sie einen oder mehrere Werte an die Array-Werte im Pfad an.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (erforderlich) — ein JSON-Pfad
- `json` (erforderlich) — JSON-Wert, der an das Array angehängt werden soll

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Ganzzahlen, die die neue Länge des Arrays an jedem Pfad darstellt.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.
- SYNTAXERR-Fehler, wenn eines der eingegebenen JSON-Argumente keine gültige JSON-Zeichenfolge ist.
- NONEXISTENT-Fehler, wenn der Pfad nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, die neue Länge des Arrays.
- Wenn mehrere Array-Werte ausgewählt wurden, gibt der Befehl die neue Länge des zuletzt aktualisierten Arrays zurück.
- WRONGTYPE-Fehler, wenn der Wert im Pfad kein Array ist.
- SYNTAXERR-Fehler, wenn eines der eingegebenen JSON-Argumente keine gültige JSON-Zeichenfolge ist.
- NONEXISTENT-Fehler, wenn der Pfad nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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\"]]"

```

Eingeschränkte Pfad-Syntax:

```

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

Suchen Sie in den Arrays am Pfad nach dem ersten Vorkommen eines skalaren JSON-Werts.

- Fehler außerhalb des Bereichs werden behandelt, indem der Index auf den Anfang und das Ende des Arrays gerundet wird.
- Wenn `start > end`, return -1 (nicht gefunden).

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (erforderlich) — ein JSON-Pfad
- `json-scalar` (erforderlich) — Skalarwert, nach dem gesucht werden soll; JSON-Skalar bezieht sich auf Werte, die keine Objekte oder Arrays sind. d. h. String, Zahl, Boolean und Null sind Skalarwerte.
- `start` (optional) — Startindex, einschließlich. Es gilt der Standardwert „0“, falls nicht vorhanden.
- `end` (optional) — Endindex, exklusiv. Es gilt der Standardwert „0“, wenn keine Angabe gemacht wird, was bedeutet, dass das letzte Element enthalten ist, was bedeutet, dass das letzte Element enthalten ist.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen. Jeder Wert ist der Index des übereinstimmenden Elements im Array am Pfad. Der Wert ist -1, falls nicht gefunden.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, der Index des übereinstimmenden Elements oder -1, falls nicht gefunden.
- WRONGTYPE-Fehler, wenn der Wert im Pfad kein Array ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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
```

Eingeschränkte Pfad-Syntax:

```
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

Fügen Sie einen oder mehrere Werte in die Array-Werte am Pfad vor dem Index ein.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (erforderlich) — ein JSON-Pfad
- `index` (erforderlich) — Array-Index, vor dem Werte eingefügt werden.
- `json` (erforderlich) — JSON-Wert, der an das Array angehängt werden soll

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Ganzzahlen, die die neue Länge des Arrays an jedem Pfad darstellt.
- Wenn ein Wert ein leeres Array ist, ist der entsprechende Rückgabewert Null.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.
- `OUTOFBOUNDARIES`-Fehler, wenn das Index-Argument außerhalb des gültigen Bereichs liegt.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, die neue Länge des Arrays.
- `WRONGTYPE`-Fehler, wenn der Wert im Pfad kein Array ist.
- `OUTOFBOUNDARIES`-Fehler, wenn das Index-Argument außerhalb des gültigen Bereichs liegt.

Beispiele

Erweiterte Pfad-Syntax:

```
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"]]"
```

Eingeschränkte Pfad-Syntax:

```
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

Ruft die Länge der Array-Werte am Pfad ab.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen, die die Arraylänge an jedem Pfad darstellen.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Array von Bulk-Strings. Jedes Element ist ein Schlüsselname im Objekt.
- Ganzzahl, Array-Länge.
- Wenn mehrere Objekte ausgewählt sind, gibt der Befehl die Länge des ersten Arrays zurück.
- **WRONGTYPE-Fehler**, wenn der Wert im Pfad kein Array ist.
- **WRONGTYPE-Fehler**, wenn der Pfad nicht vorhanden ist.

- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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)
```

Eingeschränkte Pfad-Syntax:

```
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

Entferne das Element am Index und gib es aus dem Array zurück. Das Öffnen eines leeren Arrays gibt null zurück.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.
- `index` (optional) — Position im Array, von der aus der Pop-up gestartet werden soll.
 - Ist standardmäßig -1, falls nicht angegeben, was auf das letzte Element verweist.
 - Negativer Wert bedeutet Position vom letzten Element.
 - Außerhalb der Grenzen liegende Indizes werden auf ihre jeweiligen Array-Grenzen gerundet.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Massenzeichenfolgen, die für jeden Pfad eingeblendete Werte darstellen.
- Wenn ein Wert ein leeres Array ist, ist der entsprechende Rückgabewert Null.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Bulk-Zeichenfolge, die den eingegebenen JSON-Wert darstellt
- Null, wenn das Array leer ist.
- `WRONGTYPE`-Fehler, wenn der Wert im Pfad kein Array ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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\"]]"
```

Eingeschränkte Pfad-Syntax:

```
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

Schneide die Arrays am Pfad so zu, dass er zu einem Subarray [Start, Ende] wird, beide eingeschlossen.

- Wenn das Array leer ist, tun Sie nichts, und geben Sie 0 zurück.
- Wenn start < 0 ist, behandeln Sie es als 0.
- Wenn end >= size (Größe des Arrays), behandeln Sie es als size-1.
- Wenn start >= size oder start > end, leeren Sie das Array und geben 0 zurück.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (erforderlich) — ein JSON-Pfad
- **start** (erforderlich) — Startindex, einschließlich.
- **end** (erforderlich) — Endindex, einschließlich.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Ganzzahlen, die die neue Länge des Arrays an jedem Pfad darstellt.
- Wenn ein Wert ein leeres Array ist, ist der entsprechende Rückgabewert Null.
- Wenn ein Wert kein Array ist, ist der entsprechende Rückgabewert Null.
- **OUTOFBOUNDARIES**-Fehler, wenn ein Indexargument außerhalb des gültigen Bereichs liegt.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, die neue Länge des Arrays.
- Null, wenn das Array leer ist.
- **WRONGTYPE**-Fehler, wenn der Wert im Pfad kein Array ist.
- **OUTOFBOUNDARIES**-Fehler, wenn ein Indexargument außerhalb des gültigen Bereichs liegt.

Beispiele

Erweiterte Pfad-Syntax:

```
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\\"]]"
```

Eingeschränkte Pfad-Syntax:

```
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

Löscht die Arrays oder ein Objekt am Pfad.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

- Ganzzahl, die Anzahl der entfernten Container.
- Das Löschen eines leeren Arrays oder Objekts führt zu 0 entfernten Containern.

Note

Vor der leeren Arrays oder Objekts führt zu einem entfernten Container.

- Das Löschen eines Nicht-Container-Werts gibt 0 zurück.
- Wenn sich am Pfad kein Array oder kein Objektwert befindet, gibt der Befehl 0 zurück.

Beispiele

```
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

Informationen melden. Unterstützte Unterbefehle sind:

- **MEMORY** <key>[path] — meldet die Speichernutzung in Byte eines JSON-Werts. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.
- <key>**DEPTH** [path] — Gibt die maximale Pfadtiefe des JSON-Dokuments an.

Note

Dieser Unterbefehl ist nur fest, wenn die Redis-Engine-Version 6.2.6.R2 oder höher.

- **FIELDS** <key>[path] — gibt die Anzahl der Felder im angegebenen Dokumentpfad an. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad. Jeder JSON-Wert, der kein Container ist, zählt als ein Feld. Objekte und Arrays zählen rekursiv ein Feld für jeden ihrer JSON-Werte. Jeder Containerwert, mit Ausnahme des Root-Containers, zählt als ein zusätzliches Feld.
- **HELP** — gibt die Hilfmeldungen des Befehls aus.

Syntax

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

Hängt vom Unterbefehl ab:

MEMORY

- Wenn der Pfad eine erweiterte Syntax ist:

- gibt ein Array von Ganzzahlen zurück, das die Speichergröße (in Byte) des JSON-Werts in jedem Pfad darstellt.
- gibt ein leeres Array zurück, wenn der Redis-Schlüssel nicht vorhanden ist.
- Wenn der Pfad eine eingeschränkte Syntax hat:
 - gibt eine Ganzzahl zurück, Speichergröße den JSON-Wert in Byte.
 - gibt Null zurück, wenn der Redis-Schlüssel nicht vorhanden ist.

DEPTH

- Gibt eine Ganzzahl zurück, die die maximale Pfadtiefe des JSON-Dokuments darstellt.
- Gibt Null zurück, wenn der Redis-Schlüssel nicht existiert.

FIELDS

- Wenn der Pfad eine erweiterte Syntax ist:
 - gibt ein Array von Ganzzahlen zurück, das die Anzahl der Felder mit JSON-Werten in jedem Pfad darstellt.
 - gibt ein leeres Array zurück, wenn der Redis-Schlüssel nicht vorhanden ist.
- Wenn der Pfad eine eingeschränkte Syntax hat:
 - gibt eine Ganzzahl zurück, die Anzahl der Felder des JSON-Werts.
 - gibt Null zurück, wenn der Redis-Schlüssel nicht vorhanden ist.

HELP — gibt eine Reihe von Hilfenmeldungen zurück.

Beispiele

Erweiterte Pfad-Syntax:

```
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

```

Eingeschränkte Pfad-Syntax:

```

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

Löschen Sie die JSON-Werte am Pfad in einem Dokumentschlüssel. Wenn der Pfad das Stammverzeichnis ist, entspricht dies dem Löschen des Schlüssels aus Redis.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

- Anzahl der gelöschten Elemente.
- 0, wenn der Redis-Schlüssel nicht existiert.
- 0, wenn der JSON-Pfad ungültig ist oder nicht existiert.

Beispiele

Erweiterte Pfad-Syntax:

```
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\":{\"a\":1,\"b\":2,\"c\":3},\"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\":{\"a\":1,\"b\":2,\"c\":3},\"e\":[]}"
```

Eingeschränkte Pfad-Syntax:

```

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

Ein Alias von [JSON.DEL](#)

JSON.GET

Gibt das serialisierte JSON an einem oder mehreren Pfaden zurück.

Syntax

```

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

```

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **INDENT/NEWLINE/SPACE** (optional) — steuert das Format der zurückgegebenen JSON-Zeichenfolge, d. h. „pretty print“. Der Standardwert jedes einzelnen ist eine leere Zeichenfolge. Sie können in beliebiger Kombination überfahren werden. Sie können in beliebiger Reihenfolge angegeben werden.
- **NOESCAPE** — optional, darf aus Gründen der Kompatibilität mit älteren Versionen vorhanden sein und hat keine weitere Wirkung.
- **path** (optional) — Null oder mehr JSON-Pfade, standardmäßig das Stammverzeichnis, falls keiner angegeben wird. Die Pfadargumente müssen am Ende platziert werden.


```

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

Rufen Sie serialisierte JSONs am Pfad aus mehreren Dokumentschlüsseln ab. Gibt Null für einen nicht existierenden Schlüssel oder JSON-Pfad zurück.

Syntax

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

- **key** (erforderlich) – Ein oder mehrere Redis-Schlüssel des Dokumenttyps.
- **path** (erforderlich) — ein JSON-Pfad

Ergebnis

- Reihe von Massenzeichenfolgen. Die Größe des Arrays entspricht der Anzahl der Schlüssel im Befehl. Jedes Element des Arrays wird entweder mit (a) dem serialisierten JSON gefüllt, wie es sich im Pfad befindet, oder (b) Null, wenn der Schlüssel nicht existiert oder der Pfad nicht im Dokument existiert oder der Pfad ungültig ist (Syntaxfehler).
- Wenn einer der angegebenen Schlüssel existiert und kein JSON-Schlüssel ist, gibt der Befehl den Fehler `WRONGTYPE` zurück.

Beispiele

Erweiterte Pfad-Syntax:

```
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\ "]"
```

Eingeschränkte Pfad-Syntax:

```
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

Erhöhen Sie die Zahlenwerte am Pfad um eine bestimmte Zahl.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (erforderlich) — ein JSON-Pfad
- `Zahl` (erforderlich) — eine Zahl

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Massenzeichenfolgen, die den resultierenden Wert für jeden Pfad darstellen.
- Wenn ein Wert keine Zahl ist, ist der entsprechende Rückgabewert Null.
- `WRONGTYPE`-Fehler, wenn die Zahl nicht analysiert werden kann.
- `OVERFLOW`-Fehler, wenn das Ergebnis außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt.
- `NONEXISTENT` wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Bulk-Zeichenfolge, die den resultierenden Wert darstellt.
- Wenn mehrere Werte ausgewählt wurden, gibt der Befehl das Ergebnis des zuletzt aktualisierten Arrays zurück.
- `WRONGTYPE`-Fehler, wenn der Wert im Pfad keine Zahl ist.
- `WRONGTYPE`-Fehler, wenn die Zahl nicht analysiert werden kann.
- `OVERFLOW`-Fehler, wenn das Ergebnis außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt.
- `NONEXISTENT` wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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}}"
```

Eingeschränkte Pfad-Syntax:

```

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

Multiplizieren Sie die Zahlenwerte am Pfad mit einer bestimmten Zahl.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (erforderlich) — ein JSON-Pfad
- **Zahl** (erforderlich) — eine Zahl

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Massenzeichenfolgen, die den resultierenden Wert für jeden Pfad darstellen.
- Wenn ein Wert keine Zahl ist, ist der entsprechende Rückgabewert Null.
- **WRONGTYPE**-Fehler, wenn die Zahl nicht analysiert werden kann.
- **OVERFLOW**-Fehler, wenn das Ergebnis außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt.
- **NONEXISTENT** wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Bulk-Zeichenfolge, die den resultierenden Wert darstellt.

- Wenn mehrere Werte ausgewählt wurden, gibt der Befehl das Ergebnis des zuletzt aktualisierten Arrays zurück.
- WRONGTYPE-Fehler, wenn der Wert im Pfad keine Zahl ist.
- WRONGTYPE-Fehler, wenn die Zahl nicht analysiert werden kann.
- OVERFLOW-Fehler, wenn das Ergebnis außerhalb des Bereichs der doppelt genauen 64-Bit-Gleitkommazahl liegt.
- NONEXISTENT wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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]"

```

Eingeschränkte Pfad-Syntax:

```

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

Ruft die Anzahl der Schlüssel in den Objektwerten im Pfad ab.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps

- `path` (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen, die die Objektlänge an jedem Pfad darstellen.
- Wenn ein Wert kein Objekt ist, ist der entsprechende Rückgabewert null.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, Anzahl der Schlüssel im Objekt.
- Wenn mehrere Objekte ausgewählt sind, gibt der Befehl die Länge des ersten Objekts zurück.
- `WRONGTYPE`-Fehler, wenn der Wert im Pfad kein Objekt ist.
- `WRONGTYPE`-Fehler, wenn der Pfad nicht vorhanden ist.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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)

```

Eingeschränkte Pfad-Syntax:

```

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

Ruft Schlüsselnamen in den Objektwerten am Pfad ab.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Arrays von Bulk-Strings. Jedes Element ist ein Array von Schlüsseln in einem übereinstimmenden Objekt.
- Wenn ein Wert kein Objekt ist, ist der entsprechende Rückgabewert leer.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Array von Bulk-Strings. Jedes Element ist ein Schlüsselname im Objekt.
- Wenn mehrere Objekte ausgewählt sind, gibt der Befehl die Schlüssel des ersten Objekts zurück.
- `WRONGTYPE`-Fehler, wenn der Wert im Pfad kein Objekt ist.
- `WRONGTYPE`-Fehler, wenn der Pfad nicht vorhanden ist.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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"
```

Eingeschränkte Pfad-Syntax:

```
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

Gibt den JSON-Wert am angegebenen Pfad im Redis Serialization Protocol (RESP) zurück. Wenn der Wert Container ist, ist die Antwort ein RESP-Array oder ein verschachteltes Array.

- JSON null wird dem RESP Null Bulk String zugeordnet.
- Boolesche JSON-Werte werden den jeweiligen RESP-Zeichenketten zugeordnet.
- Ganzzahlen werden RESP-Ganzzahlen zugeordnet.
- Doppelt genaue 64-Bit-Gleitkommazahlen werden RESP-Bulk-Strings zugeordnet.
- JSON-Zeichenketten werden RESP-Bulk-Strings zugeordnet.
- JSON-Arrays werden als RESP-Arrays dargestellt, wobei das erste Element die einfache Zeichenfolge [ist, gefolgt von den Elementen des Arrays.
- JSON-Objekte werden als RESP-Arrays dargestellt, wobei das erste Element die einfache Zeichenfolge {ist, gefolgt von Schlüssel-Wert-Paaren, von denen jedes eine RESP-Massenzeichenfolge ist.

Syntax

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

- `key` (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- `path` (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Arrays. Jedes Array-Element repräsentiert die RESP-Form des Werts in einem Pfad.
- Leeres Array wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Array, das die RESP-Form des Werts im Pfad darstellt.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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"
```

Eingeschränkte Pfad-Syntax:

```
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

Stellen Sie JSON-Werte für den Pfad ein.

Wenn der Pfad ein Objektmitglied aufruft:

- Wenn das übergeordnete Element nicht existiert, gibt der Befehl den Fehler NONEXISTENT zurück.
- Wenn das übergeordnete Element existiert, aber kein Objekt ist, gibt der Befehl ERROR zurück.
- Wenn das übergeordnete Element existiert und ein Objekt ist:
 - Wenn das Mitglied nicht existiert, wird ein neues Mitglied an das übergeordnete Objekt angehängt, wenn das übergeordnete Objekt das letzte untergeordnete Objekt im Pfad ist. Andernfalls gibt der Befehl den Fehler NONEXISTENT zurück.
 - Wenn das Mitglied existiert, wird sein Wert durch den JSON-Wert ersetzt.

Wenn der Pfad einen Array-Index aufruft:

- Wenn das übergeordnete Element nicht existiert, gibt der Befehl den Fehler NONEXISTENT zurück.
- Wenn das übergeordnete Element existiert, aber kein Array ist, gibt der Befehl ERROR zurück.
- Wenn das übergeordnete Element vorhanden ist, der Index jedoch außerhalb der Grenzen liegt, gibt der Befehl den Fehler OUTFOUBOUNDARIES zurück.
- Wenn das übergeordnete Element existiert und der Index gültig ist, wird das Element durch den neuen JSON-Wert ersetzt.

Wenn der Pfad ein Objekt oder Array aufruft, wird der Wert (Objekt oder Array) durch den neuen JSON-Wert ersetzt.

Syntax

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

[NX | XX] Wobei Sie 0 oder 1 der [NX | XX] -Kennungen haben können

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **Pfad** (erforderlich) — JSON-Pfad. Für einen neuen Redis-Schlüssel muss der JSON-Pfad dem Stamm „.“ entsprechen.
- **NX** (optional) — Wenn der Redis-Schlüssel nicht existiert, legen Sie den Wert nur fest, wenn der Redis-Schlüssel existiert. Wenn der Pfad nicht der Stamm ist, legen Sie den Wert nur fest, wenn der Pfad nicht existiert, d. h. fügen Sie einen Wert in das Dokument ein.
- **XX** (optional) — Wenn der Redis-Schlüssel existiert, legen Sie den Wert nur fest, wenn der Redis-Schlüssel existiert. Wenn der Pfad nicht die Wurzel ist, legen Sie den Wert nur fest, wenn der Pfad existiert, d. h. aktualisieren Sie den vorhandenen Wert.

Ergebnis

- Einfache Zeichenfolge 'OK' bei Erfolg.
- Null, wenn die NX- oder XX-Bedingungen nicht erfüllt sind.

Beispiele

Erweiterte Pfad-Syntax:

```
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]}"
```

Eingeschränkte Pfad-Syntax:

```
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

Hängt eine Zeichenfolge an die JSON-Zeichenketten im Pfad an.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.
- **json_string** (erforderlich) — JSON-Darstellung einer Zeichenfolge. Beachten Sie, dass eine JSON-Zeichenfolge in Anführungszeichen gesetzt werden muss, z. B. `"foo"`.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen, die die neue Länge der Zeichenfolge an jedem Pfad darstellen.
- Wenn ein Wert im Pfad keine Zeichenfolge ist, ist der entsprechende Rückgabewert Null.
- **SYNTAXERR** Fehler, wenn das angegebene JSON-Argument keine gültige JSON-Zeichenfolge ist.
- **NONEXISTENT** Fehler, wenn der Pfad nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, die neue Länge der Zeichenfolge.
- Wenn mehrere Zeichenfolgenwerte ausgewählt wurden, gibt der Befehl die neue Länge der zuletzt aktualisierten Zeichenfolge zurück.
- WRONGTYPE-Fehler, wenn der Wert im Pfad keine Zeichenfolge ist.
- WRONGTYPE-Fehler, wenn das angegebene JSON-Argument keine gültige JSON-Zeichenfolge ist.
- NONEXISTENT-Fehler, wenn der Pfad nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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)
```

Eingeschränkte Pfad-Syntax:

```
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.STRLLEN

Ruft die Längen der JSON-Zeichenfolgenwerte im Pfad ab.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen, die die Länge des Zeichenfolgenwerts in jedem Pfad darstellen.
- Wenn ein Wert keine Zeichenfolge ist, ist der entsprechende Rückgabewert Null.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Ganzzahl, die Länge der Zeichenfolge.
- Wenn mehrere Zeichenfolgenwerte ausgewählt wurden, gibt der Befehl die erste Zeichenfolgenlänge zurück.
- **WRONGTYPE-Fehler**, wenn der Wert im Pfad keine Zeichenfolge ist.
- **NONEXISTENT-Fehler**, wenn der Pfad nicht vorhanden ist.
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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)
```

Eingeschränkte Pfad-Syntax:

```
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

Schaltet die booleschen Werte am Pfad zwischen wahr und falsch um.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von ganzen Zahlen (0 — falsch, 1 — wahr), die den resultierenden booleschen Wert in jedem Pfad darstellen.
- Wenn ein Wert nicht boolesch ist, ist der entsprechende Rückgabewert Null.
- **NONEXISTENT** wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Zeichenfolge („true“ / "false“), die den resultierenden booleschen Wert darstellt.
- **NONEXISTENT** wenn der Dokumentschlüssel nicht vorhanden ist.
- **WRONGTYPE**Fehler, wenn der Wert im Pfad kein boolescher Wert ist.

Beispiele

Erweiterte Pfad-Syntax:

```
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)
```

Eingeschränkte Pfad-Syntax:

```
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

Berichtstyp der Werte im angegebenen Pfad.

Syntax

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

- **key** (erforderlich) — Redis-Schlüssel des JSON-Dokumenttyps
- **path** (optional) — ein JSON-Pfad. Falls nicht angegeben, gilt der Root-Standardwert für den Pfad.

Ergebnis

Wenn der Pfad eine erweiterte Syntax ist:

- Array von Zeichenketten, die den Typ des Werts in jedem Pfad darstellen. Typ {„null“, „boolean“, „string“, „number“, „integer“, „object“ und „array“}.
- Wenn ein Pfad nicht existiert, ist der entsprechende Ausgabewert null.
- Leeres Array wenn der Dokumentschlüssel nicht vorhanden ist.

Wenn der Pfad eine eingeschränkte Syntax hat:

- Zeichenfolge, Typ des Werts
- Null, wenn der Dokumentschlüssel nicht vorhanden ist.
- Null, wenn der JSON-Pfad ungültig ist oder nicht existiert.

Beispiele

Erweiterte Pfad-Syntax:

```
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
```

Eingeschränkte Pfad-Syntax:

```
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
```

Markierung Ihrer MemoryDB-Ressourcen

Um Sie bei der Verwaltung Ihrer Cluster und anderer MemoryDB-Ressourcen zu unterstützen, können Sie jeder Ressource eigene Metadaten in Form von Tags zuweisen. Mit Tags (Markierungen) können Sie Ihre AWS-Ressourcen auf unterschiedliche Weise kategorisieren (z. B. nach Zweck, Eigentümer oder Umgebung). Dies ist nützlich, wenn Sie viele Ressourcen desselben Typs haben — In diesem Fall können Sie schnell bestimmte Ressourcen basierend auf den zugewiesenen Tags (Markierungen) bestimmen. In diesem Thema werden Tags (Markierungen) und deren Erstellung beschrieben.

Warning

Als bewährte Vorgehensweise empfehlen wir Ihnen, keine sensiblen Daten in Ihre Tags (Markierungen) aufzunehmen.

Grundlagen zu Tags (Markierungen)

Ein Tag (Markierung) ist eine Markierung, die Sie einer AWS-Ressource zuordnen. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen. Mit Tags können Sie Ihre AWS-Ressourcen auf unterschiedliche Weise kategorisieren, beispielsweise nach Zweck oder Besitzer. Sie können beispielsweise eine Gruppe von Tags für die MemoryDB-Cluster Ihres Kontos definieren, mit deren Hilfe Sie den Besitzer und die Benutzergruppe der einzelnen Cluster verfolgen können.

Wir empfehlen die Verwendung von Tag (Markierung)-Schlüsseln, die die Anforderungen der jeweiligen Ressourcentypen erfüllen. Eine Anzahl einheitlicher Tag (Markierung)-Schlüssel

vereinfacht das Verwalten der Ressourcen. Sie können die Ressourcen auf Grundlage der hinzugefügten Tags (Markierungen) filtern und danach suchen. Weitere Informationen zum Implementieren einer effektiven Ressourcen-Tagging-Strategie finden Sie im [AWS Whitepaper Markierungsbewährte Methoden für](#) aus.

Tags haben für MemoryDB keine semantische Bedeutung und werden ausschließlich als Zeichenfolgen interpretiert. Außerdem werden Tags (Markierungen) nicht automatisch Ihren Ressourcen zugewiesen. Sie können Tag (Markierung)-Schlüssel und -Werte bearbeiten und Tags (Markierungen) jederzeit von einer Ressource entfernen. Sie können den Wert eines Tags auf `null` setzen. Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben. Wenn Sie eine Ressource löschen, werden alle Tags (Markierungen) der Ressource ebenfalls gelöscht.

Sie können mit Tags arbeiten, indem Sie die AWS Management Console, die AWS CLI und die MemoryDB-API.

Wenn Sie IAM verwenden, können Sie steuern, welche Benutzer in Ihrem AWS-Konto berechtigt sind, Tags zu erstellen, zu bearbeiten oder zu löschen. Weitere Informationen finden Sie unter [Berechtigungen auf Ressourcenebene](#).

Ressourcen, die markiert werden können

Sie können die meisten MemoryDB-Ressourcen markieren, die bereits in Ihrem Konto bestehen. In der Tabelle unten werden die Ressourcen aufgeführt, die das Markieren unterstützen. Wenn Sie die AWS Management Console verwenden, können Sie mithilfe des [Tag-Editors](#) Tags auf Ressourcen anwenden. Auf einigen Ressourcenbildschirmen können Sie beim Erstellen der Ressource Tags für diese Ressource angeben, z. B. ein Tag mit dem Schlüssel „Name“ und einem von Ihnen angegebenen Wert. In den meisten Fällen wendet die Konsole Tags (Markierungen) direkt nach dem Erstellen der Ressource an und nicht während des Erstellens. Die Konsole kann Ressourcen entsprechend der Name-Tag, aber dieses Tag hat keine semantische Bedeutung für den MemoryDB-Dienst.

Zudem können Sie mit einigen Aktionen zur Ressourcenerstellung Tags beim Erstellen einer Ressource angeben. Wenn Tags (Markierungen) nicht während der Ressourcenerstellung angewendet werden können, wird die Ressourcenerstellung rückgängig gemacht. Auf diese Weise werden Ressourcen entweder mit Tags (Markierungen) oder überhaupt nicht erstellt und keine Ressourcen verbleiben ohne Tags (Markierungen). Indem Sie Ressourcen zum Erstellungszeitpunkt markieren, müssen Sie anschließend keine benutzerdefinierten Skripts ausführen.

Wenn Sie die Amazon MemoryDB-API verwenden, wird die AWS CLI oder eine AWS SDK können Sie die `Tags` Parameter für die relevante MemoryDB-API-Aktion zum Anwenden von Tags. Diese sind:

- `CreateCluster`
- `CopySnapshot`
- `CreateParameterGroup`
- `CreateSubnetGroup`
- `CreateSnapshot`
- `CreateACL`
- `CreateUser`

In der folgenden Tabelle werden die markierbaren MemoryDB-Ressourcen und die bei der Erstellung über die MemoryDB-API markierbaren Ressourcen beschrieben, AWS CLI oder eine AWS-SDK.

Markierungsunterstützung für MemoryDB-Ressourcen

Unterstützt Tags (Markierungen)	Unterstützt Markierung bei der Erstellung
Ja	Ja

Sie können Tag-basierte Berechtigungen auf Ressourcenebene in Ihren IAM-Richtlinien auf die MemoryDB-API-Aktionen anwenden, die das Tagging bei der Erstellung unterstützen, um eine granulare Kontrolle über die Benutzer und Gruppen zu implementieren, die Ressourcen bei der Erstellung mit Tags versehen können. Ihre Ressourcen sind vor der Erstellung ordnungsgemäß

gesichert – Tags, die sofort auf Ihre Ressourcen angewendet werden. Daher sind alle tagbasierten Berechtigungen auf Ressourcenebene, welche die Verwendung von Ressourcen steuern, sofort wirksam. Ihre Ressourcen können nachverfolgt und genauer erfasst werden. Sie können das Markieren neuer Ressourcen gewährleisten und steuern, welche Tag (Markierung)-Schlüssel und Werte für Ihre Ressourcen festgelegt sind.

Weitere Informationen finden Sie unter [Beispiele für das Taggen von Ressourcen](#).

Weitere Informationen zur Markierung von Ressourcen für die Fakturierung finden Sie unter [Überwachung von Kosten mit Kostenzuordnungs-Tags](#).

Markieren von Clustern und Schnappschüssen

Die folgenden Regeln gelten für das Markieren im Rahmen von Anforderungsvorgängen:

- **CreateCluster :**
 - Wenn `--cluster-name` geliefert wird:

Wenn die Anforderung Tags enthält, wird der Cluster getaggt.
 - Wenn `--snapshot-name` geliefert wird:

Wenn die Anforderung Tags enthält, wird der Cluster nur mit diesen Tags getaggt. Wenn die Anforderung keine Tags enthält, werden die Snapshot-Tags dem Cluster hinzugefügt.
- **CreateSnapshot :**
 - Wenn `--cluster-name` geliefert wird:

Wenn die Anforderung Tags enthält, werden nur die Anfrage-Tags zum Snapshot hinzugefügt. Wenn die Anforderung keine Tags enthält, werden die Cluster-Tags dem Snapshot hinzugefügt.
 - Für automatische Snapshots:

Tags werden von den Cluster-Tags weitergegeben.
- **CopySnapshot :**

Wenn die Anforderung Tags enthält, werden nur die Request-Tags zum Snapshot hinzugefügt. Wenn die Anforderung keine Tags enthält, werden die Quell-Snapshot-Tags zum kopierten Snapshot hinzugefügt.
- **TagResource und UntagResource:**

Tags werden der Ressource hinzugefügt/entfernt.

Tag (Markierung)-Einschränkungen

Die folgenden grundlegenden Einschränkungen gelten für Tags (Markierungen):

- Maximale Anzahl von Tags (Markierungen) pro Ressource: 50
- Jeder Tag (Markierung) muss für jede Ressource eindeutig sein. Jeder Tag (Markierung) kann nur einen Wert haben.
- Maximale Schlüssellänge – 128 Unicode-Zeichen in UTF-8.
- Maximale Wertlänge – 256 Unicode-Zeichen in UTF-8.
- Obwohl MemoryDB jedes Zeichen in seinen Tags zulässt, können andere Dienste restriktiv sein. Folgende Zeichen sind dienstübergreifend zulässig: Buchstaben, Zahlen und Leerzeichen, die in UTF-8 dargestellt werden können, sowie die folgenden Zeichen: + - = . _ : / @
- Bei Tag-Schlüsseln und -Werten wird zwischen Groß- und Kleinschreibung unterschieden.
- Das Präfix `aws :` ist zur Verwendung in AWS reserviert. Wenn der Tag (Markierung) über einen Tag (Markierung)-Schlüssel mit diesem Präfix verfügt, können Sie den Schlüssel oder Wert des Tags (Markierung) nicht bearbeiten oder löschen. Tags (Markierungen) mit dem Präfix `aws :` werden nicht als Ihre Tags (Markierungen) pro Ressourcenlimit angerechnet.

Sie können Ressourcen nicht allein auf Grundlage ihrer Tags (Markierungen) beenden, anhalten oder löschen. Sie müssen den Ressourcenbezeichner angeben. Um Snapshots zu löschen, die Sie mit dem Tag (Markierung)-Schlüssel `DeleteMe` markiert haben, müssen Sie die `DeleteSnapshot`-Aktion mit den Ressourcenbezeichnern der Snapshots verwenden, z. B. `snap-1234567890abcdef0`.

Weitere Informationen zu MemoryDB-Ressourcen, die Sie taggen können, finden Sie unter [Ressourcen, die markiert werden können](#) aus.

Beispiele für das Taggen von Ressourcen

- Hinzufügen von Tags zu einem 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"
```

- Erstellen eines Clusters mithilfe von Tags.

```
aws memorydb create-cluster \  
--tags Key="project",Value="XYZ" Key="memorydb",Value="Service"
```

```
--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"
```

- Erstellen eines Snapshots mit Tags.

Wenn Sie in diesem Fall Tags auf Anfrage hinzufügen, erhält der Snapshot nur die Anfrage-Tags, selbst wenn der Cluster Tags enthält.

```
aws memorydb create-snapshot \  
--cluster-name testing-tags \  
--snapshot-name bkp-testing-tags-mycluster \  
--tags Key="work",Value="foo"
```

Überwachung von Kosten mit Kostenzuordnungs-Tags

Wenn Sie Ihren Ressourcen in MemoryDB für Redis Kostenzuordnungs-Tags hinzufügen, können Sie Kosten verfolgen, indem Sie Ausgaben auf Ihren Rechnungen nach Ressourcen-Tag-Werten gruppieren.

Ein MemoryDB-Kostenzuordnungs-Tag ist ein Name-Wert-Paar, das Sie definieren und mit einer MemoryDB-Ressource verknüpfen. Bei Schlüssel und Werten werden Groß-/Kleinschreibung berücksichtigt. Sie können einen Tag-Schlüssel verwenden, um eine Kategorie zu definieren, und der Tag-Wert kann ein Element in dieser Kategorie sein. So könnten Sie beispielsweise den Tag-Schlüssel `CostCenter` und den Tag-Wert `10010` definieren, um anzugeben, dass die Ressource der Kostenstelle 10010 zugewiesen ist. Sie können mit Tags auch Ressourcen kennzeichnen, die zu Test- oder Produktionszwecken verwendet werden, indem Sie einen Schlüssel wie z. B. `Environment` und Werte wie z. B. `test` oder `production` verwenden. Wir empfehlen, einheitliche Tag-Schlüssel zu verwenden, um die mit Ihren Ressourcen verknüpften Kosten einfacher verfolgen zu können.

Verwenden Sie Kostenzuordnungs-Tags, um Ihre AWS-Rechnung so zu organisieren, dass sie Ihre eigene Kostenstruktur widerspiegelt. Dazu müssen Sie sich registrieren, um Ihre AWS-Kontorechnung mit Tag (Markierung)-Schlüsselwerten zu erhalten. Um dann die Kosten kombinierter Ressourcen anzuzeigen, organisieren Sie Ihre Fakturierungsinformationen nach Ressourcen

mit gleichen Tag-Schlüsselwerten. Beispielsweise können Sie mehrere Ressourcen mit einem bestimmten Anwendungsnamen markieren und dann Ihre Fakturierungsinformationen so organisieren, dass Sie die Gesamtkosten dieser Anwendung über mehrere Services hinweg sehen können.

Sie können auch Tags miteinander kombinieren, um Kosten detaillierter zu verfolgen. Um beispielsweise Ihre Kosten für Services nach Region zu verfolgen, könnten Sie die Tag-Schlüssel `Service` und `Region` verwenden. Für eine Ressource lauten die Werte möglicherweise `MemoryDB` und `Asia Pacific (Singapore)` und für eine andere Ressource lauten sie `MemoryDB` und `Europe (Frankfurt)`. Sie können dann Ihre gesamten MemoryDB-Kosten nach Regionen aufgeschlüsselt sehen. Weitere Informationen finden Sie unter [Verwendung von Kostenzuordnungs-Tags](#) im AWS Billing-Benutzerhandbuch.

Sie können MemoryDB-Kostenzuordnungs-Tags zu MemoryDB-Clustern hinzufügen. Wenn Sie ein Tag hinzufügen, auflisten, ändern, kopieren oder entfernen, wird die Operation nur auf die angegebenen Cluster angewendet.

Eigenschaften von MemoryDB-Kostenzuordnungs-Tags

- Kostenzuordnungs-Tags werden auf MemoryDB-Ressourcen angewendet, die in CLI- und API-Operationen als ARN angegeben werden. Der Ressourcentyp ist ein „Cluster“.

ARN-Format: `arn:aws:memorydb:<region>:<customer-id>:<resource-type>/<resource-name>`

Beispiel-ARN: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`

- Der Tag-Schlüssel ist der erforderliche Name des Tags. Der Zeichenfolgenwert kann aus 1 bis 128 Unicode-Zeichen bestehen. Ihm darf kein `aws:` als Präfix vorangestellt werden. Die Zeichenfolge darf nur Unicode-Zeichen, Ziffern, Leerzeichen, Unterstriche (`_`), Punkte (`.`), Doppelpunkte (`:`), Backslashes (`\`), Gleichheitszeichen (`=`), Pluszeichen (`+`), Trennstriche (`-`) oder At-Zeichen (`@`) enthalten.
- Der Tag-Wert ist der optionale Wert des Tags. Der Zeichenfolgenwert kann aus 1 bis 256 Unicode-Zeichen bestehen. Ihm darf kein `aws:` als Präfix vorangestellt werden. Die Zeichenfolge darf nur Unicode-Zeichen, Ziffern, Leerzeichen, Unterstriche (`_`), Punkte (`.`), Doppelpunkte (`:`), Backslashes (`\`), Gleichheitszeichen (`=`), Pluszeichen (`+`), Trennstriche (`-`) oder At-Zeichen (`@`) enthalten.
- Eine MemoryDB-Ressource kann maximal 50 Tags haben.
- Die Werte innerhalb eines Tag-Satzes müssen nicht eindeutig sein. Beispiel: In einem Tag-Satz könnten die Schlüssel `Service` und `Application` beide den Wert `MemoryDB` besitzen.

AWS ordnet Ihren Tags keine semantische Bedeutung zu. Tags werden streng als Zeichenfolgen interpretiert. AWS legt nicht automatisch Tags für eine MemoryDB-Ressource fest.

Verwalten Ihrer Kostenzuordnungs-Tags mithilfe der AWS CLI

Zum Hinzufügen, Ändern oder Entfernen von Kostenzuordnung-Tags können Sie das AWS CLI verwenden.

Beispiel-ARN: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`

Themen

- [Auflisten von Tags mithilfe der AWS CLI](#)
- [Hinzufügen von Tags mithilfe der AWS CLI](#)
- [Ändern von Tags mithilfe der AWS CLI](#)
- [Entfernen von Tags mithilfe der AWS CLI](#)

Auflisten von Tags mithilfe der AWS CLI

Sie können das AWS CLI So listen Sie Tags für eine vorhandene MemoryDB-Ressource auf, indem Sie die [list-tags](#) verwenden.

Der folgende Code verwendet die AWS CLI um die Tags im MemoryDB-Cluster aufzulisten `my-cluster` in der Region `us-east-1`.

Für Linux, macOS oder Unix:

```
aws memorydb list-tags \  
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
```

Für Windows:

```
aws memorydb list-tags ^  
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster
```

Die Ausgabe dieser Operation sieht in etwa folgendermaßen aus und besteht aus einer Liste aller Tags für die Ressource.

```
{
```

```
"TagList": [  
  {  
    "Value": "10110",  
    "Key": "CostCenter"  
  },  
  {  
    "Value": "EC2",  
    "Key": "Service"  
  }  
]
```

Wenn die Ressource keine Tags enthält, ist die Ausgabe leerTagListaus.

```
{  
  "TagList": []  
}
```

Weitere Informationen finden Sie im `.AWS CLI` für MemoryDB [list-tags](#)aus.

Hinzufügen von Tags mithilfe der AWS CLI

Sie können das AWS CLI So fügen Sie Tags zu einer vorhandenen MemoryDB-Ressource hinzu, indem Sie die [tag-resource](#) CLI-Betrieb. Wenn das Tag in der Ressource nicht vorhanden ist, werden Schlüssel und Wert zur Ressource hinzugefügt. Wenn der Schlüssel in der Ressource bereits vorhanden ist, wird der diesem Schlüssel zugeordnete Wert auf den neuen Wert aktualisiert.

Der folgende Code verwendet die AWS CLI um die Schlüssel hinzuzufügen Service und Region mit den Werten `memorydb` und `us-east-1` beziehungsweise zum Cluster `my-cluster` in der Region `us-east-1`.

Für Linux, macOS oder 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
```

Für 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
```

Die Ausgabe dieser Operation sieht in etwa folgendermaßen aus und besteht aus einer Liste aller Tags für die Ressource nach der Operation.

```
{
  "TagList": [
    {
      "Value": "memorydb",
      "Key": "Service"
    },
    {
      "Value": "us-east-1",
      "Key": "Region"
    }
  ]
}
```

Weitere Informationen finden Sie im `.AWS CLI` für MemoryDB [tag-resource](#) aus.

Sie können auch die `AWS CLI` So fügen Sie Tags einem Cluster hinzu, wenn Sie mithilfe der Operation einen neuen Cluster erstellen [create-cluster](#) aus.

Ändern von Tags mithilfe der AWS CLI

Sie können das `AWS CLI` um die Tags in einem MemoryDB-Cluster zu ändern.

Ändern Sie Tags wie folgt:

- Verwenden von [tag-resource](#) So fügen Sie entweder ein neues Tag und Wert hinzu oder ändern Sie den mit einem vorhandenen Tag gehörenden Wert.
- Verwenden von [untag-resource](#) um angegebene Tags aus der Ressource zu entfernen.

zum Entfernen von Tags aus dem Cluster.

Entfernen von Tags mithilfe der AWS CLI

Sie können das `AWS CLI` So entfernen Sie Tags aus einem vorhandenen aus einem MemoryDB-Cluster mithilfe der [untag-resource](#) verwenden.

Der folgende Code verwendet die AWS CLI die Tags mit den Schlüsseln zu entfernen `Service` und `Region` aus dem Cluster `my-cluster` in der Region `us-east-1`.

Für Linux, macOS oder Unix:

```
aws memorydb untag-resource \  
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster \  
  --tag-keys Region Service
```

Für Windows:

```
aws memorydb untag-resource ^  
  --resource-arn arn:aws:memorydb:us-east-1:0123456789:cluster/my-cluster ^  
  --tag-keys Region Service
```

Die Ausgabe dieser Operation sieht in etwa folgendermaßen aus und besteht aus einer Liste aller Tags für die Ressource nach der Operation.

```
{  
  "TagList": []  
}
```

Weitere Informationen finden Sie im `.AWS CLI` für MemoryDB [untag-resource](#) aus.

Verwalten Ihrer Kostenzuordnungs-Tags mithilfe der MemoryDB-API

Sie können die MemoryDB-API verwenden, um Kostenzuordnungs-Tags hinzuzufügen, zu ändern oder zu entfernen.

Kostenzuordnungs-Tags werden für `-Cluster` auf MemoryDB angewendet. Der Cluster, der gekennzeichnet werden soll, wird mit einem ARN (Amazon Resource Name) angegeben.

Beispiel-ARN: `arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`

Themen

- [Auflisten von Tags mithilfe der MemoryDB-API](#)
- [Hinzufügen von Tags mithilfe der MemoryDB-API](#)
- [Ändern von Tags mithilfe der MemoryDB-API](#)

- [Entfernen von Tags mithilfe der MemoryDB-API](#)

Auflisten von Tags mithilfe der MemoryDB-API

Sie können mithilfe der MemoryDB-API Tags für eine vorhandene Ressource auflisten, indem Sie die [ListTags](#) verwenden.

Der folgende Code listet mithilfe der MemoryDB-API die Tags auf der -Ressource aufmy-clusterin der Region 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>
```

Hinzufügen von Tags mithilfe der MemoryDB-API

Sie können mithilfe der MemoryDB-API Tags einem vorhandenen MemoryDB-Cluster hinzufügen, indem Sie die [TagResource](#) verwenden. Wenn das Tag in der Ressource nicht vorhanden ist, werden Schlüssel und Wert zur Ressource hinzugefügt. Wenn der Schlüssel in der Ressource bereits vorhanden ist, wird der diesem Schlüssel zugeordnete Wert auf den neuen Wert aktualisiert.

Der folgende Code verwendet die MemoryDB-API, um die Schlüssel hinzuzufügenServiceundRegionmit den Wertenmemorydbundus-east-1beziehungsweise zur Ressourcemy-clusterin der Region 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>
```

Weitere Informationen finden Sie unter [TagResource](#) aus.

Ändern von Tags mithilfe der MemoryDB-API

Sie können die MemoryDB-API verwenden, um die Tags auf einem MemoryDB-Cluster zu ändern.

Den Wert eines Tags ändern:

- Verwenden Sie die Operation [TagResource](#) zum Hinzufügen eines neuen Tags und Wertes oder zum Ändern des Wertes eines vorhandenen Tags.
- Verwenden Sie [UntagResource](#) zum Entfernen von Tags aus der Ressource.

Die Ausgabe beider Operationen ist eine Liste der Tags und deren Werte für die angegebene Ressource.

Entfernen von Tags mithilfe der MemoryDB-API

Sie können die MemoryDB-API verwenden, um Tags aus einem vorhandenen MemoryDB-Cluster zu entfernen, indem Sie die [UntagResource](#) verwenden.

Der folgende Code entfernt mithilfe der MemoryDB-API die Tags mit den Schlüsseln `Service` und `Region` aus dem Cluster `my-cluster` in der Region `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>
```

Verwaltung der Wartung

Jeder Cluster verfügt über ein wöchentliches Wartungsfenster, während dem alle Systemänderungen angewendet werden. Wenn Sie beim Erstellen oder Ändern eines Clusters kein bevorzugtes

Wartungsfenster angeben, weist MemoryDB an einem zufällig ausgewählten Wochentag ein 60-minütiges Wartungsfenster innerhalb des Wartungsfensters der Region zu.

Das 60-minütige Wartungsfenster wird zufällig aus einem 8-Stunden-Zeitraum pro Region ausgewählt. Die folgende Tabelle listet die Blöcke für jede Region auf, von denen die Standard-Wartungsfenster zugewiesen werden. Sie können ein bevorzugtes Wartungsfenster außerhalb des Wartungsfensterblocks der Region auswählen.

Regionscode	Name der Region	Regionale Wartungsfenster
ap-northeast-1	Region Asien-Pazifik (Tokio)	13:00 - 21:00 UHR UTC
ap-northeast-2	Region Asien-Pazifik (Seoul)	12:00 - 20:00 UHR UTC
ap-south-1	Region Asien-Pazifik (Mumbai)	17:30 - 1:30 UHR UTC
ap-southeast-1	Region Asien-Pazifik (Singapur)	14:00 - 22:00 UHR UTC
ap-east-1	Region Asien-Pazifik (Hongkong)	13:00 - 21:00 UHR UTC
ap-southeast-2	Region Asien-Pazifik (Sydney)	12:00 - 20:00 UHR UTC
cn-north-1	Region China (Peking)	14:00 - 22:00 UHR UTC
cn-northwest-1	Region „China (Ningxia)“	14:00 - 22:00 UHR UTC
eu-west-3	Region Europa (Paris)	23:59 - 07:29 UTC
eu-central-1	Region Europa (Frankfurt)	23:00 - 07:00 UHR UTC
eu-west-1	Europe (Ireland) Region	22:00 - 06:00 UHR UTC
eu-west-2	Region Europa (London)	23:00 - 07:00 UHR UTC
sa-east-1	Region Südamerika (São Paulo)	01:00 - 09:00 UHR UTC
ca-central-1	Region Kanada (Zentral)	03:00 - 11:00 UHR UTC
us-east-1	Region USA Ost (Nord-Virginia)	03:00 - 11:00 UHR UTC
us-east-1	Region USA Ost (Ohio)	04:00 - 12:00 UHR UTC

Regionscode	Name der Region	Regionale Wartungsfenster
us-west-1	US West (N. California) Region	06:00 - 14:00 UHR UTC
us-west-2	US West (Oregon) Region	06:00 - 14:00 UHR UTC

Ändern des Wartungsfensters für einen Cluster

Das Wartungsfenster sollte in den Zeitraum mit der geringsten Nutzung fallen und daher unter Umständen von Zeit zu Zeit geändert werden. Sie können den Cluster ändern und einen Zeitraum mit einer Dauer von bis zu 24 Stunden festlegen, in dem alle angeforderten Wartungsaktivitäten durchgeführt werden sollen. In diesem Zeitraum werden alle verzögerten oder ausstehenden Cluster-Änderungen, die Sie angefordert haben, ausgeführt.

Weitere Informationen

Informationen zu Ihrem Wartungsfenster und dem Austausch von Knoten finden Sie unter:

- [Ersetzen von Knoten](#) – Verwalten des Knotenaustauschs
- [Einen MemoryDB-Cluster ändern](#) – Ändern des Wartungsfensters für einen Cluster

Bewährte Methoden

Im Folgenden finden Sie bewährte Methoden. Durch die Einhaltung dieser Methoden lassen sich die Performance und Zuverlässigkeit des Clusters verbessern.

Themen

- [Beschränkte Redis-Befehle](#)
- [Belastbarkeit in MemoryDB für Redis](#)
- [Bewährte te te.](#)
- [Bewährte Methoden: Ändern der Cluster-Größe online](#)

Beschränkte Redis-Befehle

Zur Gewährleistung einer verwalteten Service-Erfahrung beschränkt MemoryDB den Zugriff auf bestimmte Befehle, für die erweiterte Berechtigungen erforderlich sind. Die folgenden Befehle sind nicht:

- `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`

Belastbarkeit in MemoryDB für Redis

Im Zentrum der globalen AWS Infrastruktur stehen die AWS-Regionen und Availability Zones (Verfügbarkeitszonen, AZs). AWS Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die mit Netzwerken mit geringer Latenz, hohem Durchsatz und hochredundanten Vernetzungen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen über AWS Regionen und Availability Zones finden Sie unter [AWS Globale Infrastruktur](#).

Zusätzlich zu dem AWS MemoryDB for Redis bietet verschiedene Funktionen, um Ihren Anforderungen an Ausfallsicherheit und Snapshot-Anforderungen gerecht zu werden.

Themen

- [Minimieren von Ausfällen](#)

Minimieren von Ausfällen

Wenn Sie Ihre MemoryDB für die Redis-Implementierung planen, sollten Sie sie so planen, dass Ausfälle sich minimale Auswirkungen auf Ihre Anwendung und Ihre Daten haben. In diesem Abschnitt werden verschiedene Ansätze vorgestellt, mit denen Sie Ihre Anwendung und Ihre Daten vor Ausfällen schützen können.

Minimieren von Ausfällen: MemoryDB-Cluster

Ein MemoryDB-Cluster besteht aus einem einzelnen primären Knoten, von dem Ihre Anwendung sowohl lesen als auch in diesen schreiben kann, sowie 0 bis 5 Replikationsknoten mit Lesezugriff. Wir empfehlen jedoch dringend, mindestens 1 Replikat für hohe Verfügbarkeit zu verwenden. Wenn Daten in den primären Knoten geschrieben werden, werden diese im Transaktionslog beibehalten und auf den Replikationsknoten asynchron aktualisiert.

Wenn ein Lesereplikat ausfällt,

1. MemoryDB erkennt das ausgefallene Replikat.
2. MemoryDB stellt den ausgefallenen Knoten offline.

3. MemoryDB startet einen Ersatzknoten in derselben AZ und stellt diesen bereit.
4. Der neue Knoten wird mit dem Transaktionslog synchronisiert.

Währenddessen kann die Anwendung weiterhin Lese- und Schreibvorgänge auf den anderen Knoten ausführen.

MemoryDB Multi-AZ

Wenn Multi-AZ in Ihren MemoryDB-Clustern aktiviert ist, werden Ausfälle des primären Knotens erkannt und dieser wird automatisch ersetzt.

1. MemoryDB erkennt den Ausfall des primären Knotens.
2. MemoryDB schlägt zu einem Replikat fehl, nachdem sichergestellt wurde, dass es mit dem ausgefallenen Primär übereinstimmt.
3. MemoryDB erstellt in der AZ des ausgefallenen primären Knotens ein neues Replikat.
4. Der neue Knoten wird mit dem Transaktionslog synchronisiert.

Das Failover zu einem Replikationsknoten erfolgt in der Regel schneller als das Erstellen und Bereitstellen eines neuen primären Knotens. Dadurch kann Ihre Anwendung schneller wieder auf den primären Knoten schreiben.

Weitere Informationen finden Sie unter [Minimierung von Ausfallzeiten in MemoryDB mit Multi-AZ](#).

Bewährte te te.

Wenn Sie Redis Version 7 oder höher verwenden, empfehlen wir die Verwendung von [sharded Pub/Sub](#). Sie verbessern auch den Durchsatz und die Latenz durch das [erweiterte E/A-Multiplexing](#), das bei Verwendung von Redis Version 7 oder höher automatisch verfügbar ist und keine Client-Änderungen erfordert. Es ist ideal für Pub/Sub-Workloads, die bei mehreren Client-Verbindungen häufig an Durchsatz gebunden sind.

Bewährte Methoden: Ändern der Cluster-Größe online

Resharding umfasst das Hinzufügen und Entfernen von Shards oder Knoten für den Cluster sowie die Neuverteilung von Schlüsselräumen. Daher haben viele Aspekte Einfluss auf die Resharding-Operation, z. B. Workload des Clusters, Speichernutzung und allgemeine Datengröße. Für optimale Ergebnisse empfehlen wir, dass Sie die allgemeinen bewährten Methoden zu Clustern für eine gleichmäßige Verteilung von Workload-Verteilung befolgen. Außerdem empfehlen wir, die folgenden Schritte durchzuführen.

Vor dem Beginn des Resharding sollten Sie Folgendes durchführen:

- Testen Sie Ihre Anwendung – Testen Sie das Verhalten Ihrer Anwendung während des Reshardings nach Möglichkeit in einer Staging-Umgebung.
- Erhalten Sie frühzeitige Benachrichtigungen bei Skalierungsproblemen – Resharding ist ein rechenintensiver Vorgang. Aus diesem Grund empfehlen wir, die CPU-Auslastung bei Multicore-Instances während des Reshardings unter 80 Prozent und bei Single-Core-Instances unter 50 Prozent zu halten. Überwachen Sie MemoryDB-Metriken und initiieren Sie das Resharding, bevor Ihre Anwendung Skalierungsprobleme beobachtet. Die Überwachung folgender Metriken ist nützlich: `CPUUtilization`, `NetworkBytesIn`, `NetworkBytesOut`, `CurrConnections`, `NewConnections`, `FreeableMemory`, `SwapUsage` und `BytesUsedForMemoryDB`.
- Stellen Sie vor dem Hochskalieren sicher, dass ausreichend freier Speicher verfügbar ist – Stellen Sie beim Hochskalieren sicher, dass der freie Speicher auf den beizubehaltenden Shards mindestens das 1,5-fache des Arbeitsspeichers beträgt, der auf den Shards verwendet wird, die Sie entfernen möchten.
- Initiieren Sie Resharding außerhalb der Spitzenzeiten – Diese Vorgehensweise hilft, die Auswirkungen auf die Latenz und den Durchsatz auf den Client während des Resharding-Vorgangs zu reduzieren. Außerdem wird das Resharding schneller abgeschlossen, da bei der Slot-Verteilung mehr Ressourcen verwendet werden können.

- Überprüfen Sie das Client-Timeout-Verhalten – Einige Clients stellen möglicherweise eine höhere Latenz während der Online-Cluster-Größenänderung fest. Es kann helfen, bei Ihrer Client-Bibliothek einen höheren Timeout zu konfigurieren, da dem System so Zeit zur Verbindungsherstellung unter höheren Lastbedingungen auf dem Server gegeben wird. Manchmal wird eine große Anzahl an Verbindungen zum Server geöffnet. Fügen Sie in diesen Fällen exponentielles Backoff hinzu, um Logik erneut zu verbinden. Hierdurch wird verhindert, dass ein Schub neuer Verbindungen den Server gleichzeitig erreicht.

Während des Resharding-Vorgangs sollten Sie Folgendes durchführen:

- Vermeiden Sie teure Befehle – Vermeiden Sie die Ausführung rechen- und E/A-intensiver Operationen, wie z. B. die Befehle KEYS und SMEMBERS. Wir empfehlen diesen Ansatz, da diese Operationen die Last auf dem Cluster erhöhen und Einfluss auf die Performance des Clusters haben. Verwenden Sie stattdessen die Befehle SCAN und SSCAN.
- Befolgen Sie die bewährten Methoden von Lua – Vermeiden Sie lange laufende Lua-Skripte und deklarieren Sie Schlüssel, die in Lua-Skripten verwendet werden, immer im Voraus. Wir empfehlen diesen Ansatz, um festzustellen, dass im Lua-Skript keine slotübergreifenden Befehle verwendet werden. Vergewissern Sie sich, dass die in Lua-Skripten verwendeten Schlüssel zum gleichen Slot gehören.

Beachten Sie nach dem Resharding Folgendes:

- Die Skalierung nach oben ist möglicherweise nur zum Teil erfolgreich, wenn auf den Ziel-Shards nicht ausreichend Arbeitsspeicher verfügbar ist. In diesem Fall prüfen Sie den verfügbaren Speicher und wiederholen Sie ggf. die Operation.
- Slots mit großen Elementen werden nicht migriert. Dies gilt besonders für Slots mit Elementen, die nach der Serialisierung größer als 256 MB sind.
- FLUSHALL und FLUSHDB Befehle werden in Lua-Skripten während einer Resharding-Operation nicht unterstützt.

Grundlegendes zur MemoryDB-Replikation

MemoryDB implementiert die Replikation mit Daten, die auf bis zu 500 Shards partitioniert sind.

Jeder Shard in einem Cluster verfügt über einen einzelnen primären Lese-/Schreibknoten und bis zu 5 schreibgeschützte Replikatknoten. Jeder primäre Knoten kann bis zu 100 MB/s unterstützen. Sie

können einen Cluster mit einer höheren Anzahl an Shards und einer geringeren Anzahl an Replikaten mit bis zu 500 Knoten pro Cluster erstellen. Diese Cluster-Konfiguration reicht von 500 Shards und 0 Replikaten bis hin zu 100 Shards und 4 Replikaten, was dem Höchstwert für die Anzahl erlaubter Replikate entspricht.

Konsistenz

In MemoryDB sind die primären Knoten stark konsistent. Erfolgreiche Schreibvorgänge werden dauerhaft in einem verteilten Multi-AZ-Transaktionsprotokoll gespeichert, bevor sie an die Clients zurückgegeben werden. Leseoperationen auf Primärdateien geben immer die meisten up-to-date Daten zurück, was die Auswirkungen aller vorherigen erfolgreichen Schreibvorgänge widerspiegelt. Diese starke Konsistenz wird bei allen primären Failovers beibehalten.

In MemoryDB sind Replikatknoten irgendwann konsistent. Leseoperationen aus Replikaten (mithilfe eines READONLY Befehls) spiegeln möglicherweise nicht immer die Auswirkungen der letzten erfolgreichen Schreibvorgänge wider, da Lag-Metriken in veröffentlicht werden CloudWatch. Lesevorgänge von einem einzelnen Replikat sind jedoch sequentiell konsistent. Erfolgreiche Schreibvorgänge werden für jedes Replikat in derselben Reihenfolge wirksam, in der sie auf dem primären Replikat ausgeführt wurden.

Replikation in einem Cluster

Jedes Read Replikat in einem Shard enthält eine Kopie der Daten vom primären Knoten des Shards. Asynchrone Replikationsmechanismen, die die Transaktionsprotokolle verwenden, werden verwendet, um die Read-Replikate mit den primären zu synchronisieren. Anwendungen können aus jedem Knoten im Cluster lesen. Anwendungen können nur in die primären Knoten schreiben. Read Replicas verbessern die Leseskalierbarkeit. Da MemoryDB die Daten in dauerhaften Transaktionsprotokollen speichert, besteht kein Risiko, dass Daten verloren gehen. Die Daten werden über die Shards in einem Memory-DB-Cluster partitioniert.

Anwendungen verwenden den Cluster-Endpunkt des MemoryDB-Clusters, um eine Verbindung mit den Knoten im Cluster herzustellen. Weitere Informationen finden Sie unter [Verbindungsendpunkte ermitteln](#).

MemoryDB-Cluster sind regional und können nur Knoten aus einer Region enthalten. Um die Fehlertoleranz zu verbessern, müssen Sie Primär- und Lesereplikate für mehrere Availability Zones innerhalb dieser Region bereitstellen.

Die Verwendung der Replikation, die Ihnen Multi-AZ bietet, wird für alle MemoryDB-Cluster dringend empfohlen. Weitere Informationen finden Sie unter [Minimierung von Ausfallzeiten in MemoryDB mit Multi-AZ](#).

Minimierung von Ausfallzeiten in MemoryDB mit Multi-AZ

Es gibt eine Reihe von Fällen, in denen MemoryDB möglicherweise einen primären Knoten ersetzen muss. Dazu gehören bestimmte Arten von geplanten Wartungsarbeiten und der unwahrscheinliche Fall eines Ausfalls eines primären Knotens oder einer Availability Zone.

Die Reaktion auf einen Knotenausfall hängt davon ab, welcher Knoten ausgefallen ist. In allen Fällen stellt MemoryDB jedoch sicher, dass beim Austausch von Knoten oder bei einem Failover keine Daten verloren gehen. Wenn beispielsweise ein Replikat ausfällt, wird der ausgefallene Knoten ersetzt und die Daten werden aus dem Transaktionslog synchronisiert. Wenn der primäre Knoten ausfällt, wird ein Failover zu einem konsistenten Replikat ausgelöst, wodurch sichergestellt wird, dass während des Failovers keine Daten verloren gehen. Die Schreibvorgänge werden jetzt vom neuen primären Knoten aus ausgeführt. Der alte primäre Knoten wird dann ersetzt und aus dem Transaktionsprotokoll synchronisiert.

Wenn ein primärer Knoten auf einem einzelnen Knoten-Shard ausfällt (keine Replikate), akzeptiert MemoryDB keine Schreibvorgänge mehr, bis der primäre Knoten ersetzt und aus dem Transaktionsprotokoll synchronisiert wurde.

Der Austausch von Knoten kann zu einigen Ausfallzeiten für den Cluster führen, aber wenn Multi-AZ aktiv ist, werden die Ausfallzeiten minimiert. Für die Rolle des primären Knotens wird automatisch ein Failover auf eines der Replikate ausgeführt. Es ist nicht erforderlich, einen neuen primären Knoten zu erstellen und bereitzustellen, da MemoryDB dies transparent handhabt. Dieser Failover und die Replikatheraufstufung stellen sicher, dass Sie weiter in den neuen primären Knoten schreiben können, sobald die Heraufstufung abgeschlossen wurde.

Im Falle eines geplanten Knotenaustauschs, der aufgrund von Wartungsupdates oder Service-Updates initiiert wurde, sollten Sie sich darüber im Klaren sein, dass der geplante Knotenaustausch abgeschlossen ist, während der Cluster eingehende Schreibanforderungen bearbeitet.

Multi-AZ auf Ihren MemoryDB-Clustern verbessert Ihre Fehlertoleranz. Dies gilt insbesondere in Fällen, in denen die primären Knoten Ihres Clusters nicht mehr erreichbar sind oder aus irgendeinem Grund ausfallen. Multi-AZ auf MemoryDB-Clustern erfordert, dass jeder Shard über mehr als einen Knoten verfügt und wird automatisch aktiviert.

Themen

- [Fehlerszenarien mit Multi-AZ-Antworten](#)
- [Testen des automatischen Failovers](#)

Fehlerszenarien mit Multi-AZ-Antworten

Wenn Multi-AZ aktiv ist, führt ein ausgefallener primärer Knoten ein Failover auf ein verfügbares Replikat durch. Das Replikat wird automatisch mit dem Transaktionsprotokoll synchronisiert und wird zum primären Knoten, was viel schneller ist, als einen neuen primären Knoten zu erstellen und erneut bereitzustellen. Bei diesem Vorgang dauert gewöhnlich nur wenige Sekunden, bis Sie wieder in den Cluster schreiben können.

Wenn Multi-AZ aktiv ist, überwacht MemoryDB kontinuierlich den Status des primären Knotens. Sollte der primäre Knoten ausfallen, wird abhängig von der Art des Ausfalls eine der folgenden Aktionen durchgeführt.

Themen

- [Fehlerszenarien, wenn nur der Primärknoten ausfällt](#)
- [Ausfallszenarien, wenn der primäre Knoten und einige Replikate ausfallen](#)
- [Fehlerszenarien, wenn der gesamte Cluster ausfällt](#)

Fehlerszenarien, wenn nur der Primärknoten ausfällt

Wenn nur der primäre Knoten ausfällt, wird ein Replikat automatisch zum primären Knoten. Anschließend wird ein Ersatzreplikat in derselben Availability Zone wie das ausgefallene primäre Replikat erstellt und bereitgestellt.

Wenn nur der primäre Knoten ausfällt, geht MemoryDB Multi-AZ wie folgt vor:

1. Der ausgefallene primäre Knoten wird in den Offline-Zustand versetzt.
2. Ein up-to-date Replikat wird automatisch zum primären Replikat.

Schreibvorgänge können fortgesetzt werden, sobald der Failover-Vorgang abgeschlossen ist, in der Regel nur wenige Sekunden.

3. Ein Ersatzreplikat wird gestartet und bereitgestellt.

Das Ersatzreplikat wird in der Availability Zone gestartet, in der sich der ausgefallene primäre Knoten befand, sodass die Verteilung der Knoten beibehalten wird.

4. Das Replikat wird mit dem Transaktionslog synchronisiert.

Weitere Informationen zum Suchen der Endpunkte eines Clusters finden Sie in den folgenden Themen:

- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(MemoryDB-API\)](#)

Ausfallszenarien, wenn der primäre Knoten und einige Replikate ausfallen

Wenn das primäre und mindestens ein Replikat ausfallen, wird ein up-to-date Replikat zum primären Cluster heraufgestuft. Neue Replikate werden außerdem in denselben Availability Zones wie die ausgefallenen Knoten erstellt und bereitgestellt.

Wenn der primäre Knoten und einige Replikate ausfallen, geht MemoryDB Multi-AZ wie folgt vor:

1. Der ausgefallene primäre Knoten und die ausgefallenen Replikate werden offline geschaltet.
2. Ein verfügbares Replikat wird zum primären Knoten.

Schreibvorgänge können fortgesetzt werden, sobald der Failover abgeschlossen ist, in der Regel nur wenige Sekunden.

3. Ersatzreplikate werden erstellt und bereitgestellt.

Die Ersatzreplikate werden in den Availability Zones der ausgefallenen Knoten erstellt, sodass die Verteilung der Knoten erhalten bleibt.

4. Alle Knoten werden mit dem Transaktionsprotokoll synchronisiert.

Weitere Informationen zum Suchen der Endpunkte eines Clusters finden Sie in den folgenden Themen:

- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(AWSCLI\)](#)
- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(MemoryDB-API\)](#)

Fehlerszenarien, wenn der gesamte Cluster ausfällt

Bei einem umfassenden Ausfall werden in denselben Availability Zones, der sich die Originalknoten befanden, alle Knoten neu erstellt und bereitgestellt.

In diesem Szenario gibt es keinen Datenverlust, da die Daten im Transaktionslog gespeichert wurden.

Wenn der gesamte Cluster ausfällt, geht MemoryDB Multi-AZ wie folgt vor:

1. Der ausgefallene primäre Knoten und die Replikate werden offline geschaltet.
2. Ein primärer Ersatzknoten wird erstellt und bereitgestellt, der mit dem Transaktionsprotokoll synchronisiert wird.
3. Ersatzreplikate werden erstellt und bereitgestellt und mit dem Transaktionslog synchronisiert.

Die Ersetzungen werden in den Availability Zones der ausgefallenen Knoten erstellt, sodass die Verteilung der Knoten erhalten bleibt.

Weitere Informationen zum Suchen der Endpunkte eines Clusters finden Sie in den folgenden Themen:

- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(AWSCLI\)](#)
- [Ermitteln des Endpunkts für einen MemoryDB-Cluster \(MemoryDB-API\)](#)

Testen des automatischen Failovers

Sie können das automatische Failover mithilfe der MemoryDB-Konsole, der und der AWS CLI MemoryDB-API testen.

Beim Testen ist Folgendes zu beachten:

- Sie können diesen Vorgang bis zu fünfmal in einem Zeitraum von 24 Stunden verwenden.
- Wenn Sie diesen Vorgang auf Shards in verschiedenen Clustern aufrufen, können Sie die Aufrufe gleichzeitig tätigen.
- In einigen Fällen können Sie diesen Vorgang mehrmals auf verschiedenen Shards im selben MemoryDB-Cluster aufrufen. In solchen Fällen muss die erste Knotenersetzung abgeschlossen werden, bevor ein nachfolgender Aufruf ausgeführt werden kann.
- Um festzustellen, ob der Knotenaustausch abgeschlossen ist, überprüfen Sie die Ereignisse mithilfe der MemoryDB for Redis-KonsoleAWS CLI, der oder der MemoryDB-API. Suchen Sie nach den folgenden Ereignissen im Zusammenhang mitFailoverShard, die hier in der Reihenfolge ihres wahrscheinlichen Eintretens aufgeführt sind:

1. Cluster-Nachricht: FailoverShard API called for shard <shard-id>
2. Cluster-Nachricht: Failover from primary node <primary-node-id> to replica node <node-id> completed
3. Cluster-Nachricht: Recovering nodes <node-id>
4. Cluster-Nachricht: Finished recovery for nodes <node-id>

Weitere Informationen finden Sie unter:

- [DescribeEvents](#) in der MemoryDB-API-Referenz
- Diese API wurde entwickelt, um das Verhalten Ihrer Anwendung im Falle eines MemoryDB-Failovers zu testen. Sie wurde nicht als Betriebstool zum Einleiten eines Failovers konzipiert, um ein Problem mit dem Cluster zu beheben. Darüber hinaus wird unter bestimmten Bedingungen wie umfangreichen Betriebsereignissen die API möglicherweise von AWS blockiert.

Themen

- [Testen des automatischen Failovers mithilfe der AWS Management Console](#)
- [Testen des automatischen Failovers mithilfe der AWS CLI](#)
- [Testen des automatischen Failovers mit der MemoryDB-API](#)

Testen des automatischen Failovers mithilfe der AWS Management Console

Verwenden Sie das folgende Verfahren, um das automatische Failover mit der Konsole zu testen.

1. [Melden Sie sich bei der Konsole von MemoryDB for Redis an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie das Optionsfeld links neben dem Cluster, den Sie testen möchten. Dieser Cluster muss über mindestens einen Replikatknoten verfügen.
3. Bestätigen Sie im Bereich Details, dass dieser Cluster Multi-AZ-fähig ist. Wenn der Cluster nicht Multi-AZ-fähig ist, wählen Sie einen anderen Cluster aus oder bearbeiten Sie diesen Cluster so, dass Multi-AZ aktiviert wird. Weitere Informationen finden Sie unter [Einen MemoryDB-Cluster ändern](#).
4. Wählen Sie den Cluster-Namen aus.
5. Wählen Sie auf der Seite Shards und Nodes für den Shard, auf dem Sie das Failover testen möchten, den Namen des Shards aus.
6. Wählen Sie für den Knoten Failover Primary aus.
7. Wählen Sie Continue, um ein Failover des primären Knotens auszuführen, oder wählen Sie Cancel, um die Operation ohne ein Failover des primären Knotens abubrechen.

Während des Failover-Vorgangs zeigt die Konsole den Status des Knotens weiterhin als `available` an. Um den Status des Failover-Tests zu verfolgen, wählen Sie im Navigationsbereich der Konsole `Events` aus. Suchen Sie auf der Registerkarte `Events` nach Ereignissen, für die angegeben wird, dass Ihr Failover gestartet (`FailoverShard API called`) und abgeschlossen (`Recovery completed`) wurde.

Testen des automatischen Failovers mithilfe der AWS CLI

[Sie können das automatische Failover auf jedem Multi-AZ-fähigen Cluster mithilfe der AWS CLI Operation `Failover-Shard` testen.](#)

Parameter

- `--cluster-name` – Erforderlich. Der Cluster, der getestet werden soll.

- `--shard-name` – Erforderlich. Der Name des Shards, auf dem Sie das automatische Failover testen möchten. Du kannst in einem fortlaufenden Zeitraum von 24 Stunden maximal fünf Shards testen.

Im folgenden Beispiel wird der verwendete AWS CLI, um den Shard `0001` im MemoryDB-Cluster aufzurufen `failover-shard.my-cluster`

Für Linux, macOS oder Unix:

```
aws memorydb failover-shard \  
  --cluster-name my-cluster \  
  --shard-name 0001
```

Für Windows:

```
aws memorydb failover-shard ^  
  --cluster-name my-cluster ^  
  --shard-name 0001
```

Sie können den Status des Failovers mit der AWS CLI-Operation `describe-events` verfolgen.

Es wird die folgende JSON-Antwort zurückgegeben:

```
{  
  "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"  
    }  
  ]  
}
```

Weitere Informationen finden Sie unter:

- [Failover-Shard](#)
- [describe-events](#)

Testen des automatischen Failovers mit der MemoryDB-API

Das folgende Beispiel ruft `FailoverShard` den Shard `0003` im Cluster `memorydb00` auf.

Example Testen des automatischen Failovers

```
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>
```

Verwenden Sie den `DescribeEvents` MemoryDB-API-Vorgang, um den Fortschritt Ihres Failovers zu verfolgen.

Weitere Informationen finden Sie unter:

- [FailoverShard](#)
- [DescribeEvents](#)

Ändern der Anzahl von Replikaten

Sie können die Anzahl der Read Replicas in Ihrem Memory-DB-Cluster mithilfe der AWS Management Console, der oder der MemoryDB-API dynamisch erhöhen oder verringern. AWS CLI Alle Shards müssen dieselbe Anzahl von Replikaten haben.

Erhöhen der Anzahl der Replikate in einem Cluster

Sie können die Anzahl der Replikate in einem Memory-DB-Cluster auf maximal fünf pro Shard erhöhen. Verwenden Sie dazu die AWS Management Console, den oder die AWS CLI MemoryDB-API.

Themen

- [Verwendung der AWS Management Console](#)
- [Verwendung der AWS CLI](#)
- [Verwenden der MemoryDB-API](#)

Verwendung der AWS Management Console

Informationen zur Erhöhung der Anzahl der Replikate in einem MemoryDB-Cluster (Konsole) finden Sie unter [Knoten zu einem Cluster hinzufügen/entfernen](#)

Verwendung der AWS CLI

Um die Anzahl der Replikate in einem Memory-DB-Cluster zu erhöhen, verwenden Sie den `update-cluster`-Befehl mit den folgenden Parametern:

- `--cluster-name` – Erforderlich. Identifiziert, in welchem Cluster Sie die Anzahl der Replikate erhöhen möchten, möchten Sie die Anzahl der Replikate erhöhen möchten,
- `--replica-configuration` – Erforderlich. Ermöglicht Ihnen, die Anzahl der Replikate festzulegen. Um die Anzahl der Replikate zu erhöhen, legen Sie die `ReplicaCount` Eigenschaft auf die Anzahl der Replikate fest, die sich am Ende dieser Operation in diesem Shard befinden sollen, müssen Sie die Eigenschaft auf die Anzahl der Replikate festlegen, die sich in diesem Shard befinden sollen.

Example

Das folgende Beispiel erhöht die Anzahl der Replikate im Cluster `my-cluster` auf 2.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --replica-configuration \  
    ReplicaCount=2
```

Für Windows:

```
aws memorydb update-cluster ^
  --cluster-name my-cluster ^
  --replica-configuration ^
    ReplicaCount=2
```

Es gibt die folgende JSON-Antwort zurück:

```
{
  "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
  }
}
```

Verwenden Sie den folgenden Befehl, um die Details des aktualisierten Clusters einzusehen, sobald sein Status von aktualisiert auf Verfügbar geändert wird:

Für Linux, macOS oder Unix:

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

Für Windows:

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

Es wird die folgende JSON-Antwort zurückgegeben:

```
{
  "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
            }
        },
        {
            "Name": "my-cluster-0002-003",
            "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
            }
        },
        {
            "Name": "my-cluster-0003-003",
            "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
}
]
}

```

Weitere Informationen zur Erhöhung der Anzahl der Replikate mithilfe der CLI finden Sie unter [update-cluster](#) in der Befehlsreferenz. AWS CLI

Verwenden der MemoryDB-API

Verwenden Sie die `UpdateCluster` Aktion mit den folgenden Parametern, um die Anzahl der Replikate in einem MemoryDB-Shard zu erhöhen:

- `ClusterName` – Erforderlich. Identifiziert, in welchem Cluster Sie die Anzahl der Replikate erhöhen möchten, möchten Sie die Anzahl der Replikate erhöhen möchten,
- `ReplicaConfiguration` – Erforderlich. Ermöglicht Ihnen, die Anzahl der Replikate festzulegen. Um die Anzahl der Replikate zu erhöhen, legen Sie die `ReplicaCount` Eigenschaft auf die Anzahl der Replikate fest, die sich am Ende dieser Operation in diesem Shard befinden sollen, müssen Sie die Eigenschaft auf die Anzahl der Replikate festlegen, die sich in diesem Shard befinden sollen.

Example

Das folgende Beispiel erhöht die Anzahl der Replikate im Cluster `sample-cluster` auf drei. Wenn das Beispiel fertig ist, befinden sich in jedem Shard drei Replikate. Diese Zahl gilt unabhängig davon, ob es sich um einen MemoryDB-Cluster mit einem einzelnen Shard oder um einen MemoryDB-Cluster mit mehreren Shards handelt.

```
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>
```

Weitere Hinweise zur Erhöhung der Anzahl der Replikate mithilfe der API finden Sie unter [UpdateCluster](#).

Verringern der Anzahl der Replikate in einem Cluster

Sie können die Anzahl der Replikate in einem Cluster für MemoryDB verringern. Sie können die Anzahl der Replikate auf Null reduzieren, aber Sie können kein Failover auf ein Replikat durchführen, wenn Ihr primärer Knoten ausfällt.

Sie können die AWS Management Console, die AWS CLI oder die MemoryDB-API verwenden, um die Anzahl der Replikate in einem Cluster zu verringern.

Themen

- [Verwendung der AWS Management Console](#)
- [Verwendung der AWS CLI](#)
- [Verwenden der MemoryDB-API](#)

Verwendung der AWS Management Console

Informationen zum Verringern der Anzahl der Replikate in einem MemoryDB-Cluster (Konsole) finden Sie unter [Knoten zu einem Cluster hinzufügen/entfernen](#)

Verwendung der AWS CLI

Um die Anzahl der Replikate in einem Memory-DB-Cluster zu verringern, verwenden Sie den `update-cluster`-Befehl mit den folgenden Parametern:

- `--cluster-name` – Erforderlich. Identifiziert, in welchem Cluster Sie die Anzahl der Replikate verringern möchten.
- `--replica-configuration` – Erforderlich.

`ReplicaCount`— Stellen Sie diese Eigenschaft ein, um die Anzahl der von Ihnen gewünschten Replikatknoten anzugeben.

Example

Das folgende Beispiel verwendet `--replica-configuration`, um die Anzahl der Replikate im Cluster `my-cluster` auf den angegebenen Wert zu reduzieren.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \
```

```
--cluster-name my-cluster \  
--replica-configuration \  
    ReplicaCount=1
```

Für Windows:

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

Es wird die folgende JSON-Antwort zurückgegeben:

```
{  
  "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  
  }  
}
```

Verwenden Sie den folgenden Befehl, um die Details des aktualisierten Clusters einzusehen, sobald sein Status von aktualisiert auf Verfügbar geändert wird:

Für Linux, macOS oder Unix:

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

Für Windows:

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

Es wird die folgende JSON-Antwort zurückgegeben:

```
{  
  "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
}
]
}

```

Weitere Informationen zur Verringerung der Anzahl der Replikate mithilfe der CLI finden Sie unter [update-cluster](#) in der Befehlsreferenz. AWS CLI

Verwenden der MemoryDB-API

Verwenden Sie die `UpdateCluster` Aktion mit den folgenden Parametern, um die Anzahl der Replikate in einem MemoryDB-Cluster zu verringern:

- `ClusterName` – Erforderlich. Identifiziert, in welchem Cluster Sie die Anzahl der Replikate verringern möchten.
- `ReplicaConfiguration` – Erforderlich. Ermöglicht Ihnen, die Anzahl der Replikate festzulegen.

`ReplicaCount`— Stellen Sie diese Eigenschaft ein, um die Anzahl der von Ihnen gewünschten Replikatknoten anzugeben.

Example

Das folgende Beispiel verwendet `ReplicaCount`, um die Anzahl der Replikate im Cluster `sample-cluster` auf eins zu reduzieren. Wenn das Beispiel fertig ist, befindet sich in jedem Shard ein Replikat. Diese Zahl gilt unabhängig davon, ob es sich um einen MemoryDB-Cluster mit einem einzelnen Shard oder um einen MemoryDB-Cluster mit mehreren Shards handelt.

```
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>
```

Weitere Hinweise zur Verringerung der Anzahl der Replikate mithilfe der API finden Sie unter [UpdateCluster](#).

Snapshot und Wiederherstellung

MemoryDB for Redis-Cluster sichern Daten automatisch in einem Multi-AZ-Transaktionsprotokoll. Sie können jedoch wählen, ob Sie Snapshots eines Clusters entweder regelmäßig oder bei Bedarf erstellen point-in-time möchten. Diese Snapshots können verwendet werden, um einen Cluster zu einem früheren Zeitpunkt neu zu erstellen oder um einen brandneuen Cluster zu erstellen. Der Snapshot besteht aus den Metadaten des Clusters sowie allen Daten im Cluster. Alle Snapshots werden in Amazon Simple Storage Service (Amazon S3) geschrieben, der dauerhaften Speicher bietet. Sie können Ihre Daten jederzeit wiederherstellen, indem Sie einen neuen MemoryDB-Cluster erstellen und ihn mit Daten aus einem Snapshot füllen. Mit MemoryDB können Sie Snapshots mithilfe der API, der AWS Command Line Interface (AWS CLI) und der AWS Management Console MemoryDB-API verwalten.

Themen

- [Snapshot-Einschränkungen](#)

- [Snapshot-Kosten](#)
- [Automatische Snapshots planen](#)
- [Manuelle Snapshots erstellen](#)
- [Einen endgültigen Snapshot erstellen](#)
- [Beschreibung von Schnappschüssen](#)
- [Kopieren eines Snapshots](#)
- [Einen Snapshot exportieren](#)
- [Wiederherstellung aus einem Snapshot](#)
- [Einen neuen Cluster mit einem extern erstellten Snapshot erstellen](#)
- [Schnappschüsse taggen](#)
- [Löschen eines Snapshots](#)

Snapshot-Einschränkungen

Beachten Sie bei der Planung oder Erstellung von Snapshots die folgenden Einschränkungen:

- Für MemoryDB-Cluster sind Snapshot und Restore für alle unterstützten Knotentypen verfügbar.
- In einem zusammenhängenden Zeitraum von 24 Stunden können Sie nicht mehr als 20 manuelle Snapshots pro Cluster erstellen.
- MemoryDB unterstützt nur das Erstellen von Snapshots auf Clusterebene. MemoryDB unterstützt nicht das Erstellen von Snapshots auf Shard- oder Knotenebene.
- Während des Snapshot-Vorgangs können Sie keine anderen API- oder CLI-Operationen auf dem Cluster ausführen.
- Wenn Sie einen Cluster löschen und einen endgültigen Snapshot anfordern, erstellt MemoryDB den Snapshot immer von den primären Knoten. Dadurch wird sichergestellt, dass Sie die allerneuesten Daten erfassen, bevor der Cluster gelöscht wird.

Snapshot-Kosten

Mit MemoryDB können Sie einen Snapshot für jeden aktiven MemoryDB-Cluster kostenlos speichern. Speicherplatz für zusätzliche Snapshots wird für alle Regionen mit 0,085 USD/GB pro Monat berechnet. AWS Für die Erstellung eines Snapshots oder für die Wiederherstellung von Daten aus einem Snapshot in einem MemoryDB-Cluster fallen keine Datenübertragungsgebühren an.

Automatische Snapshots planen

Für jeden MemoryDB-Cluster können Sie automatische Snapshots aktivieren. Wenn automatische Snapshots aktiviert sind, erstellt MemoryDB täglich einen Snapshot des Clusters. Es gibt keine Auswirkungen auf den Cluster, und die Änderung erfolgt sofort. Weitere Informationen finden Sie unter [Wiederherstellung aus einem Snapshot](#).

Wenn Sie automatische Snapshots planen, sollten Sie die folgenden Einstellungen planen:

- **Snapshot-Fenster** — Ein Zeitraum an jedem Tag, an dem MemoryDB mit der Erstellung eines Snapshots beginnt. Die Mindestlänge für das Snapshot-Fenster beträgt 60 Minuten. Sie können das Snapshot-Fenster so einstellen, dass es für Sie am bequemsten ist, oder für eine Tageszeit, zu der Snapshots in Zeiten mit besonders hoher Auslastung vermieden werden.

Wenn Sie kein Snapshot-Fenster angeben, weist MemoryDB automatisch eines zu.

- **Aufbewahrungslimit für Snapshots** — Die Anzahl der Tage, an denen der Snapshot in Amazon S3 aufbewahrt wird. Wenn Sie beispielsweise das Aufbewahrungslimit auf 5 festlegen, wird ein heute aufgenommener Snapshot 5 Tage lang aufbewahrt. Wenn das Aufbewahrungslimit abläuft, wird der Snapshot automatisch gelöscht.

Das maximale Aufbewahrungslimit für Snapshots beträgt 35 Tage. Wenn das Aufbewahrungslimit für Snapshots auf 0 festgelegt ist, sind automatische Snapshots für den Cluster deaktiviert. MemoryDB-Daten sind auch bei deaktiviertem automatischem Snapshoting weiterhin vollständig dauerhaft.

Sie können automatische Snapshots aktivieren oder deaktivieren, wenn Sie einen MemoryDB-Cluster mithilfe der MemoryDB-Konsole, der oder der MemoryDB-API erstellen. AWS CLI Sie können automatische Snapshots aktivieren, wenn Sie einen MemoryDB-Cluster erstellen, indem Sie im Abschnitt Snapshots das Kontrollkästchen Automatische Backups aktivieren aktivieren aktivieren. Weitere Informationen finden Sie unter [Einen MemoryDB-Cluster erstellen](#).

Manuelle Snapshots erstellen

Zusätzlich zu automatischen Snapshots können Sie jederzeit einen manuellen Snapshot erstellen. Im Gegensatz zu automatischen Snapshots, die nach einem bestimmten Aufbewahrungszeitraum automatisch gelöscht werden, haben manuelle Snapshots keine Aufbewahrungsfrist, nach deren Ablauf sie automatisch gelöscht werden. Sie müssen jeden manuellen Snapshot manuell löschen. Selbst wenn Sie einen Cluster oder Knoten löschen, bleiben alle manuellen Snapshots von diesem Cluster oder Knoten erhalten. Wenn Sie einen manuellen Snapshot nicht mehr behalten möchten, müssen Sie ihn explizit selbst löschen.

Manuelle Snapshots eignen sich zum Testen und Archivieren. Angenommen, Sie haben beispielsweise einen Satz grundlegender Daten für Testzwecke entwickelt. Sie können einen manuellen Snapshot der Daten erstellen und ihn jederzeit wiederherstellen. Nachdem Sie eine Anwendung getestet haben, die die Daten ändert, können Sie die Daten zurücksetzen, indem Sie einen neuen Cluster erstellen und die Daten anhand Ihres Basis-Snapshots wiederherstellen. Wenn der Cluster bereit ist, können Sie Ihre Anwendungen erneut mit den grundlegenden Daten testen und Sie können diesen Vorgang so oft wie nötig wiederholen.

Sie können nicht nur direkt einen manuellen Snapshot erstellen, sondern auch auf eine der folgenden Arten einen manuellen Snapshot erstellen:

- [Kopieren eines Snapshots](#)— Es spielt keine Rolle, ob der Quell-Snapshot automatisch oder manuell erstellt wurde.
- [Einen endgültigen Snapshot erstellen](#)— Erstellen Sie unmittelbar vor dem Löschen eines Clusters einen Snapshot.

Andere wichtige Themen

- [Snapshot-Einschränkungen](#)
- [Snapshot-Kosten](#)

Sie können mithilfe der [AWS CLI](#), der [AWS Management Console](#) oder der [MemoryDB-API](#) einen manuellen Snapshot eines Knotens erstellen.

Einen manuellen Snapshot erstellen (Konsole)

Um einen Snapshot eines Clusters (Konsole) zu erstellen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)

2. wählen Sie im linken Navigationsbereich Clusters aus.

Der Bildschirm MemoryDB-Clusters wird angezeigt.

3. wählen Sie das Optionsfeld links neben dem Namen des MemoryDB-Clusters, den Sie sichern möchten.
4. Wählen Sie Aktionen und dann Snapshot erstellen aus.
5. Geben Sie im Snapshot-Fenster einen Namen für Ihren Snapshot in das Feld Snapshot-Name ein. Wir empfehlen, dass der Name angibt, welcher Cluster gesichert wurde und an welchem Datum und zu welcher Uhrzeit der Snapshot erstellt wurde.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
6. Wählen Sie unter Verschlüsselung aus, ob Sie einen Standard-Verschlüsselungsschlüssel oder einen vom Kunden verwalteten Schlüssel verwenden möchten. Weitere Informationen finden Sie unter [Verschlüsselung während der Übertragung \(TLS\) in MemoryDB](#).
 7. Fügen Sie unter Tags optional Tags hinzu, um Ihre Schnappschüsse zu durchsuchen und zu filtern oder Ihre AWS Kosten nachzuverfolgen.
 8. Wählen Sie Take Snapshot (Snapshot erstellen) aus.

Der Status des Clusters ändert sich in snapshotting. Wenn der Status wieder verfügbar ist, ist der Snapshot abgeschlossen.

Manuellen Snapshot erstellen (AWS CLI)

Um mithilfe von einen manuellen Snapshot eines Clusters zu erstellen AWS CLI, verwenden Sie den `create-snapshot` AWS CLI Vorgang mit den folgenden Parametern:

- `--cluster-name`— Name des MemoryDB-Clusters, der als Quelle für den Snapshot verwendet werden soll. Verwenden Sie diesen Parameter, wenn Sie einen MemoryDB-Cluster sichern.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
-
- `--snapshot-name` – der Name des zu erstellenden Snapshots.

Verwandte Themen

Weitere Informationen finden Sie unter `create-snapshot` in der Referenz zum AWS CLI -Befehl.

Erstellen eines manuellen Snapshots (MemoryDB-API)

Um mithilfe der MemoryDB-API einen manuellen Snapshot eines Clusters zu erstellen, verwenden Sie den `CreateSnapshot` MemoryDB-API-Vorgang mit den folgenden Parametern:

- `ClusterName`— Name des MemoryDB-Clusters, der als Quelle für den Snapshot verwendet werden soll. Verwenden Sie diesen Parameter, wenn Sie einen MemoryDB-Cluster sichern.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
-
- `SnapshotName` – der Name des zu erstellenden Snapshots.

Verwandte Themen

Weitere Informationen finden Sie unter [CreateSnapshot](#)

Einen endgültigen Snapshot erstellen

Sie können einen endgültigen Snapshot mit der MemoryDB-Konsole, der AWS CLI, der oder der MemoryDB-API erstellen.

Erstellen eines endgültigen Snapshots (Konsole)

Sie können einen endgültigen Snapshot erstellen, wenn Sie einen MemoryDB-Cluster mit der MemoryDB-Konsole löschen.

Um beim Löschen eines MemoryDB-Clusters einen endgültigen Snapshot zu erstellen, wählen Sie auf der Löschseite Ja und geben Sie dem Snapshot einen Namen unter. [Schritt 4: Löschen eines Clusters](#)

Erstellen eines endgültigen Snapshots (AWS CLI)

Sie können einen endgültigen Snapshot erstellen, wenn Sie einen MemoryDB-Cluster mit dem löschen. AWS CLI

Beim Löschen eines MemoryDB-Clusters

Verwenden Sie den `delete-cluster` AWS CLI Vorgang mit den folgenden Parametern, um beim Löschen eines Clusters einen endgültigen Snapshot zu erstellen:

- `--cluster-name` Name des zu löschenden Clusters.
- `--final-snapshot-name`— Name des endgültigen Snapshots.

Mit dem folgenden Code wird `bkup-20210515-final` beim Löschen des Clusters der endgültige Snapshot erstellt `myCluster`.

Für Linux, macOS oder Unix:

```
aws memorydb delete-cluster \  
    --cluster-name myCluster \  
    --final-snapshot-name bkup-20210515-final
```

Für Windows:

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

```
--final-snapshot-name bkup-20210515-final
```

Weitere Informationen finden Sie unter [delete-cluster](#) in der AWS CLI Befehlsreferenz.

Erstellen eines endgültigen Snapshots (MemoryDB-API)

Sie können einen endgültigen Snapshot erstellen, wenn Sie einen MemoryDB-Cluster mithilfe der MemoryDB-API löschen.

Beim Löschen eines MemoryDB-Clusters

Verwenden Sie den `DeleteCluster` MemoryDB-API-Vorgang mit den folgenden Parametern, um einen endgültigen Snapshot zu erstellen.

- `ClusterName` Name des zu löschenden Clusters.
- `FinalSnapshotName`— Name des Snapshots.

Der folgende MemoryDB-API-Vorgang erstellt den Snapshot, `bkup-20210515-final` wenn der Cluster gelöscht wird. `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>
```

Weitere Informationen finden Sie unter [DeleteCluster](#)

Beschreibung von Schnappschüssen

Die folgenden Verfahren zeigen Ihnen, wie Sie eine Liste Ihrer Schnappschüsse anzeigen. Wenn Sie möchten, können Sie sich auch die Details eines bestimmten Snapshots ansehen.

Beschreibung von Snapshots (Konsole)

Um Schnappschüsse mit dem anzuzeigen AWS Management Console

1. Loggen Sie sich in die Konsole ein
2. wählen Sie im linken Navigationsbereich Snapshots aus.
3. Verwenden Sie die Suche, um nach manuellen, automatischen oder allen Schnappschüssen zu filtern.
4. Um die Details eines bestimmten Snapshots zu sehen, wählen Sie das Optionsfeld links neben dem Namen des Snapshots. Wählen Sie „Aktionen“ und dann „Details anzeigen“.
5. Optional können Sie auf der Seite „Details anzeigen“ zusätzliche Snapshot-Aktionen wie Kopieren, Wiederherstellen oder Löschen ausführen. Sie können dem Snapshot auch Tags hinzufügen

Schnappschüsse beschreiben (AWS CLI)

Verwenden Sie den `describe-snapshots` CLI-Vorgang, um eine Liste von Snapshots und optional Details zu einem bestimmten Snapshot anzuzeigen.

Beispiele

Der folgende Vorgang verwendet den Parameter `--max-results`, um bis zu 20 Snapshots aufzulisten, die Ihrem Konto zugeordnet sind. Wenn Sie den Parameter weglassen, `--max-results` werden bis zu 50 Schnappschüsse aufgeführt.

```
aws memorydb describe-snapshots --max-results 20
```

Der folgende Vorgang verwendet den Parameter, `--cluster-name` um nur die Snapshots aufzulisten, die dem Cluster zugeordnet sind. `my-cluster`

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

Der folgende Vorgang verwendet den Parameter `--snapshot-name`, um die Details des Snapshots `my-snapshot` anzuzeigen.

```
aws memorydb describe-snapshots --snapshot-name my-snapshot
```

Weitere Informationen finden Sie unter [describe-snapshots](#).

Beschreibung von Snapshots (MemoryDB-API)

Verwenden Sie den Vorgang, um eine Liste von Snapshots anzuzeigen. `DescribeSnapshots`

Beispiele

Der folgende Vorgang verwendet den Parameter `MaxResults`, um bis zu 20 Snapshots aufzulisten, die Ihrem Konto zugeordnet sind. Wenn Sie den Parameter weglassen, `MaxResults` werden bis zu 50 Schnappschüsse aufgeführt.

```
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>
```

Der folgende Vorgang verwendet den Parameter `ClusterName`, um alle Snapshots aufzulisten, die dem Cluster zugeordnet sind. `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>
```

Der folgende Vorgang verwendet den Parameter `SnapshotName`, um die Details für den Snapshot `MyBackup` anzuzeigen.

```
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>
```

Weitere Informationen finden Sie unter [DescribeSnapshots](#).

Kopieren eines Snapshots

Sie können von jedem Snapshot eine Kopie erstellen, unabhängig davon, ob er automatisch oder manuell erstellt wurde. Beim Kopieren eines Snapshots wird derselbe KMS-Verschlüsselungsschlüssel wie die Quelle für das Ziel verwendet, sofern er nicht ausdrücklich überschrieben wird. Sie können Ihren Snapshot auch exportieren, sodass Sie von außerhalb von MemoryDB darauf zugreifen können. Hinweise zum Exportieren Ihres Snapshots finden Sie unter [Einen Snapshot exportieren](#)

Die folgenden Verfahren zeigen Ihnen, wie Sie einen Snapshot kopieren.

Einen Snapshot kopieren (Konsole)

Um einen Snapshot zu kopieren (Konsole)

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](#)
2. Um eine Liste Ihrer Snapshots zu sehen, wählen Sie im linken Navigationsbereich Snapshots aus.
3. Wählen Sie in der Liste der Snapshots das Optionsfeld links neben dem Namen des Snapshots aus, den Sie kopieren möchten.
4. Wählen Sie „Aktionen“ und anschließend „Kopieren“.
5. Gehen Sie auf der Seite „Snapshot kopieren“ wie folgt vor:
 - a. Geben Sie im Feld Neuer Snapshot-Name einen Namen für Ihren neuen Snapshot ein.
 - b. Lassen Sie das optionale Feld Target S3 Bucket leer. Dieses Feld sollte nur zum Exportieren Ihres Snapshots verwendet werden und erfordert spezielle S3-Berechtigungen. Informationen zum Exportieren eines Snapshots finden Sie unter [Einen Snapshot exportieren](#).
 - c. Wählen Sie, ob Sie den AWS KMS Standard-Verschlüsselungsschlüssel oder einen benutzerdefinierten Schlüssel verwenden möchten. Weitere Informationen finden Sie unter [Verschlüsselung während der Übertragung \(TLS\) in MemoryDB](#).
 - d. Optional können Sie der Snapshot-Kopie auch Tags hinzufügen.
 - e. Wählen Sie die Option Kopieren aus.

Einen Snapshot kopieren (AWS CLI)

Verwenden Sie den `copy-snapshot` Vorgang, um einen Snapshot zu kopieren.

Parameter

- `--source-snapshot-name`— Name des zu kopierenden Snapshots.
- `--target-snapshot-name`— Name der Kopie des Snapshots.
- `--target-bucket`— Reserviert für den Export eines Snapshots. Verwenden Sie diesen Parameter nicht, wenn Sie eine Kopie eines Snapshots erstellen. Weitere Informationen finden Sie unter [Einen Snapshot exportieren](#).

Im folgenden Beispiel wird eine Kopie eines automatischen Snapshots erstellt.

Für Linux, macOS oder Unix:

```
aws memorydb copy-snapshot \  
  --source-snapshot-name automatic.my-primary-2021-03-27-03-15 \  
  --target-snapshot-name my-snapshot-copy
```

Für Windows:

```
aws memorydb copy-snapshot ^  
  --source-snapshot-name automatic.my-primary-2021-03-27-03-15 ^  
  --target-snapshot-name my-snapshot-copy
```

Weitere Informationen finden Sie unter [copy-snapshot](#).

Einen Snapshot kopieren (MemoryDB-API)

Verwenden Sie den `copy-snapshot` Vorgang mit den folgenden Parametern, um einen Snapshot zu kopieren:

Parameter

- `SourceSnapshotName`— Name des zu kopierenden Snapshots.
- `TargetSnapshotName`— Name der Kopie des Snapshots.
- `TargetBucket`— Reserviert für den Export eines Snapshots. Verwenden Sie diesen Parameter nicht, wenn Sie eine Kopie eines Snapshots erstellen. Weitere Informationen finden Sie unter [Einen Snapshot exportieren](#).

Im folgenden Beispiel wird eine Kopie eines automatischen Snapshots erstellt.

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>
```

Weitere Informationen finden Sie unter [CopySnapshot](#).

Einen Snapshot exportieren

MemoryDB for Redis unterstützt den Export Ihres MemoryDB-Snapshots in einen Amazon Simple Storage Service (Amazon S3) -Bucket, sodass Sie von außerhalb von MemoryDB darauf zugreifen können. Exportierte MemoryDB-Snapshots sind vollständig kompatibel mit Open-Source-Redis und können mit der entsprechenden Redis-Version oder den entsprechenden Redis-Tools geladen werden. Sie können einen Snapshot mit der MemoryDB-Konsole, der oder der MemoryDB-API exportieren. AWS CLI

Das Exportieren eines Snapshots kann hilfreich sein, wenn Sie einen Cluster in einer anderen Region starten müssen. AWS Sie können Ihre Daten in eine AWS Region exportieren, die RDB-Datei in die neue AWS Region kopieren und dann diese RDB-Datei verwenden, um den neuen Cluster zu starten, anstatt darauf zu warten, dass der neue Cluster durch Use gefüllt wird. Informationen zum Erstellen eines neuen Clusters durch Seeding finden Sie unter [Einen neuen Cluster mit einem extern erstellten Snapshot erstellen](#). Ein weiterer möglicher Grund für das Exportieren der Daten eines Clusters besteht in der Verwendung der .rdb-Datei zur Offline-Verarbeitung.

Important

- Der MemoryDB-Snapshot und der Amazon S3 S3-Bucket, in den Sie ihn kopieren möchten, müssen sich in derselben AWS Region befinden.

Obwohl in einen Amazon S3 S3-Bucket kopierte Snapshots verschlüsselt sind, empfehlen wir dringend, anderen keinen Zugriff auf den Amazon S3 S3-Bucket zu gewähren, in dem Sie Ihre Snapshots speichern möchten.

- Das Exportieren eines Snapshots nach Amazon S3 wird für Cluster, die Data Tiering verwenden, nicht unterstützt. Weitere Informationen finden Sie unter [Daten-Tiering](#).

Bevor Sie einen Snapshot in einen Amazon S3 S3-Bucket exportieren können, müssen Sie über einen Amazon S3 S3-Bucket in derselben AWS Region wie der Snapshot verfügen. Gewähren Sie MemoryDB-Zugriff auf den Bucket. Die ersten beiden Schritte zeigen, wie Sie dabei vorgehen.

Warning

Die folgenden Szenarien legen Ihre Daten auf möglicherweise unerwünschte Weise offen.

- Wenn eine andere Person Zugriff auf den Amazon S3 S3-Bucket hat, in den Sie Ihren Snapshot exportiert haben.

Um den Zugriff auf Ihre Snapshots zu kontrollieren, gewähren Sie nur denjenigen Zugriff auf den Amazon S3 S3-Bucket, die Sie auf Ihre Daten zugreifen möchten. Informationen zum Verwalten des Zugriffs auf einen Amazon-S3-Bucket finden Sie unter [Zugriffsverwaltung](#) im Entwicklerhandbuch zu Amazon S3.

- Wenn eine andere Person berechtigt ist, den CopySnapshot API-Vorgang zu verwenden.

Benutzer oder Gruppen, die über Berechtigungen zur Nutzung des CopySnapshot API-Vorgangs verfügen, können ihre eigenen Amazon S3 S3-Buckets erstellen und Snapshots in diese kopieren. Um den Zugriff auf Ihre Snapshots zu kontrollieren, verwenden Sie eine AWS Identity and Access Management (IAM-) Richtlinie, um zu kontrollieren, wer die API verwenden darf. CopySnapshot Weitere Informationen zur Verwendung von IAM zur Steuerung der Verwendung von MemoryDB-API-Vorgängen finden Sie [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#) im MemoryDB-Benutzerhandbuch.

Themen

- [Schritt 1: Einen Amazon-S3-Bucket erstellen](#)
- [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#)
- [Schritt 3: Exportieren Sie einen MemoryDB-Snapshot](#)

Schritt 1: Einen Amazon-S3-Bucket erstellen

Das folgende Verfahren verwendet die Amazon S3 S3-Konsole, um einen Amazon S3 S3-Bucket zu erstellen, in den Sie Ihren MemoryDB-Snapshot exportieren und speichern.

So erstellen Sie einen Amazon-S3-Bucket

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie Create Bucket (Bucket erstellen) aus.
3. Verfahren Sie unter Create a Bucket – Select a Bucket Name and Region wie folgt:
 - a. Geben Sie für Bucket-Name einen Namen für Ihren Amazon-S3-Bucket ein.

- b. Wählen Sie aus der Regionsliste eine AWS Region für Ihren Amazon S3 S3-Bucket aus. Diese AWS Region muss dieselbe AWS Region sein wie der MemoryDB-Snapshot, den Sie exportieren möchten.
- c. Wählen Sie Erstellen aus.

Weitere Informationen zum Erstellen eines Amazon-S3-Buckets finden Sie unter [Erstellen von Buckets](#) im Handbuch für Amazon Simple Storage Service.

Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket

AWS Regionen, die vor dem 20. März 2019 eingeführt wurden, sind standardmäßig aktiviert. Sie können sofort mit der Arbeit in diesen AWS Regionen beginnen. Regionen, die nach dem 20. März 2019 eingeführt wurden, sind standardmäßig deaktiviert. Sie müssen diese Regionen aktivieren oder sich für sie anmelden, bevor Sie sie verwenden können, wie unter [AWS Regionen verwalten](#) beschrieben.

Gewähren Sie MemoryDB-Zugriff auf Ihren S3-Bucket in einer Region AWS

Gehen Sie wie folgt vor, um die richtigen Berechtigungen für einen Amazon S3 S3-Bucket in einer AWS Region zu erstellen.

Um MemoryDB-Zugriff auf einen S3-Bucket zu gewähren

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie den Namen des Amazon S3 S3-Buckets, in den Sie den Snapshot kopieren möchten. Dies sollte der in [Schritt 1: Einen Amazon-S3-Bucket erstellen](#) erstellte S3-Bucket sein.
3. Wählen Sie den Tab Permissions und dann unter Permissions die Option Bucket Policy aus.
4. Aktualisieren Sie die Richtlinie, um MemoryDB die erforderlichen Berechtigungen zur Ausführung von Vorgängen zu gewähren:
 - Fügen Sie ["Service" : "*region-full-name*.memorydb-snapshot.amazonaws.com"] zu Principal hinzu.
 - Fügen Sie die folgenden Berechtigungen hinzu, die zum Exportieren eines Snapshots in den Amazon-S3-Bucket erforderlich sind.
 - "s3:PutObject"
 - "s3:GetObject"

- "s3:ListBucket"
- "s3:GetBucketAcl"
- "s3:ListMultipartUploadParts"
- "s3:ListBucketMultipartUploads"

Nachfolgend finden Sie ein Beispiel dafür, wie die aktualisierte Richtlinie aussehen könnte.

```
{
  "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/*"
      ]
    }
  ]
}
```

Schritt 3: Exportieren Sie einen MemoryDB-Snapshot

Jetzt haben Sie Ihren S3-Bucket erstellt und MemoryDB-Zugriffsberechtigungen erteilt. Ändern Sie den S3-Objektbesitz auf ACLs aktiviert — Bucket-Besitzer bevorzugt. Als Nächstes können Sie die MemoryDB-Konsole, die AWS CLI oder die MemoryDB-API verwenden, um Ihren Snapshot dorthin

zu exportieren. Im Folgenden wird davon ausgegangen, dass Sie über die folgenden, zusätzlichen S3-spezifischen IAM-Berechtigungen verfügen.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "s3:GetBucketLocation",
      "s3:ListAllMyBuckets",
      "s3:PutObject",
      "s3:GetObject",
      "s3:DeleteObject",
      "s3:ListBucket"
    ],
    "Resource": "arn:aws:s3:::*"
  }]
}
```

Exportieren eines MemoryDB-Snapshots (Konsole)

Der folgende Prozess verwendet die MemoryDB-Konsole, um einen Snapshot in einen Amazon S3 S3-Bucket zu exportieren, sodass Sie von außerhalb von MemoryDB darauf zugreifen können. Der Amazon S3 S3-Bucket muss sich in derselben AWS Region wie der MemoryDB-Snapshot befinden.

So exportieren Sie einen MemoryDB-Snapshot in einen Amazon S3 S3-Bucket

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Um eine Liste Ihrer Snapshots zu sehen, wählen Sie im linken Navigationsbereich Snapshots aus.
3. Wählen Sie in der Liste der Snapshots das Optionsfeld links neben dem Namen des Snapshots aus, den Sie exportieren möchten.
4. Wählen Sie die Option Kopieren aus.
5. Gehen Sie im Dialogfeld Create a Copy of the Backup (Eine Kopie der Sicherung erstellen?) wie folgt vor:
 - a. Geben Sie im Feld Neuer Snapshot-Name einen Namen für Ihren neuen Snapshot ein.

Der Name muss zwischen 1 und 1.000 Zeichen lang sein und UTF-8-codierbar sein.

MemoryDB fügt dem Wert, den Sie hier eingeben `.rdb`, eine Shard-ID und hinzu. Wenn Sie beispielsweise eingeben, erstellt MemoryDB `my-exported-snapshot.my-exported-snapshot-0001.rdb`

- b. Wählen Sie aus der Liste Ziel-S3-Speicherort den Namen des Amazon S3 S3-Buckets aus, in den Sie Ihren Snapshot kopieren möchten (den Bucket, in dem Sie ihn erstellt haben [Schritt 1: Einen Amazon-S3-Bucket erstellen](#)).

Der Ziel-S3-Standort muss ein Amazon S3 S3-Bucket in der AWS Region des Snapshots mit den folgenden Berechtigungen sein, damit der Exportvorgang erfolgreich ist.

- Objektzugriff – Lesen und Schreiben.
- Berechtigungszugriff – Lesen.

Weitere Informationen finden Sie unter [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#).

- c. Wählen Sie die Option Kopieren aus.

Note

Wenn Ihr S3-Bucket nicht über die erforderlichen Berechtigungen verfügt, damit MemoryDB einen Snapshot dorthin exportieren kann, erhalten Sie eine der folgenden Fehlermeldungen. Kehren Sie zurück zu [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#), um die angegebenen Berechtigungen hinzuzufügen, und versuchen Sie erneut, Ihren Snapshot zu exportieren.

- MemoryDB wurden keine LESEberechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Leseberechtigungen für den Bucket hinzu.

- MemoryDB wurden keine Schreibberechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Schreibberechtigungen für den Bucket hinzu.

- MemoryDB wurden keine READ_ACP-Berechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Read-Zugriff für Berechtigungen für den Bucket hinzu.

Wenn Sie Ihren Snapshot in eine andere AWS Region kopieren möchten, verwenden Sie Amazon S3, um ihn zu kopieren. Weitere Informationen finden Sie unter [Objekte kopieren](#) im Amazon Simple Storage Service-Benutzerhandbuch.

Exportieren eines MemoryDB-Snapshots (CLI)AWS

Exportieren Sie den Snapshot mithilfe der `copy-snapshot` CLI-Operation mit den folgenden Parametern in einen Amazon S3 S3-Bucket:

Parameter

- `--source-snapshot-name`— Name des zu kopierenden Snapshots.
- `--target-snapshot-name`— Name der Kopie des Snapshots.

Der Name muss zwischen 1 und 1.000 Zeichen lang sein und UTF-8-codierbar sein.

MemoryDB fügt dem hier eingegebenen Wert eine Shard-ID und `.rdb` hinzu. Wenn Sie beispielsweise eingeben, erstellt MemoryDB `my-exported-snapshot.my-exported-snapshot-0001.rdb`

- `--target-bucket`— Name des Amazon S3 S3-Buckets, in den Sie den Snapshot exportieren möchten. Eine Kopie des Snapshots wird im angegebenen Bucket erstellt.

Damit der Exportvorgang erfolgreich ist, `--target-bucket` muss es sich um einen Amazon S3 S3-Bucket in der AWS Region des Snapshots mit den folgenden Berechtigungen handeln.

- Objektzugriff – Lesen und Schreiben.
- Berechtigungszugriff – Lesen.

Weitere Informationen finden Sie unter [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#).

Mit dem folgenden Vorgang wird ein Snapshot nach `my-s3-bucket` kopiert.

Für Linux, macOS oder 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
```

Für 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

Wenn Ihr S3-Bucket nicht über die erforderlichen Berechtigungen verfügt, damit MemoryDB einen Snapshot dorthin exportieren kann, erhalten Sie eine der folgenden Fehlermeldungen. Kehren Sie zurück zu [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#), um die angegebenen Berechtigungen hinzuzufügen, und versuchen Sie erneut, Ihren Snapshot zu exportieren.

- MemoryDB wurden keine LESEberechtigungen %s für den S3-Bucket erteilt.
Lösung: Fügen Sie Leseberechtigungen für den Bucket hinzu.
- MemoryDB wurden keine Schreibberechtigungen %s für den S3-Bucket erteilt.
Lösung: Fügen Sie Schreibberechtigungen für den Bucket hinzu.
- MemoryDB wurden keine READ_ACP-Berechtigungen %s für den S3-Bucket erteilt.
Lösung: Fügen Sie Read-Zugriff für Berechtigungen für den Bucket hinzu.

Weitere Informationen finden Sie unter `copy-snapshot` in der Referenz zum AWS CLI -Befehl.

Wenn Sie Ihren Snapshot in eine andere AWS Region kopieren möchten, verwenden Sie Amazon S3 Copy. Weitere Informationen finden Sie unter [Objekte kopieren](#) im Amazon Simple Storage Service-Benutzerhandbuch.

Exportieren eines MemoryDB-Snapshots (MemoryDB-API)

Exportieren Sie den Snapshot mithilfe der CopySnapshot API-Operation mit diesen Parametern in einen Amazon S3 S3-Bucket.

Parameter

- `SourceSnapshotName`— Name des zu kopierenden Snapshots.
- `TargetSnapshotName`— Name der Kopie des Snapshots.

Der Name muss zwischen 1 und 1.000 Zeichen lang sein und UTF-8-codierbar sein.

MemoryDB fügt dem Wert, den Sie hier eingeben, `.rdb`, eine Shard-ID und hinzu. Wenn Sie z. B. `my-exported-snapshot` eingeben, erhalten Sie `my-exported-snapshot-0001.rdb`.

- **TargetBucket**— Name des Amazon S3 S3-Buckets, in den Sie den Snapshot exportieren möchten. Eine Kopie des Snapshots wird im angegebenen Bucket erstellt.

Damit der Exportvorgang erfolgreich ist, **TargetBucket** muss es sich um einen Amazon S3 S3-Bucket in der AWS Region des Snapshots mit den folgenden Berechtigungen handeln.

- Objektzugriff – Lesen und Schreiben.
- Berechtigungszugriff – Lesen.

Weitere Informationen finden Sie unter [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#).

Im folgenden Beispiel wird eine Kopie eines automatischen Snapshots in den Amazon S3 S3-Bucket `my-s3-bucket` erstellt.

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

Wenn Ihr S3-Bucket nicht über die erforderlichen Berechtigungen verfügt, damit MemoryDB einen Snapshot dorthin exportieren kann, erhalten Sie eine der folgenden Fehlermeldungen. Kehren Sie zurück zu [Schritt 2: Gewähren Sie MemoryDB-Zugriff auf Ihren Amazon S3 S3-Bucket](#), um die angegebenen Berechtigungen hinzuzufügen, und versuchen Sie erneut, Ihren Snapshot zu exportieren.

- MemoryDB wurden keine LESEberechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Leseberechtigungen für den Bucket hinzu.

- MemoryDB wurden keine Schreibberechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Schreibberechtigungen für den Bucket hinzu.

- MemoryDB wurden keine READ_ACP-Berechtigungen %s für den S3-Bucket erteilt.

Lösung: Fügen Sie Read-Zugriff für Berechtigungen für den Bucket hinzu.

Weitere Informationen finden Sie unter [CopySnapshot](#)

Wenn Sie Ihren Snapshot in eine andere AWS Region kopieren möchten, verwenden Sie Amazon S3 Copy, um den exportierten Snapshot in den Amazon S3 S3-Bucket in einer anderen AWS Region zu kopieren. Weitere Informationen finden Sie unter [Objekte kopieren](#) im Amazon Simple Storage Service-Benutzerhandbuch.

Wiederherstellung aus einem Snapshot

Sie können die Daten jederzeit aus einer MemoryDB- oder ElastiCache für Redis .rdb-Snapshot-Datei in einem neuen Cluster wiederherstellen.

Der Wiederherstellungsprozess von MemoryDB for Redis unterstützt Folgendes:

- Migration von einer oder mehreren .rdb-Snapshot-Dateien, die Sie ElastiCache für Redis erstellt haben, zu einem MemoryDB-Cluster.

Die .rdb-Dateien müssen für die Wiederherstellung in S3 verschoben werden.

- Angabe einer Anzahl von Shards im neuen Cluster, die sich von der Anzahl der Shards im Cluster unterscheidet, der zur Erstellung der Snapshot-Datei verwendet wurde.
- Angeben eines anderen Knotentyps für den neuen Cluster größer oder kleiner. Vergewissern Sie sich beim Skalieren zu einem kleineren Knotentyp davon, dass der Speicher des neuen Knotentyps für Ihre Daten und den Redis-Overhead ausreicht.
- Die Steckplätze des neuen MemoryDB-Clusters werden anders konfiguriert als in dem Cluster, der zur Erstellung der Snapshot-Datei verwendet wurde.

Important

- MemoryDB-Cluster unterstützen nicht mehrere Datenbanken. Daher schlägt Ihre Wiederherstellung bei der Wiederherstellung in MemoryDB fehl, wenn die RDB-Datei auf mehr als eine Datenbank verweist.
- Sie können einen Snapshot nicht aus einem Cluster wiederherstellen, der Data Tiering verwendet (z. B. den Knotentyp R6GD), in einen Cluster, der kein Daten-Tiering verwendet (z. B. den R6G-Knotentyp).

Ob Sie beim Wiederherstellen eines Clusters aus einem Snapshot Änderungen vornehmen, hängt von Ihren Entscheidungen ab. Sie treffen diese Optionen auf der Seite „Cluster wiederherstellen“, wenn Sie die MemoryDB-Konsole für die Wiederherstellung verwenden. Sie treffen diese Auswahl, indem Sie Parameterwerte festlegen, wenn Sie die MemoryDB-API AWS CLI oder die MemoryDB-API für die Wiederherstellung verwenden.

Während des Wiederherstellungsvorgangs erstellt MemoryDB den neuen Cluster und füllt ihn dann mit Daten aus der Snapshot-Datei. Wenn dieser Vorgang abgeschlossen ist, ist der Cluster aufgewärmt und bereit, Anfragen anzunehmen.

 **Important**

Bevor Sie fortfahren, stellen Sie sicher, dass Sie einen Snapshot des Clusters erstellt haben, von dem Sie eine Wiederherstellung durchführen möchten. Weitere Informationen finden Sie unter [Manuelle Snapshots erstellen](#).

Informationen zur Wiederherstellung aus einem extern erstellten Snapshot finden Sie unter [Einen neuen Cluster mit einem extern erstellten Snapshot erstellen](#).

Die folgenden Verfahren zeigen Ihnen, wie Sie mithilfe der MemoryDB-Konsole, der oder der MemoryDB-API einen Snapshot in einem neuen Cluster wiederherstellen. AWS CLI

Wiederherstellung aus einem Snapshot (Konsole)

Um einen Snapshot auf einem neuen Cluster (Konsole) wiederherzustellen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`](#).
2. Wählen Sie im Navigationsbereich Snapshots aus.
3. Wählen Sie in der Liste der Snapshots die Schaltfläche neben dem Namen des Snapshots aus, von dem Sie eine Wiederherstellung durchführen möchten.
4. Wählen Sie „Aktionen“ und anschließend „Wiederherstellen“
5. Geben Sie unter Clusterkonfiguration Folgendes ein:
 - a. Clustername — Erforderlich. Der Name des neuen Clusters.
 - b. Beschreibung — Optional. Die Beschreibung des neuen Clusters.
6. Füllen Sie den Abschnitt Subnetzgruppen aus:
 - Erstellen Sie für Subnetzgruppen eine neue Subnetzgruppe oder wählen Sie eine vorhandene aus der verfügbaren Liste aus, die Sie auf diesen Cluster anwenden möchten. Wenn Sie eine neue erstellen:
 - Geben Sie einen Namen ein
 - Geben Sie eine Beschreibung ein

- Wenn Sie Multi-AZ aktiviert haben, muss die Subnetzgruppe mindestens zwei Subnetze enthalten, die sich in verschiedenen Availability Zones befinden. Weitere Informationen finden Sie unter [Subnetze und Subnetzgruppen](#).
- Wenn Sie eine neue Subnetzgruppe erstellen und noch keine VPC haben, werden Sie aufgefordert, eine VPC zu erstellen. Weitere Informationen finden Sie unter [Was ist Amazon VPC?](#) im Amazon VPC-Benutzerhandbuch.

7. Füllen Sie den Abschnitt Clustereinstellungen aus:

- a. Akzeptieren Sie aus Gründen der Redis-Versionskompatibilität die Standardeinstellung 6.0.
- b. Akzeptieren Sie für Port den Redis-Standardport 6379 oder geben Sie die Portnummer ein, falls Sie einen anderen Port verwenden möchten.
- c. Akzeptieren Sie für Parametergruppe die `default.memorydb-redis6` Parametergruppe.

Parametergruppen steuern die Laufzeitparameter Ihres Clusters. Weitere Informationen zu Parametergruppen finden Sie unter [Redis-spezifische Parameter](#).

- d. Wählen Sie unter Knotentyp einen Wert für den gewünschten Knotentyp (zusammen mit der zugehörigen Speichergröße) aus.

Wenn Sie ein Mitglied der R6gd-Knotentypfamilie wählen, aktivieren Sie automatisch das Data-Tiering in Ihrem Cluster. Weitere Informationen finden Sie unter [Daten-Tiering](#).

- e. Wählen Sie unter Anzahl der Shards die Anzahl der Shards aus, die Sie für diesen Cluster benötigen.

Sie können die Anzahl der Shards in Ihrem Cluster dynamisch ändern. Weitere Informationen finden Sie unter [Skalieren von MemoryDB-Clustern](#).

- f. Wählen Sie für Replicas per shard (Replikate pro Shard) die Anzahl der Read Replica-Knoten aus, die sich in jedem Shard befinden sollen.

Es bestehen die folgenden Einschränkungen;

- Wenn Sie Multi-AZ aktiviert haben, stellen Sie sicher, dass mindestens ein Replikat pro Shard vorhanden ist.
- Die Anzahl der Replikate ist für jeden Shard gleich, wenn der Cluster mithilfe der Konsole erstellt wird.

- g. Wählen Sie Weiter

- h. Füllen Sie den Abschnitt Erweiterte Einstellungen aus:

- i. Wählen Sie für Security groups (Sicherheitsgruppen) die gewünschten Sicherheitsgruppen für diesen Cluster aus. Eine security group (Sicherheitsgruppe) fungiert als Firewall, um den Netzwerkzugriff auf Ihren Cluster zu steuern. Sie können die Standardsicherheitsgruppe für Ihre VPC verwenden oder eine neue erstellen.

Weitere Informationen zu Sicherheitsgruppen finden Sie unter [Sicherheitsgruppen für Ihre VPC](#) im Benutzerhandbuch zu Amazon VPC.

- ii. Daten werden auf folgende Weise verschlüsselt:
 - Verschlüsselung im Ruhezustand – Ermöglicht die Verschlüsselung von Daten, die auf der Festplatte gespeichert sind. Weitere Informationen finden Sie unter [Verschlüsselung im Ruhezustand](#).

 Note

Sie haben die Möglichkeit, einen anderen Verschlüsselungsschlüssel anzugeben, indem Sie Customer Managed AWS KMS key und dann den Schlüssel auswählen.

- Verschlüsselung während der Übertragung – Ermöglicht die Verschlüsselung von Daten während der Übertragung. Dies ist standardmäßig aktiviert. Weitere Informationen finden Sie unter [Verschlüsselung während der Übertragung](#).

Wenn Sie keine Verschlüsselung auswählen, wird eine offene Zugriffskontrollliste namens „Open Access“ mit einem Standardbenutzer erstellt. Weitere Informationen finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).

- iii. Geben Sie für Snapshot optional einen Aufbewahrungszeitraum für Snapshots und ein Snapshot-Fenster an. Standardmäßig ist die Option Automatische Snapshots aktivieren ausgewählt.
- iv. Geben Sie für das Wartungsfenster optional ein Wartungsfenster an. Das Wartungsfenster ist die Zeit, in der Regel eine Stunde, jede Woche, zu der MemoryDB die Systemwartung für Ihren Cluster plant. Sie können MemoryDB erlauben, den Tag und die Uhrzeit für Ihr Wartungsfenster auszuwählen (keine Präferenz), oder Sie können Tag, Uhrzeit und Dauer selbst wählen (Wartungsfenster angeben). Treffen Sie bei Wahl von Specify maintenance window eine Auswahl in den Listen Start day,

Start time und Duration (in Stunden) für Ihr Wartungsfenster. Alle Uhrzeiten sind in UCT angegeben.

Weitere Informationen finden Sie unter [Verwaltung der Wartung](#).

- v. Wählen Sie für Benachrichtigungen ein bestehendes Amazon Simple Notification Service (Amazon SNS)-Thema oder wählen Sie Manuelle ARN-Eingabe und geben Sie den Amazon-Ressourcennamen (ARN) des Themas ein. Mit Amazon SNS können Sie Push-Benachrichtigungen an mit dem Internet verbundene Smart-Geräte senden. Standardmäßig sind Benachrichtigungen deaktiviert. [Weitere Informationen finden Sie unter <https://aws.amazon.com/sns/>](#).
- i. Bei Tags können Sie optional Tags anwenden, um Ihre Cluster zu durchsuchen und zu filtern oder Ihre AWS Kosten nachzuverfolgen.
- j. Überprüfen Sie alle Ihre Einträge und ausgewählten Optionen und machen Sie dann evtl. erforderliche Korrekturen. Wenn Sie bereit sind, wählen Sie Create cluster, um den Cluster zu starten, oder Cancel, um die Operation abubrechen.

Sobald als Status des Clusters available erscheint, können Sie EC2 Zugriff darauf erteilen, eine Verbindung mit ihm herstellen und ihn verwenden. Weitere Informationen finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#) und [Schritt 3: Connect zum Cluster herstellen](#).

Important

Sobald Ihr Cluster verfügbar ist, wird Ihnen jede ganze oder angebrochene Stunde in Rechnung gestellt, die der Cluster aktiv ist, auch wenn Sie ihn nicht aktiv nutzen. Damit Ihnen keine Kosten mehr für diesen Cluster entstehen, müssen Sie ihn löschen. Siehe [Schritt 4: Löschen eines Clusters](#).

Wiederherstellung aus einem Snapshot (AWS CLI)

Wenn Sie einen der beiden `create-cluster` Operationen verwenden, stellen Sie sicher, dass Sie den Parameter angeben `--snapshot-name` oder `--snapshot-arns` den neuen Cluster mit den Daten aus dem Snapshot als Seed versehen.

Weitere Informationen finden Sie hier:

- [Einen Cluster erstellen \(AWS CLI\)](#) im MemoryDB-Benutzerhandbuch.

- [create-cluster](#) in der Befehlsreferenz. AWS CLI

Wiederherstellung aus einem Snapshot (MemoryDB-API)

Sie können einen MemoryDB-Snapshot mithilfe des MemoryDB-API-Vorgangs wiederherstellen.

CreateCluster

Achten Sie bei der Verwendung des CreateCluster Vorgangs darauf, den Parameter anzugeben SnapshotName oder den neuen Cluster mit den Daten aus dem Snapshot SnapshotArns zu versorgen.

Weitere Informationen finden Sie hier:

- [Einen Cluster erstellen \(MemoryDB-API\)](#) im MemoryDB-Benutzerhandbuch.
- [CreateCluster](#) in der MemoryDB-API-Referenz.

Einen neuen Cluster mit einem extern erstellten Snapshot erstellen

Wenn Sie einen neuen MemoryDB-Cluster erstellen, können Sie ihn mit Daten aus einer Redis-.rdb-Snapshot-Datei versorgen.

Informationen zum Seeding eines neuen MemoryDB-Clusters aus einem MemoryDB-Snapshot oder einem Redis-Snapshot finden Sie unter [ElastiCache Wiederherstellung aus einem Snapshot](#)

Wenn Sie eine Redis-.rdb-Datei zum Seeding eines neuen MemoryDB-Clusters verwenden, können Sie Folgendes tun:

- Geben Sie eine Anzahl von Shards im neuen Cluster an. Diese Zahl kann sich von der Anzahl der Shards im Cluster unterscheiden, der zur Erstellung der Snapshot-Datei verwendet wurde.
- Geben Sie einen anderen Knotentyp für den neuen Cluster an — größer oder kleiner als der Knotentyp, der in dem Cluster verwendet wurde, der den Snapshot erstellt hat. Wenn Sie zu einem kleineren Knotentyps skalieren, stellen Sie sicher, dass der Speicher des neuen Knotentyps für Ihre Daten und den Redis-Overhead ausreicht.

Important

- Sie müssen sicherstellen, dass Ihre Snapshot-Daten die Ressourcen des Knotens nicht überschreiten.

Wenn der Snapshot zu groß ist, hat der resultierende Cluster den Status `restore-failed`. In diesem Fall müssen Sie den Cluster löschen und von Neuem beginnen.

Eine vollständige Liste der Knotentypen und Spezifikationen finden Sie unter [MemoryDB-Knotentyp-spezifische Parameter](#).

- Sie können eine .rdb-Redis-Datei nur mit serverseitiger Amazon-S3-Verschlüsselung (SSE-S3) verschlüsseln. Weitere Informationen finden Sie unter [Schützen von Daten mithilfe serverseitiger Verschlüsselung](#).

Schritt 1: Erstellen Sie einen Redis-Snapshot auf einem externen Cluster

Um den Snapshot für das Seeding Ihres MemoryDB-Clusters zu erstellen

1. Stellen Sie eine Verbindung zu Ihrer vorhandenen Redis-Instance her.

2. Führen Sie entweder den Redis BGSAVE - oder den SAVE Vorgang aus, um einen Snapshot zu erstellen. Achten Sie auf den Speicherort der .rdb-Datei.

BGSAVE ist asynchron und blockiert während der Verarbeitung keine anderen Clients. Weitere Informationen finden Sie unter [BGSAVE](#) auf der Redis-Website.

SAVE ist synchron und blockiert andere Vorgänge, bis sie Verarbeitung abgeschlossen ist. Weitere Informationen finden Sie unter [SAVE](#) auf der Redis-Website.

Weitere Informationen zum Erstellen eines Snapshots finden Sie unter [Redis-Persistenz](#) auf der Redis-Website.

Schritt 2: Erstellen eines Amazon-S3-Buckets und -Ordners

Wenn Sie die Snapshot-Datei erstellt haben, müssen Sie sie in einen Ordner innerhalb eines Amazon S3 S3-Buckets hochladen. Hierzu müssen bereits ein Amazon-S3-Bucket und ein Ordner innerhalb dieses Buckets vorhanden sein. Wenn Sie bereits einen Amazon-S3-Bucket und Ordner mit den entsprechenden Berechtigungen besitzen, können Sie mit [Schritt 3: Laden Sie Ihren Snapshot auf Amazon S3 hoch](#) fortfahren.

So erstellen Sie einen Amazon-S3-Bucket

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.
2. Führen Sie die Anweisungen zum Erstellen eines Amazon S3 Buckets unter [Erstellen eines Buckets](#) im Benutzerhandbuch zu Amazon Simple Storage Service aus.

Der Name Ihres Amazon-S3-Buckets muss DNS-konform sein. Andernfalls kann MemoryDB nicht auf Ihre Backup-Datei zugreifen. Die Regeln für die DNS-Konformität lauten:

- Bucket-Namen müssen mindestens 3 und dürfen höchstens 63 Zeichen umfassen.
- Die Namen müssen eine Folge aus einer oder mehreren Beschriftungen darstellen, die durch einen Punkt (.) getrennt sind, wobei jede Beschriftung:
 - mit einem Kleinbuchstaben oder einer Zahl beginnen.
 - mit einem Kleinbuchstaben oder einer Zahl beginnen.
 - Enthält nur Kleinbuchstaben, Zahlen und Bindestriche.
- Er darf nicht als IP-Adresse (z. B. 192.0.2.0) formatiert sein.

Wir empfehlen dringend, dass Sie Ihren Amazon S3 S3-Bucket in derselben AWS Region wie Ihr neuer MemoryDB-Cluster erstellen. Dieser Ansatz stellt sicher, dass die höchste Datenübertragungsgeschwindigkeit erreicht wird, wenn MemoryDB Ihre RDB-Datei aus Amazon S3 liest.

 Note

Um Ihre Daten so sicher wie möglich zu halten, sollten Sie die Berechtigungen für Ihr Amazon-S3-Bucket so restriktiv wie möglich gestalten. Gleichzeitig müssen die Berechtigungen weiterhin zulassen, dass der Bucket und sein Inhalt für das Seeding Ihres neuen MemoryDB-Clusters verwendet werden können.

So fügen Sie einem Amazon S3 Bucket einen Ordner hinzu

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie den Namen des Buckets aus, in den die .rdb-Datei hochgeladen werden soll.
3. Wählen Sie Create folder.
4. Geben Sie einen Namen für den neuen Ordner ein.
5. Wählen Sie Speichern.

Notieren Sie sich sowohl den Bucketnamen als auch den Ordernamen.

Schritt 3: Laden Sie Ihren Snapshot auf Amazon S3 hoch

Laden Sie jetzt die .rdb-Datei hoch, die Sie in [Schritt 1: Erstellen Sie einen Redis-Snapshot auf einem externen Cluster](#) erstellt haben. Sie laden sie in den Amazon-S3-Bucket und -Ordner hoch, die sie in [Schritt 2: Erstellen eines Amazon-S3-Buckets und -Ordners](#) erstellt haben. Weitere Informationen zu dieser Aufgabe finden Sie unter [Objekte hochladen](#). Wählen Sie zwischen den Schritten 2 und 3 den Namen des von Ihnen erstellten Ordners aus.

So laden Sie Ihre .rdb-Datei in einen Amazon-S3-Ordner hoch

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.

2. Wählen Sie den Namen des Amazon-S3-Bucket aus, den Sie in Schritt 2 erstellt haben.
3. Wählen Sie den Namen des Ordners aus, den Sie in Schritt 2 erstellt haben.
4. Klicken Sie auf Hochladen.
5. Klicken Sie auf Add files.
6. Navigieren Sie zu der Datei oder den Dateien, die Sie hochladen möchten, und wählen Sie dann die Datei oder die Dateien aus. Halten Sie zur Auswahl mehrerer Dateien die Strg-Taste während der Auswahl der Dateinamen gedrückt.
7. Klicken Sie auf Open.
8. Vergewissern Sie sich, dass die richtige (n) Datei (en) auf der Upload-Seite aufgeführt sind, und wählen Sie dann Upload.

Notieren Sie den Pfad zu Ihrer .rdb-Datei. Wenn der Bucket-Name z. B. myBucket und der Pfad myFolder/redis.rdb lautet, geben Sie myBucket/myFolder/redis.rdb ein. Sie benötigen diesen Pfad, um den neuen Cluster mit den Daten in diesem Snapshot zu versorgen.

Weitere Informationen finden Sie unter [Regeln zur Benennung von Buckets](#) im Amazon Simple Storage Service-Benutzerhandbuch.

Schritt 4: Gewähren Sie MemoryDB Lesezugriff auf die .rdb-Datei

AWS Regionen, die vor dem 20. März 2019 eingeführt wurden, sind standardmäßig aktiviert. Sie können sofort mit der Arbeit in diesen AWS Regionen beginnen. Regionen, die nach dem 20. März 2019 eingeführt wurden, sind standardmäßig deaktiviert. Sie müssen diese Regionen aktivieren oder sich für sie anmelden, bevor Sie sie verwenden können, wie unter [AWS Regionen verwalten](#) beschrieben.

Gewähren Sie MemoryDB Lesezugriff auf die RDB-Datei

Um MemoryDB Lesezugriff auf die Snapshot-Datei zu gewähren

1. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie den Namen des S3-Buckets aus, in dem sich Ihre .rdb-Datei befindet.
3. Wählen Sie den Namen des Ordners aus, in dem sich Ihre .rdb-Datei befindet.
4. Wählen Sie den Namen Ihrer .rdb-Snapshot-Datei. Der Name der ausgewählten Datei erscheint oberhalb der Registerkarten oben auf der Seite.

5. Wählen Sie die Registerkarte Berechtigungen.
6. Wählen Sie unter Permissions (Berechtigungen) Bucket policy (Bucket-Richtlinie) aus und klicken Sie dann auf Edit (Bearbeiten).
7. Aktualisieren Sie die Richtlinie, um MemoryDB die erforderlichen Berechtigungen zur Ausführung von Vorgängen zu gewähren:
 - Fügen Sie ["Service" : "*region-full-name*.memorydb-snapshot.amazonaws.com"] zu Principal hinzu.
 - Fügen Sie die folgenden Berechtigungen hinzu, die für das Exportieren eines Snapshots in den Amazon-S3-Bucket erforderlich sind:
 - "s3:GetObject"
 - "s3:ListBucket"
 - "s3:GetBucketAcl"

Nachfolgend finden Sie ein Beispiel dafür, wie die aktualisierte Richtlinie aussehen könnte.

```
{
  "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. Wählen Sie Speichern.

Schritt 5: Den MemoryDB-Cluster mit den.rdb-Dateidaten versorgen

Jetzt sind Sie bereit, einen MemoryDB-Cluster zu erstellen und ihn mit den Daten aus der RDB-Datei zu versorgen. Folgen Sie den Anweisungen unter, um den Cluster zu erstellen. [Einen MemoryDB-Cluster erstellen](#)

Die Methode, mit der Sie MemoryDB mitteilen, wo sich der Redis-Snapshot befindet, den Sie auf Amazon S3 hochgeladen haben, hängt von der Methode ab, mit der Sie den Cluster erstellen:

Füllen Sie den MemoryDB-Cluster mit den Daten der .rdb-Datei

- Verwenden der MemoryDB-Konsole

Erweitern Sie nach Auswahl der Redis-Engine den Bereich Advanced Redis settings und suchen Sie Import data to cluster. Geben Sie im Feld Seed-RDB-Datei-S3-Speicherort den Amazon-S3-Pfad für die Datei(en) ein. Wenn Sie mehrere .rdb-Dateien besitzen, geben Sie den Pfad für jede Datei in eine durch Kommas getrennten Liste ein. Der Amazon-S3-Pfad sieht etwa aus wie *myBucket/myFolder/myBackupFilename.rdb*.

- Mit dem AWS CLI

Verwenden Sie bei Einsatz der Operation `create-cluster` oder `create-cluster` den Parameter `--snapshot-arns`, um einen vollqualifizierten ARN für jede .rdb-Datei anzugeben. z. B. *arn:aws:s3:::myBucket/myFolder/myBackupFilename.rdb*. Der ARN muss in die Snapshot-Dateien aufgelöst werden, die Sie in Amazon S3 gespeichert haben.

- Verwenden der MemoryDB-API

Wenn Sie den `CreateCluster` oder den `CreateCluster` MemoryDB-API-Vorgang verwenden, verwenden Sie den Parameter, `SnapshotArns` um einen vollqualifizierten ARN für jede .rdb-Datei anzugeben. z. B. *arn:aws:s3:::myBucket/myFolder/myBackupFilename.rdb*. Der ARN muss in die Snapshot-Dateien aufgelöst werden, die Sie in Amazon S3 gespeichert haben.

Während der Erstellung Ihres Clusters werden die Daten in Ihrem Snapshot in den Cluster geschrieben. Sie können den Fortschritt überwachen, indem Sie sich die MemoryDB-Ereignismeldungen ansehen. Rufen Sie dazu die MemoryDB-Konsole auf und wählen Sie Ereignisse. Sie können auch die AWS MemoryDB-Befehlszeilenschnittstelle oder die MemoryDB-API verwenden, um Ereignismeldungen abzurufen.

Schnappschüsse taggen

Sie können jedem Snapshot Ihre eigenen Metadaten in Form von Tags zuweisen. Mithilfe von Tags können Sie Ihre Snapshots auf unterschiedliche Weise kategorisieren, z. B. nach Zweck, Eigentümer oder Umgebung. Dies ist nützlich, wenn Sie viele Ressourcen desselben Typs haben — In diesem Fall können Sie schnell bestimmte Ressourcen basierend auf den zugewiesenen Tags (Markierungen) bestimmen. Weitere Informationen finden Sie unter [Ressourcen, die markiert werden können](#).

Mithilfe von Tags zur Kostenzuweisung können Sie Ihre Kosten für mehrere AWS Dienste verfolgen, indem Sie Ihre Ausgaben auf Rechnungen nach Tagwerten gruppieren. Weitere Informationen zu Kostenzuordnungs-Tags finden Sie unter [Verwenden von Kostenzuordnungs-Tags](#).

Mithilfe der MemoryDB-Konsole AWS CLI, der oder der MemoryDB-API können Sie Kostenzuweisungs-Tags zu Ihren Snapshots hinzufügen, auflisten, ändern, entfernen oder kopieren. Weitere Informationen finden Sie unter [Überwachung von Kosten mit Kostenzuordnungs-Tags](#).

Löschen eines Snapshots

Ein automatischer Snapshot wird automatisch gelöscht, wenn sein Aufbewahrungslimit abläuft. Wenn Sie einen Cluster löschen, werden auch alle seine automatischen Snapshots gelöscht.

MemoryDB bietet einen API-Löschvorgang, mit dem Sie einen Snapshot jederzeit löschen können, unabhängig davon, ob der Snapshot automatisch oder manuell erstellt wurde. Da manuelle Snapshots kein Aufbewahrungslimit haben, ist manuelles Löschen die einzige Möglichkeit, sie zu entfernen.

Sie können einen Snapshot mit der MemoryDB-Konsole, der oder der AWS CLI MemoryDB-API löschen.

Löschen eines Snapshots (Konsole)

Das folgende Verfahren löscht einen Snapshot mithilfe der MemoryDB-Konsole.

So löschen Sie einen Snapshot

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Snapshots aus.

Der Bildschirm „Schnapschüsse“ wird mit einer Liste Ihrer Schnapschüsse angezeigt.

3. Wählen Sie das Optionsfeld links neben dem Namen des Snapshots, den Sie löschen möchten.
4. Wählen Sie `Actions` und dann `Delete` aus.
5. Wenn Sie diesen Snapshot löschen möchten, geben Sie ihn `delete` in das Textfeld ein und wählen Sie dann `Löschen`. Um den Löschvorgang abzubrechen, wählen Sie `Abbrechen`. Der Status wird in `deleting` geändert.

Löschen eines Snapshots (AWS CLI)

Verwenden Sie den AWS CLI Vorgang `delete-snapshot` mit dem folgenden Parameter, um einen Snapshot zu löschen.

- `--snapshot-name`— Name des zu löschenden Snapshots.

Der folgende Code löscht den Snapshot `myBackup`.

```
aws memorydb delete-snapshot --snapshot-name myBackup
```

Weitere Informationen finden Sie unter [delete-snapshot](#) in der AWS CLI -CLI-Befehlsreferenz.

Löschen eines Snapshots (MemoryDB-API)

Verwenden Sie den DeleteSnapshot API-Vorgang mit dem folgenden Parameter, um einen Snapshot zu löschen.

- SnapshotName— Name des zu löschenden Snapshots.

Der folgende Code löscht den 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>
```

Weitere Informationen finden Sie unter [DeleteSnapshot](#).

Skalierung

Die Datenmenge, die von einer Anwendung verarbeitet wird, ist selten statisch. Sie steigt und sinkt mit dem Unternehmenswachstum und unterliegt normalen Schwankungen im Bedarf. Wenn Sie Ihre Anwendungen selbst verwalten, müssen Sie ausreichend Hardware für Ihre Bedarfsspitzen bereitstellen, was teuer sein kann. Durch die Verwendung von MemoryDB for Redis können Sie die Skalierung an den aktuellen Bedarf anpassen und zahlen nur für verwendete Ressourcen.

Im Folgenden finden Sie das richtige Thema für die Skalierungsaktionen, die Sie ausführen möchten.

Skalierung von MemoryDB

Action	SpeicherDB
Ausskalieren	Online-Resharding und Shard-Rebalancing für MemoryDB

Action	SpeicherDB	
Ändern von Knotentypen	Vertikales Online-Skalieren durch Ändern des Knotentyps	
Ändern der Anzahl der Shards	Skalieren von MemoryDB-Clustern	

Skalieren von MemoryDB-Clustern

Wenn sich die Anforderungen an Ihre Cluster ändern, können Sie beschließen, die Leistung zu verbessern oder die Preise zu senken, indem Sie die Anzahl der Shards in Ihrem MemoryDB-Cluster ändern. Wir empfehlen, dazu die horizontale Online-Skalierung zu verwenden, da Ihr Cluster während des Skalierungsprozesses weiterhin Anforderungen bedienen kann.

Zu den Bedingungen, unter denen Sie den Cluster möglicherweise neu skalieren, gehören folgende:

- Speicherbelastung

Wenn die Knoten in Ihrem Cluster einer hohen Speicherbelastung ausgesetzt sind, können Sie sich für eine Skalierung entscheiden, sodass Sie über mehr Ressourcen verfügen, um Daten besser speichern und Anforderungen verarbeiten zu können.

Sie können feststellen, ob Ihre Knoten unter Speicherauslastung stehen, indem Sie die folgenden Metriken überwachen: `FreeableMemorySwapUsage`, und `BytesUsedForMemoryDB`.

- CPU- oder Netzwerkengpässe:

Wenn Latenz-/Durchsatzprobleme Ihren Cluster belasten, müssen Sie möglicherweise zur Lösung der Probleme eine Skalierung durchführen.

Sie können Ihre Latenz und Ihren Durchsatz überwachen, indem Sie die folgenden Metriken überwachen: `CPUUtilization`, `NetworkBytesIn`, `NetworkBytesOut`, `CurrConnections`, und `NewConnections`.

- Ihr Cluster ist übermäßig skaliert:

Der aktuelle Bedarf an Ihrem Cluster ist so hoch, dass eine Skalierung die Leistung nicht beeinträchtigt und Ihre Kosten reduziert.

Sie können die Nutzung Ihres Clusters überwachen, um anhand der folgenden Metriken festzustellen, ob Sie sicher skalieren können oder nicht: `FreeableMemorySwapUsage`, `BytesUsedForMemoryDB`, `CPUUtilization`, `NetworkBytesIn`, `NetworkBytesOut`, `CurrConnections`, und `NewConnections`.

Leistungsbeeinträchtigung durch Skalierung

Wenn Sie den Offline-Prozess skalieren, ist Ihr Cluster für einen erheblichen Teil des Prozesses offline und kann daher keine Anforderungen erfüllen. Wenn Sie mithilfe der Onlinemethode

skalieren, da die Skalierung eine rechenintensive Operation ist, gibt es eine Leistungseinbuße. Ihr Cluster führt jedoch weiterhin während der Skalierungsoperation Anforderungen aus. Wie stark die Verschlechterung ist, hängt von Ihrer normalen CPU-Auslastung und Ihren Daten ab.

Es gibt zwei Möglichkeiten, Ihren MemoryDB-Cluster zu skalieren: horizontale und vertikale Skalierung.

- Durch die horizontale Skalierung können Sie die Anzahl der Shards im Cluster ändern, indem Sie Shards hinzufügen oder entfernen. Online-Resharding ermöglicht Ein- und Ausskalieren, während der Cluster weiterhin alle eingehenden Anforderungen bedient.
- Vertikale Skalierung – Ändern Sie den Knotentyp, um die Größe des Clusters anzupassen. Vertikale Online-Skalierung ermöglicht Aufwärts- und Abwärtsskalieren, während der Cluster weiterhin alle eingehenden Anforderungen bedient.

Wenn Sie die Größe der Speicherkapazität des Clusters durch einskalieren oder herunterskalieren verkleinern, stellen Sie sicher, dass die neue Konfiguration über ausreichend Speicher für Ihre Daten und den Redis-Overhead verfügt.

Offline-Resharding und Shard-Rebalancing für MemoryDB

Der Hauptvorteil, den Sie durch die Offline-Shard-Rekonfiguration erhalten, besteht darin, dass Sie mehr tun können, als nur Shards zu Ihrem Cluster hinzuzufügen oder zu entfernen. Wenn Sie das Reshards offline durchführen, können Sie nicht nur die Anzahl der Shards in Ihrem Cluster ändern, sondern auch Folgendes tun:

- Ändern Sie den Knotentyp Ihres Clusters.
- Upgrade auf eine neuere Engine-Version.

Note

Das Offline-Resharding wird auf Clustern mit aktiviertem Daten-Tiering nicht unterstützt. Weitere Informationen finden Sie unter [Daten-Tiering](#)..

Der Hauptnachteil der Offline-Shard-Neukonfiguration besteht darin, dass Ihr Cluster offline ist und mit dem Wiederherstellungsabschnitt des Prozesses beginnt und so lange fortfährt, bis Sie die

Endpunkte in Ihrer Anwendung aktualisieren. Die Dauer, in der Ihr Cluster offline ist, hängt von der Datenmenge in Ihrem Cluster ab.

Um deinen Shards MemoryDB-Cluster offline neu zu konfigurieren

1. Erstellen Sie einen manuellen Snapshot des bestehenden MemoryDB-Clusters. Weitere Informationen finden Sie unter [Manuelle Snapshots erstellen](#).
2. Erstellen Sie einen neuen Cluster, indem Sie ihn aus dem Snapshot wiederherstellen. Weitere Informationen finden Sie unter [Wiederherstellung aus einem Snapshot](#).
3. Aktualisieren Sie in Ihrer Anwendung die Endpunkte auf die neuen Cluster-Endpunkte. Weitere Informationen finden Sie unter [Verbindungsendpunkte ermitteln](#).

Online-Resharding und Shard-Rebalancing für MemoryDB

Durch die Verwendung von Online Resharding und Shard Rebalancing mit MemoryDB können Sie Ihre MemoryDB dynamisch und ohne Ausfallzeiten skalieren. Dieser Ansatz bedeutet, dass Ihr Cluster weiterhin Anfragen bearbeiten kann, selbst wenn die Skalierung oder das Rebalancing in Bearbeitung ist.

Sie haben die folgenden Möglichkeiten:

- Scale-out — Erhöhen Sie die Lese- und Schreibkapazität, indem Sie Ihrem MemoryDB-Cluster Shards hinzufügen.

Wenn Sie Ihrem Cluster einen oder mehrere Shards hinzufügen, entspricht die Anzahl der Knoten in jedem neuen Shard der Anzahl der Knoten im kleinsten der vorhandenen Shards.

- Einkalieren — Reduzieren Sie die Lese- und Schreibkapazität und damit die Preise, indem Sie Shards aus Ihrem MemoryDB-Cluster entfernen.

Derzeit gelten die folgenden Einschränkungen für das MemoryDB-Online-Resharding:

- Es gibt Einschränkungen bei Slots oder Keyspaces und großen Elementen:

Wenn ein Schlüssel in einem Shard ein großes Element enthält, wird dieser Schlüssel beim horizontalen Skalieren oder Neuausrichten nicht in eine neue Shard migriert. Diese Funktionalität kann zu unsymmetrischen Shards führen.

Wenn einer der Schlüssel in einer Shard ein großes Element enthält (Elemente größer als 256 MB nach der Serialisierung), wird diese Shard beim Skalieren nicht gelöscht. Diese Funktionalität kann dazu führen, dass einige Shards nicht gelöscht werden.

- Beim Skalieren entspricht die Anzahl der Knoten in allen neuen Shards der Anzahl der Knoten in den vorhandenen Shards.

Weitere Informationen finden Sie unter [Bewährte Methoden: Ändern der Cluster-Größe online](#).

Sie können Ihre MemoryDB-Cluster mithilfe der AWS Management Console, der und der MemoryDB-API horizontal skalieren oder neu ausgleichen. AWS CLI

Hinzufügen von Shards mit Online-Resharding

Sie können Shards zu Ihrem MemoryDB-Cluster hinzufügen. AWS Management Console AWS CLI

Hinzufügen von Shards (Konsole)

Sie können die verwenden AWS Management Console, um einen oder mehrere Shards zu Ihrem MemoryDB-Cluster hinzuzufügen. Das folgende Verfahren beschreibt den Prozess.

1. Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.
2. Wählen Sie in der Cluster-Liste den Cluster-Namen aus, aus dem Sie einen Shard hinzufügen möchten.
3. Wählen Sie auf der Registerkarte Shards and Nodes die Option Shards hinzufügen/löschen
4. Geben Sie im Feld Neue Anzahl von Shards die gewünschte Anzahl von Shards ein.
5. Wählen Sie Bestätigen, um die Änderungen beizubehalten, oder Abbrechen, um sie zu verwerfen.

Shards hinzufügen (AWS CLI)

Der folgende Prozess beschreibt, wie Sie die Shards in Ihrem MemoryDB-Cluster neu konfigurieren können, indem Sie Shards mit der hinzufügen AWS CLI.

Verwenden Sie die folgenden Parameter mit `update-cluster`.

Parameter

- `--cluster-name` – Erforderlich. Gibt an, auf welchem Cluster (Cluster) der Shard-Rekonfigurationsvorgang ausgeführt werden soll.
- `--shard-configuration` – Erforderlich. Ermöglicht das Festlegen der Anzahl der Shards
 - `ShardCount`— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Shards anzugeben.

Example

Im folgenden Beispiel wird die Anzahl der Shards im `Clustermy-cluster` auf 2 geändert.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --shard-configuration \  
    ShardCount=2
```

Für Windows:

```
aws memorydb update-cluster ^  
  --cluster-name my-cluster ^  
  --shard-configuration ^  
    ShardCount=2
```

Es gibt die folgende JSON-Antwort zurück:

```
{  
  "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
  }
}
```

Verwenden Sie den folgenden Befehl, um die Details des aktualisierten Clusters anzuzeigen, sobald sich sein Status von Update zu Verfügbar ändert:

Für Linux, macOS oder Unix:

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

Für Windows:

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

Es wird die folgende JSON-Antwort zurückgegeben:

```
{
  "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
}
]
}

```

Weitere Informationen finden Sie unter [update-cluster](#) in der AWS CLI Befehlsreferenz.

Hinzufügen von Shards (MemoryDB API)

Sie können die MemoryDB-API verwenden, um die Shards in Ihrem MemoryDB-Cluster online neu zu konfigurieren, indem Sie die `UpdateCluster` Operation verwenden.

Verwenden Sie die folgenden Parameter mit `UpdateCluster`.

Parameter

- `ClusterName` – Erforderlich. Gibt an, auf welchem Cluster der Shard-Rekonfigurationsvorgang ausgeführt werden soll.
- `ShardConfiguration` – Erforderlich. Ermöglicht das Festlegen der Anzahl der Shards
 - `ShardCount`— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Shards anzugeben.

Weitere Informationen finden Sie unter [UpdateCluster](#).

Entfernen von Shards mit Online-Resharding

Sie können Shards aus Ihrem MemoryDB-Cluster entfernen, indem Sie die AWS Management Console, die AWS CLI, oder die MemoryDB-API.

Entfernen von Shards (Konsole)

Der folgende Prozess beschreibt, wie Sie die Shards in Ihrem MemoryDB-Cluster neu konfigurieren können, indem Sie Shards mit der AWS Management Console entfernen.

Important

Bevor Sie Shards aus Ihrem Cluster entfernen, überprüft MemoryDB, ob alle Ihre Daten in die verbleibenden Shards passen. Wenn die Daten passen, werden Shards wie gewünscht aus dem Cluster gelöscht. Wenn die Daten nicht in die verbleibenden Shards passen, wird der Prozess beendet und der Cluster behält dieselbe Shard-Konfiguration wie vor der Anforderung.

Sie können die AWS Management Console verwenden, um einen oder mehrere Shards aus Ihrem MemoryDB-Cluster zu entfernen. Sie können nicht alle Shards in einem Cluster entfernen. Stattdessen müssen Sie den Cluster löschen. Weitere Informationen finden Sie unter [Schritt 4: Löschen eines Clusters](#). Im folgenden Verfahren wird der Vorgang zum Entfernen eines oder mehrerer Shards beschrieben.

1. Melden Sie sich bei der MemoryDB for Redis-Konsole an der AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.

2. Wählen Sie in der Cluster-Liste den Cluster-Namen aus, aus dem Sie einen Shard entfernen möchten.
3. Wählen Sie auf der Registerkarte Shards and Nodes die Option Shards hinzufügen/löschen
4. Geben Sie im Feld Neue Anzahl von Shards die Anzahl der gewünschten Shards ein (mindestens 1).
5. Wählen Sie Bestätigen, um die Änderungen beizubehalten, oder Abbrechen, um sie zu verwerfen.

Entfernen von Shards (AWS CLI)

Der folgende Prozess beschreibt, wie Sie die Shards in Ihrem MemoryDB-Cluster neu konfigurieren können, indem Sie Shards mit der entfernenAWS CLI.

Important

Bevor Sie Shards aus Ihrem Cluster entfernen, überprüft MemoryDB, ob alle Ihre Daten in die verbleibenden Shards passen. Wenn die Daten passen, werden Shards wie gewünscht aus dem Cluster gelöscht und ihre Keyspaces den verbleibenden Shards zugeordnet. Wenn die Daten nicht in die verbleibenden Shards passen, wird der Prozess beendet und der Cluster behält dieselbe Shard-Konfiguration wie vor der Anfrage.

Sie können die verwendenAWS CLI, um einen oder mehrere Shards aus Ihrem MemoryDB-Cluster zu entfernen. Sie können nicht alle Shards in einem Cluster entfernen. Stattdessen müssen Sie den Cluster löschen. Weitere Informationen finden Sie unter [Schritt 4: Löschen eines Clusters](#).

Verwenden Sie die folgenden Parameter mit `update-cluster`.

Parameter

- `--cluster-name` – Erforderlich. Gibt an, auf welchem Cluster (Cluster) der Shard-Rekonfigurationsvorgang ausgeführt werden soll.
- `--shard-configuration` – Erforderlich. Ermöglicht es Ihnen, die Anzahl der Shards mithilfe der `ShardCount` Eigenschaft festzulegen:

`ShardCount`— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Shards anzugeben.

Example

Im folgenden Beispiel wird die Anzahl der Shards im `Clustermy-cluster` auf 2 geändert.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --shard-configuration \  
    ShardCount=2
```

Für Windows:

```
aws memorydb update-cluster ^  
  --cluster-name my-cluster ^  
  --shard-configuration ^  
    ShardCount=2
```

Es gibt die folgende JSON-Antwort zurück:

```
{  
  "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  
  }  
}
```

```
}  
}
```

Verwenden Sie den folgenden Befehl, um die Details des aktualisierten Clusters anzuzeigen, sobald sich sein Status von Update zu Verfügbar ändert:

Für Linux, macOS oder Unix:

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

Für Windows:

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

Es wird die folgende JSON-Antwort zurückgegeben:

```
{  
  "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
  }
]
}

```

Weitere Informationen finden Sie unter [update-cluster](#) in der AWS CLI Befehlsreferenz.

Entfernen von Shards (MemoryDB API)

Sie können die MemoryDB-API verwenden, um die Shards in Ihrem MemoryDB-Cluster online neu zu konfigurieren, indem Sie die `UpdateCluster` Operation verwenden.

Im Folgenden wird beschrieben, wie Sie die Shards in Ihrem MemoryDB-Cluster neu konfigurieren können, indem Sie Shards mithilfe der MemoryDB-API entfernen.

Important

Bevor Sie Shards aus Ihrem Cluster entfernen, überprüft MemoryDB, ob alle Ihre Daten in die verbleibenden Shards passen. Wenn die Daten passen, werden Shards wie gewünscht aus dem Cluster gelöscht und ihre Keyspaces den verbleibenden Shards zugeordnet. Wenn die

Daten nicht in die verbleibenden Shards passen, wird der Prozess beendet und der Cluster behält dieselbe Shard-Konfiguration wie vor der Anfrage.

Sie können die MemoryDB-API verwenden, um einen oder mehrere Shards aus Ihrem MemoryDB-Cluster zu entfernen. Sie können nicht alle Shards in einem Cluster entfernen. Stattdessen müssen Sie den Cluster löschen. Weitere Informationen finden Sie unter [Schritt 4: Löschen eines Clusters](#).

Verwenden Sie die folgenden Parameter mit `UpdateCluster`.

Parameter

- `ClusterName` – Erforderlich. Gibt an, auf welchem Cluster (Cluster) der Shard-Rekonfigurationsvorgang ausgeführt werden soll.
- `ShardConfiguration` – Erforderlich. Ermöglicht es Ihnen, die Anzahl der Shards mithilfe der `ShardCount` Eigenschaft festzulegen:

`ShardCount`— Legen Sie diese Eigenschaft fest, um die Anzahl der gewünschten Shards anzugeben.

Vertikales Online-Skalieren durch Ändern des Knotentyps

Durch die Verwendung von vertikaler Skalierung, mit MemoryDB, können Sie Ihren Cluster dynamisch ohne Ausfallzeiten skalieren. Dadurch kann Ihr Cluster Anfragen auch während der Skalierung bearbeiten.

Note

Die Skalierung zwischen einem Cluster mit Daten-Tiering (z. B. ein Cluster, der einen R6gd-Knotentyp verwendet) und einem Cluster ohne Daten-Tiering (z. B. ein Cluster, der einen R6g-Knotentyp verwendet) wird nicht unterstützt. Weitere Informationen finden Sie unter [Daten-Tiering](#).

Sie haben die folgenden Möglichkeiten:

- Hochskalierung — Erhöhen Sie die Lese- und Schreibkapazität, indem Sie den Knotentyp Ihres MemoryDB-Clusters auf die Verwendung eines größeren Knotens anpassen.

MemoryDB passt die Größe Ihres Clusters dynamisch an, während dieser online bleibt und Anfragen bearbeitet.

- Herunterskalierung – Verringern Sie die Lese- und Schreibkapazität, indem Sie den Knotentyp auf die Verwendung eines kleineren Knotens anpassen. Auch hier passt MemoryDB die Größe Ihres Clusters dynamisch an, während dieser online bleibt und Anfragen bearbeitet. In diesem Fall verringern Sie die Kosten durch die Verkleinerung des Knotens.

Note

Aufwärts- und Abwärtsskalieren basiert auf der Erstellung von neu ausgewählten Knotentypen und der Synchronisierung der neuen Knoten mit den vorherigen. Um reibungsloses Abwärts-/Aufwärtsskalieren zu gewährleisten, führen Sie Folgendes durch:

- Während die vertikale Skalierung ausgelegt ist, vollständig online zu bleiben, basiert sie auf der Synchronisierung von Daten zwischen dem alten und dem neuen Knoten. Wir empfehlen, dass Sie Abwärts-/Aufwärtsskalieren zu einem Zeitpunkt durchführen, an dem der Datenverkehr am geringsten ist.
- Testen Sie das Verhalten Ihrer Anwendung während der Skalierung möglichst in einer Staging-Umgebung.

Online-aufwärtsskalieren

Themen

- [Aufwärtsskalieren von MemoryDB-Clustern \(Konsole\)](#)
- [Skalierung von MemoryDB-Clustern \(AWSCLI\)](#)
- [Skalierung von MemoryDB-Clustern \(MemoryDB-API\)](#)

Aufwärtsskalieren von MemoryDB-Clustern (Konsole)

Im folgenden Verfahren wird beschrieben, wie Sie einen MemoryDB-Cluster mit der skalierenAWS Management Console. Während dieses Vorgangs verarbeitet Ihr MemoryDB-Cluster weiterhin Anfragen mit minimaler Ausfallzeit.

So führen Sie eine Aufwärtsskalierung eines Clusters durch (Konsole)

1. Melden Sie sich bei der MemoryDB for Redis-Konsole anAWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.
2. Wählen Sie aus der Liste der Cluster den Cluster aus.
3. Wählen Sie Aktionen und dann Ändern.
4. Gehen Sie im Dialogfeld „Cluster ändern“ wie folgt vor:
 - Wählen Sie in der Liste Node type den Knotentyp aus, auf den Sie skalieren möchten. Wählen Sie zur Aufwärtsskalierung einen Knotentyp, der größer als Ihr bestehender Knoten ist.
5. Wählen Sie Save Changes (Änderungen speichern).

Der Status des Clusters ändert sich in Modifizieren. Wenn der Status zu available wechselt, ist die Änderung abgeschlossen und Sie können den neuen Cluster verwenden.

Skalierung von MemoryDB-Clustern (AWSCLI)

Im folgenden Verfahren wird beschrieben, wie Sie einen MemoryDB-Cluster mit der skalierenAWS CLI. Während dieses Vorgangs verarbeitet Ihr MemoryDB-Cluster weiterhin Anfragen mit minimaler Ausfallzeit.

So führen Sie eine Aufwärtsskalierung eines MemoryDB-ClustersAWS durch (CLI)

1. Führen Sie den AWS CLI `list-allowed-node-type-updates`-Befehl mit dem folgenden Parameter aus, um die Knotentypen zu bestimmen, die für eine Aufwärtsskalierung verfügbar sind.

Für Linux, macOS oder Unix:

```
aws memorydb list-allowed-node-type-updates \  
  --cluster-name my-cluster-name
```

Für Windows:

```
aws memorydb list-allowed-node-type-updates ^  
  --cluster-name my-cluster-name
```

Die Ausgabe des obigen Befehls sieht in etwa folgendermaßen aus (JSON format).

```
{
  "ScaleUpNodeTypes": [
    "db.r6g.2xlarge",
    "db.r6g.large"
  ],
  "ScaleDownNodeTypes": [
    "db.r6g.large"
  ],
}
```

Weitere Informationen finden Sie unter [list-allowed-node-type-updates](#) in der AWS CLI Referenz.

2. Ändern Sie Ihren Cluster mithilfe des `aws memorydb update-cluster` Befehls und der folgenden Parameter, um die Aufwärtsskalierung auf den neuen, größeren Knotentyp zu ändern.
 - `--cluster-name`— Der Name des Clusters, zu dem Sie aufwärtsskalieren.
 - `--node-type`— Der neue Knotentyp, auf den der Cluster skaliert werden soll. Der Wert muss einer der Knotentypen sein, die in Schritt 1 mit dem Befehl `list-allowed-node-type-updates` zurückgegeben wurden.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \
  --cluster-name my-cluster \
  --node-type db.r6g.2xlarge
```

Für Windows:

```
aws memorydb update-cluster ^
  --cluster-name my-cluster ^
  --node-type db.r6g.2xlarge ^
```

Weitere Informationen finden Sie unter [update-cluster](#).

Skalierung von MemoryDB-Clustern (MemoryDB-API)

Im nachfolgenden Verfahren wird ein Cluster mithilfe der MemoryDB-API vom derzeitigen Knotentyp auf einen neuen, größeren Knotentyp skaliert. Während dieses Prozesses aktualisiert MemoryDB die DNS-Einträge, sodass sie auf die neuen Knoten verweisen. Sie können Cluster mit Auto-Failover-Funktion skalieren, während der Cluster weiterhin online bleibt und eingehende Anfragen bedient.

Die Zeit, die für die Skalierung auf einen größeren Knotentyp benötigt wird, hängt von Ihrem Knotentyp und der Datenmenge in Ihrem aktuellen Cluster ab.

So führen Sie eine Aufwärtsskalierung eines MemoryDB-Clusters durch (MemoryDB API)

1. Führen Sie die `ListAllowedNodeTypeUpdates` MemoryDB-API-Aktion mit dem folgenden Parameter aus, um die Knotentypen zu bestimmen, die für eine Aufwärtsskalierung verfügbar sind.
 - `ClusterName`— der Name des Clusters. Verwenden Sie diesen Parameter, um einen bestimmten Cluster und nicht alle Cluster zu beschreiben.

```
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>
```

Weitere Informationen finden Sie [ListAllowedNodeTypeUpdates](#) in der MemoryDB for Redis API-Referenz.

2. Führen Sie die `UpdateCluster` MemoryDB-API-Aktion mit dem folgenden Parameter aus, um den derzeitigen Cluster auf den neuen Knotentyp zu skalieren.
 - `ClusterName`— der Name des Clusters.
 - `NodeType`— der neue, größere Knotentyp des Clusters in diesem Cluster. Der Wert muss einer der Instance-Typen sein, die in Schritt 1 mit dem Aufruf `ListAllowedNodeTypeUpdates` zurückgegeben wurden.

```
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>
```

Weitere Informationen finden Sie unter [UpdateCluster](#).

Online-abwärtsskalieren

Themen

- [Abwärtsskalieren eines MemoryDB-Clusters \(Konsole\)](#)
- [Herunterskalieren von MemoryDB-Clustern \(AWSCLI\)](#)
- [Herunterskalieren von MemoryDB-Clustern \(MemoryDB-API\)](#)

Abwärtsskalieren eines MemoryDB-Clusters (Konsole)

Im folgenden Verfahren wird beschrieben, wie Sie einen MemoryDB-Cluster mit der rückskalierenAWS Management Console. Während dieses Vorgangs verarbeitet Ihr MemoryDB-Cluster weiterhin Anfragen mit minimaler Ausfallzeit.

So führen Sie eine Abwärtsskalierung eines MemoryDB-Clusters durch (Konsole)

1. Melden Sie sich bei der MemoryDB for Redis-Konsole anAWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.
2. Wählen Sie aus der Liste der Cluster Ihren bevorzugten Cluster.
3. Wählen Sie Aktionen und dann Ändern.

4. Gehen Sie im Dialogfeld „Cluster ändern“ wie folgt vor:
 - Wählen Sie in der Liste Node type den Knotentyp aus, auf den Sie skalieren möchten. Wählen Sie zur Abwärtsskalierung einen Knotentyp, der kleiner als Ihr bestehender Knoten ist. Beachten Sie, dass nicht alle Knotentypen für das Herunterskalieren zur Verfügung stehen.
5. Wählen Sie Save Changes (Änderungen speichern).

Der Status des Clusters ändert sich in Modifizieren. Wenn der Status zu available wechselt, ist die Änderung abgeschlossen und Sie können den neuen Cluster verwenden.

Herunterskalieren von MemoryDB-Clustern (AWSCLI)

Im folgenden Verfahren wird beschrieben, wie Sie einen MemoryDB-Cluster mit der rückskalierenAWS CLI. Während dieses Vorgangs verarbeitet Ihr MemoryDB-Cluster weiterhin Anfragen mit minimaler Ausfallzeit.

So führen Sie eine Abwärtsskalierung eines MemoryDB-ClustersAWS durch (CLI)

1. Führen Sie den AWS CLI-Befehl `list-allowed-node-type-updates` mit dem folgenden Parameter aus, um die Knotentypen zu bestimmen, die für eine Abwärtsskalierung verfügbar sind.

Für Linux, macOS oder Unix:

```
aws memorydb list-allowed-node-type-updates \  
  --cluster-name my-cluster-name
```

Für Windows:

```
aws memorydb list-allowed-node-type-updates ^\  
  --cluster-name my-cluster-name
```

Die Ausgabe des obigen Befehls sieht in etwa folgendermaßen aus (JSON format).

```
{  
  "ScaleUpNodeTypes": [  
    "db.r6g.2xlarge",  
    "db.r6g.large"  ]  
}
```

```
    ],  
    "ScaleDownNodeTypes": [  
        "db.r6g.large"  
    ],  
}
```

Weitere Informationen finden Sie unter [list-allowed-node-type-updates](#).

2. Ändern Sie Ihren Cluster mithilfe des `update-cluster` Befehls und der folgenden Parameter, um ihn auf den neuen, kleineren Knotentyp herunterzuskalieren.

- `--cluster-name`— Der Name des Clusters, auf den Sie abwärtsskalieren.
- `--node-type`— Der neue Knotentyp, auf den der Cluster skaliert werden soll. Der Wert muss einer der Knotentypen sein, die in Schritt 1 mit dem Befehl `list-allowed-node-type-updates` zurückgegeben wurden.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
    --cluster-name my-cluster \  
    --node-type db.r6g.large
```

Für Windows:

```
aws memorydb update-cluster ^  
    --cluster-name my-cluster ^  
    --node-type db.r6g.large
```

Weitere Informationen finden Sie unter [update-cluster](#).

Herunterskalieren von MemoryDB-Clustern (MemoryDB-API)

Im nachfolgenden Verfahren wird ein Cluster mithilfe der MemoryDB-API vom derzeitigen Knotentyp auf einen neuen, kleineren Knotentyp skaliert. Während dieses Vorgangs verarbeitet Ihr MemoryDB-Cluster weiterhin Anfragen mit minimaler Ausfallzeit.

Die Zeit, die für die Skalierung auf einen kleineren Knotentyp benötigt wird, hängt von Ihrem Knotentyp und der Datenmenge in Ihrem aktuellen Cluster ab.

Abwärtsskalieren (MemoryDB API)

1. Führen Sie die [ListAllowedNodeTypeUpdates](#)API mit dem folgenden Parameter aus, um die Knotentypen zu bestimmen, die für eine Abwärtsskalierung verfügbar sind:

- `ClusterName`— der Name des Clusters. Verwenden Sie diesen Parameter, um einen bestimmten Cluster und nicht alle Cluster zu beschreiben.

```
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. Führen Sie die [UpdateCluster](#)API mit dem folgenden Parameter aus, um Ihren derzeitigen Cluster auf den neuen Knotentyp abwärtszuskalieren.

- `ClusterName`— der Name des Clusters.
- `NodeType`— der neue, kleinere Knotentyp des Clusters in diesem Cluster. Der Wert muss einer der Instance-Typen sein, die in Schritt 1 mit dem Aufruf `ListAllowedNodeTypeUpdates` zurückgegeben wurden.

```
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>
```

Konfiguration von Engine-Parametern unter Verwendung von Parametergruppen

MemoryDB for Redis verwendet Parameter, um die Laufzeiteigenschaften Ihrer Knoten und Cluster zu steuern. In der Regel enthalten neuere Engine-Versionen zusätzliche Parameter zur Unterstützung der neueren Funktionalität. Parametertabellen finden Sie unter [Redis-spezifische Parameter](#).

Wie zu erwarten, werden einige Parameterwerte, wie z. B. `maxmemory` durch die Engine und den Knotentyp bestimmt. Eine Tabelle dieser Parameterwerte nach Knotentyp finden Sie unter [MemoryDB-Knotentyp-spezifische Parameter](#).

Themen

- [Parameterverwaltung](#)
- [Stufen der Parametergruppen](#)
- [Erstellen einer Parametergruppe](#)
- [Auflisten von Parametergruppen nach Namen](#)
- [Auflisten der Werte einer Parametergruppe](#)
- [Modifizieren einer Parametergruppe](#)
- [Löschen einer Parametergruppe](#)
- [Redis-spezifische Parameter](#)

Parameterverwaltung

Parameter werden zur leichteren Parameterverwaltung in benannte Gruppen zusammengruppiert. Eine Parametergruppe stellt eine Kombination spezifischer Werte für die Parameter dar, die der Engine-Software während des Systemstarts übergeben werden. Diese Werte bestimmen, wie sich die Engine-Prozesse auf jedem Knoten während der Laufzeit verhalten. Die Parameterwerte für eine bestimmte Parametergruppe gelten für alle Knoten, die der betreffenden Gruppe zugehören. Dabei spielt es keine Rolle, zu welchem Cluster sie gehören.

Zur Feinabstimmung der Leistung Ihres Clusters können Sie einige Parameterwerte bearbeiten oder die Parametergruppe des Clusters ändern.

- Standard-Parametergruppen können nicht geändert oder gelöscht werden. Wenn Sie benutzerdefinierte Parameterwerte benötigen, müssen Sie eine benutzerdefinierte Parametergruppe erstellen.
- Die Familie der Parametergruppe und der Cluster, dem Sie sie zuordnen, müssen kompatibel sein. Wenn auf Ihrem Cluster beispielsweise Redis Version 6 ausgeführt wird, können Sie nur Standardparametergruppen oder benutzerdefinierte Parametergruppen aus der Familie `memorydb_redis6` verwenden.
- Wenn Sie die Parameter eines Clusters ändern, wird die Änderung sofort auf den Cluster angewendet. Dies gilt unabhängig davon, ob Sie die Parametergruppe des Clusters oder einen Parameterwert innerhalb der Parametergruppe des Clusters ändern.

Stufen der Parametergruppen

Parametergruppenebenen von MemoryDB für Redis

Global Default

Die Stammparametergruppe der obersten Ebene für alle MemoryDB for Redis-Kunden in der Region.

Die globale Standardparametergruppe:

- Ist für MemoryDB reserviert und steht dem Kunden nicht zur Verfügung.

Customer Default

Eine Kopie der Parametergruppe Global Default, die für den Kunden erstellt wurde.

Die Standard-Parametergruppe „Kunde“:

- Wird von MemoryDB erstellt und ist Eigentum von MemoryDB.
- Steht dem Kunden zur Verwendung als Parametergruppe für alle Cluster zur Verfügung, auf denen eine Engine-Version ausgeführt wird, die von dieser Parametergruppe unterstützt wird.
- Kann vom Kunden nicht bearbeitet werden.

Customer Owned

Eine Kopie der Standard-Parametergruppe „Kunde“. Eine Parametergruppe, die dem Kunden gehört, wird immer dann erstellt, wenn der Kunde eine Parametergruppe erstellt.

Die Parametergruppe „Kundenbesitz“:

- Wird vom Kunden erstellt und ist dessen Eigentum.
- Kann beliebigen kompatiblen Clustern des Kunden zugeordnet werden.
- Kann vom Kunden geändert werden, um eine benutzerdefinierte Parametergruppe zu erstellen.

Nicht alle Parameterwerte können geändert werden. Weitere Informationen finden Sie unter [Redis-spezifische Parameter](#).

Erstellen einer Parametergruppe

Sie müssen eine neue Parametergruppe erstellen, wenn Sie die Standardwerte für einen oder mehrere Parameterwerte ändern möchten. Sie können eine Parametergruppe mit der MemoryDB-Konsole, der AWS CLI, oder der MemoryDB-API erstellen.

Erstellen einer Parametergruppe (Konsole)

Das folgende Verfahren zeigt, wie Sie mit der MemoryDB-Konsole eine Parametergruppe erstellen.

Um eine Parametergruppe mit der MemoryDB-Konsole zu erstellen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Eine Liste aller verfügbaren Parametergruppen finden Sie, wenn Sie im linken Navigationsbereich Parametergruppen auswählen.
3. Um eine Parametergruppe zu erstellen, wählen Sie Parametergruppe erstellen.

Die Seite „Parametergruppe erstellen“ wird angezeigt.

4. Geben Sie in das Feld Name einen eindeutigen Namen für diese Parametergruppe ein.

Wenn Sie einen Cluster erstellen oder die Parametergruppe eines Clusters ändern, wählen Sie die Parametergruppe nach Namen aus. Daher wird empfohlen, einen informativen und die Familie der Parametergruppe identifizierenden Namen zu wählen.

Namenseinschränkungen für die Parametergruppe:

- Er muss mit einem ASCII-Buchstaben beginnen.
 - Er kann nur ASCII-Buchstaben, Ziffern und Bindestriche („-“) enthalten.
 - Er muss 1 – 255 Zeichen enthalten.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
5. Geben Sie in das Feld Description eine Beschreibung für die Parametergruppe ein.
 6. Wählen Sie im Feld Redis-Versionskompatibilität eine Engine-Version aus, der diese Parametergruppe entspricht.
 7. Fügen Sie in den Tags optional Tags hinzu, um Ihre Parametergruppen zu suchen und zu filtern oder Ihre AWS Kosten zu verfolgen.

8. Um die Parametergruppe zu erstellen, wählen Sie Create.

Um den Vorgang zu beenden, ohne die Parametergruppe zu erstellen, wählen Sie Cancel.

9. Die erstellte Parametergruppe besitzt die Standardwerte der Familie. Zum Ändern der Standardwerte müssen Sie die Parametergruppe abändern. Weitere Informationen finden Sie unter [Modifizieren einer Parametergruppe](#).

Parametergruppe erstellen (AWS CLI)

Um eine Parametergruppe mit dem zu erstellen AWS CLI, verwenden Sie den Befehl `create-parameter-group` mit diesen Parametern.

- `--parameter-group-name` Der Name der Parametergruppe.

Namenseinschränkungen für die Parametergruppe:

- Er muss mit einem ASCII-Buchstaben beginnen.
- Er kann nur ASCII-Buchstaben, Ziffern und Bindestriche („-“) enthalten.
- Er muss 1 – 255 Zeichen enthalten.
- Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
- Er darf nicht mit einem Bindestrich enden.
- `--family` Die Engine und Versionsfamilie der Parametergruppe.
- `--description` Eine vom Benutzer eingegebene Beschreibung der Parametergruppe.

Example

Im folgenden Beispiel wird eine Parametergruppe mit dem Namen `myRedis6X` erstellt, wobei die Familie `memorydb_redis6` als Vorlage verwendet wird.

Für Linux, macOS oder Unix:

```
aws memorydb create-parameter-group \  
  --parameter-group-name myRedis6x \  
  --family memorydb_redis6 \  
  --description "My first parameter group"
```

Für Windows:

```
aws memorydb create-parameter-group ^
  --parameter-group-name myRedis6x ^
  --family memorydb_redis6 ^
  --description "My first parameter group"
```

Die Ausgabe dieses Befehls sollte in etwa folgendermaßen aussehen.

```
{
  "ParameterGroup": {
    "Name": "myRedis6x",
    "Family": "memorydb_redis6",
    "Description": "My first parameter group",
    "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x"
  }
}
```

Die erstellte Parametergruppe besitzt die Standardwerte der Familie. Zum Ändern der Standardwerte müssen Sie die Parametergruppe abändern. Weitere Informationen finden Sie unter [Modifizieren einer Parametergruppe](#).

Weitere Informationen finden Sie unter [create-parameter-group](#).

Eine Parametergruppe erstellen (MemoryDB-API)

Um eine Parametergruppe mithilfe der MemoryDB-API zu erstellen, verwenden Sie die `CreateParameterGroup` Aktion mit diesen Parametern.

- `ParameterGroupName` Der Name der Parametergruppe.

Namenseinschränkungen für die Parametergruppe:

- Er muss mit einem ASCII-Buchstaben beginnen.
- Er kann nur ASCII-Buchstaben, Ziffern und Bindestriche („-“) enthalten.
- Er muss 1 – 255 Zeichen enthalten.
- Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
- Er darf nicht mit einem Bindestrich enden.
- `Family` Die Engine und Versionsfamilie der Parametergruppe. z. B. `memorydb_redis6`.
- `Description` Eine vom Benutzer eingegebene Beschreibung der Parametergruppe.

Example

Im folgenden Beispiel wird eine Parametergruppe mit dem Namen myRedis6X erstellt, wobei die Familie memorydb_redis6 als Vorlage verwendet wird.

```
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>
```

Die Antwort auf diese Aktion sollte in etwa folgendermaßen aussehen.

```
<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>
```

Die erstellte Parametergruppe besitzt die Standardwerte der Familie. Zum Ändern der Standardwerte müssen Sie die Parametergruppe abändern. Weitere Informationen finden Sie unter [Modifizieren einer Parametergruppe](#).

Weitere Informationen finden Sie unter [CreateParameterGroup](#).

Auflisten von Parametergruppen nach Namen

Sie können die Parametergruppen mithilfe der MemoryDB-Konsole, der oder der MemoryDB-API auflisten. AWS CLI

Auflisten von Parametergruppen nach Namen (Konsole)

Das folgende Verfahren zeigt, wie Sie mit der MemoryDB-Konsole eine Liste der Parametergruppen anzeigen können.

Um Parametergruppen mithilfe der MemoryDB-Konsole aufzulisten

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Eine Liste aller verfügbaren Parametergruppen finden Sie, wenn Sie im linken Navigationsbereich Parametergruppen auswählen.

Parametergruppen nach Namen auflisten (AWS CLI)

Verwenden Sie den Befehl, um mit dem AWS CLI eine Liste von Parametergruppen zu generieren `describe-parameter-groups`. Wenn Sie den Namen einer Parametergruppe angeben, wird nur die betreffende Parametergruppe aufgelistet. Wenn Sie keinen Namen einer Parametergruppe angeben, werden bis zu `--max-results` Parametergruppen aufgelistet. In beiden Fällen werden Name, Familie und Beschreibung der Parametergruppe aufgelistet.

Example

Der folgende Beispielcode listet die Parametergruppe `myRedis6x` auf.

Für Linux, macOS oder Unix:

```
aws memorydb describe-parameter-groups \  
  --parameter-group-name myRedis6x
```

Für Windows:

```
aws memorydb describe-parameter-groups ^  
  --parameter-group-name myRedis6x
```

Die Ausgabe dieses Befehls sieht folgendermaßen aus und listet Name, Familie und Beschreibung der Parametergruppe auf.

```
{
  "ParameterGroups": [
    {
      "Name": "myRedis6x",
      "Family": "memorydb_redis6",
      "Description": "My first parameter group",
      "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
    }
  ]
}
```

Example

Der folgende Beispielcode listet die Parametergruppe myRedis6x für Parametergruppen auf, die auf der Redis-Engine-Version 5.0.6 und höher ausgeführt werden.

Für Linux, macOS oder Unix:

```
aws memorydb describe-parameter-groups \
  --parameter-group-name myRedis6x
```

Für Windows:

```
aws memorydb describe-parameter-groups ^
  --parameter-group-name myRedis6x
```

Die Ausgabe dieses Befehls sieht ungefähr so aus und listet den Namen, die Familie und die Beschreibung für die Parametergruppe auf.

```
{
  "ParameterGroups": [
    {
      "Name": "myRedis6x",
      "Family": "memorydb_redis6",
      "Description": "My first parameter group",
      "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
    }
  ]
}
```

```
    }
  ]
}
```

Example

Der folgende Beispielcode listet bis zu 20 Parametergruppen auf.

```
aws memorydb describe-parameter-groups --max-results 20
```

Die JSON-Ausgabe dieses Befehls sieht ungefähr so aus und listet den Namen, die Familie und die Beschreibung für jede Parametergruppe auf.

```
{
  "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"
    },
    ...
  ]
}
```

Weitere Informationen finden Sie unter [describe-parameter-groups](#).

Parametergruppen nach Namen auflisten (MemoryDB-API)

Verwenden Sie die Aktion, um mithilfe der MemoryDB-API eine Liste von Parametergruppen zu generieren. `DescribeParameterGroups` Wenn Sie den Namen einer Parametergruppe angeben, wird nur die betreffende Parametergruppe aufgelistet. Wenn Sie keinen Namen einer Parametergruppe angeben, werden bis zu `MaxResults` Parametergruppen aufgelistet. In beiden Fällen werden Name, Familie und Beschreibung der Parametergruppe aufgelistet.

Example

Der folgende Beispielcode listet bis zu 20 Parametergruppen auf.

```
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>
```

Die Antwort auf diese Aktion sieht ungefähr so aus und listet im Fall von memorydb_redis6 den Namen, die Familie und die Beschreibung für jede Parametergruppe auf.

```
<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

Der folgende Beispielcode listet die Parametergruppe myRedis6X auf.

```
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>
```

Die Antwort auf diese Aktion sieht folgendermaßen aus und listet den Namen, die Familie und die Beschreibung auf.

```
<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>
```

Weitere Informationen finden Sie unter [DescribeParameterGroups](#).

Auflisten der Werte einer Parametergruppe

Sie können die Parameter und ihre Werte für eine Parametergruppe mithilfe der MemoryDB-Konsole, der oder der MemoryDB-API auflisten. AWS CLI

Auflisten der Werte einer Parametergruppe (Konsole)

Das folgende Verfahren zeigt, wie Sie die Parameter und ihre Werte für eine Parametergruppe mithilfe der MemoryDB-Konsole auflisten.

Um die Parameter einer Parametergruppe und ihre Werte mit der MemoryDB-Konsole aufzulisten

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Eine Liste aller verfügbaren Parametergruppen finden Sie, wenn Sie im linken Navigationsbereich Parametergruppen auswählen.
3. Wählen Sie die Parametergruppe aus, für die Sie die Parameter und Werte auflisten möchten, indem Sie den Namen der Parametergruppe als Namen (nicht das Feld daneben) auswählen.

Die Parameter und deren Werten werden unten auf dem Bildschirm aufgelistet. Aufgrund der Anzahl der Parameter müssen Sie möglicherweise nach oben und unten scrollen, um den Parameter zu finden, an dem Sie interessiert sind.

Werte einer Parametergruppe auflisten (AWS CLI)

Um die Parameter einer Parametergruppe und ihre Werte mithilfe von aufzulisten AWS CLI, verwenden Sie den Befehl `describe-parameters`.

Example

Der folgende Beispielpcode listet alle Parameter und ihre Werte für die Parametergruppe `myRedis6x` auf.

Für Linux, macOS oder Unix:

```
aws memorydb describe-parameters \  
  --parameter-group-name myRedis6x
```

Für Windows:

```
aws memorydb describe-parameters ^  
  --parameter-group-name myRedis6x
```

Weitere Informationen finden Sie unter [describe-parameters](#).

Auflisten der Werte einer Parametergruppe (MemoryDB-API)

Verwenden Sie die Aktion, um die Parameter einer Parametergruppe und ihre Werte mithilfe der MemoryDB-API aufzulisten. `DescribeParameters`

Weitere Informationen finden Sie unter [DescribeParameters](#).

Modifizieren einer Parametergruppe

Important

Die Standard-Parametergruppe kann nicht abgeändert werden.

Einige Parameterwerte in einer Parametergruppe können bearbeitet werden. Diese Parameterwerte werden auf die zur Parametergruppe gehörenden Cluster angewendet. Weitere Informationen über den Zeitpunkt, an dem die Änderung eines Parameterwertes von einer Parametergruppe übernommen wird, finden Sie unter [Redis-spezifische Parameter](#).

Ändern einer Parametergruppe (Konsole)

Das folgende Verfahren zeigt, wie Sie den Wert des Parameters mithilfe der MemoryDB-Konsole ändern können. Dieses Verfahren ist bei der Bearbeitung des Wertes aller Parameter gleich.

Um den Wert eines Parameters mit der MemoryDB-Konsole zu ändern

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](#)
2. Eine Liste aller verfügbaren Parametergruppen finden Sie, wenn Sie im linken Navigationsbereich Parametergruppen auswählen.
3. Wählen Sie die Parametergruppe aus, die Sie ändern möchten, indem Sie auf das Optionsfeld links neben dem Namen der Parametergruppe klicken.

Wählen Sie „Aktionen“ und dann „Details anzeigen“. Alternativ können Sie auch den Namen der Parametergruppe wählen, um zur Detailseite zu gelangen.

4. Um den Parameter zu ändern, wählen Sie Bearbeiten. Alle bearbeitbaren Parameter können bearbeitet werden. Möglicherweise müssen Sie zwischen den Seiten wechseln, um den Parameter zu finden, den Sie ändern möchten. Sie können den Parameter auch anhand des Namens, des Werts oder des Typs im Suchfeld suchen.
5. Nehmen Sie alle erforderlichen Änderungen an den Parametern vor.
6. Klicken Sie auf Save changes (Änderungen speichern), um die Änderungen zu speichern.
7. Wenn Sie die Parameterwerte über mehrere Seiten hinweg geändert haben, können Sie alle Änderungen überprüfen, indem Sie „Änderungen in der Vorschau anzeigen“ wählen. Um die Änderungen zu bestätigen, wählen Sie Änderungen speichern. Um weitere Änderungen vorzunehmen, wählen Sie Zurück.
8. Auf der Seite mit den Parameterdetails haben Sie auch die Möglichkeit, auf Standardwerte zurückzusetzen. Um auf Standardwerte zurückzusetzen, wählen Sie Auf Standardwerte zurücksetzen. Checkboxes werden auf der linken Seite aller Parameter angezeigt. Sie können diejenigen auswählen, die Sie zurücksetzen möchten, und zur Bestätigung auf Weiter zum Zurücksetzen klicken.

Wählen Sie im Dialogfeld „Bestätigen“, um die Aktion zum Zurücksetzen zu bestätigen.

9. Auf der Seite mit den Parameterdetails können Sie die Anzahl der Parameter festlegen, die Sie auf jeder Seite sehen möchten. Verwenden Sie das Zahnrad auf der rechten Seite, um diese Änderungen vorzunehmen. Sie können auch die gewünschten Spalten auf der Detailseite aktivieren/deaktivieren. Diese Änderungen gelten für die gesamte Konsolensitzung.

Den Namen des Parameters, den Sie in einem dieser Themen bearbeitet haben, finden Sie unter [Redis-spezifische Parameter](#).

Ändern einer Parametergruppe (AWS CLI)

Um den Wert eines Parameters mit dem zu ändern AWS CLI, verwenden Sie den Befehl `update-parameter-group`.

Den Namen und die zulässigen Werte des Parameters, den Sie ändern möchten, finden Sie unter [Redis-spezifische Parameter](#)

Weitere Informationen finden Sie unter [update-parameter-group](#).

Ändern einer Parametergruppe (MemoryDB-API)

Verwenden Sie die Aktion, um die Parameterwerte einer Parametergruppe mithilfe der MemoryDB-API zu ändern. `UpdateParameterGroup`

Den Namen und die zulässigen Werte des Parameters, den Sie ändern möchten, finden Sie unter [Redis-spezifische Parameter](#)

Weitere Informationen finden Sie unter [UpdateParameterGroup](#).

Löschen einer Parametergruppe

Sie können eine benutzerdefinierte Parametergruppe mithilfe der MemoryDB-Konsole, der oder der AWS CLI MemoryDB-API löschen.

Parametergruppen, die Clustern zugeordnet sind, können nicht gelöscht werden. Standard-Parametergruppen können ebenfalls nicht gelöscht werden.

Löschen einer Parametergruppe (Konsole)

Das folgende Verfahren zeigt, wie Sie eine Parametergruppe mithilfe der MemoryDB-Konsole löschen.

Um eine Parametergruppe mit der MemoryDB-Konsole zu löschen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Eine Liste aller verfügbaren Parametergruppen finden Sie, wenn Sie im linken Navigationsbereich Parametergruppen auswählen.
3. Wählen Sie die Parametergruppen aus, die Sie löschen möchten, indem Sie das Optionsfeld links neben dem Namen der Parametergruppe auswählen.

Wählen Sie `Actions` und dann `Delete` aus.

4. Der Bestätigungsbildschirm `Delete Parameter Groups` wird angezeigt.
5. Um die Parametergruppen zu löschen, geben Sie `Löschen` in das Bestätigungstextfeld ein.

Um die Parametergruppen beizubehalten, wählen Sie `Cancel`.

Löschen einer Parametergruppe (AWS CLI)

Um eine Parametergruppe mit dem zu löschen AWS CLI, verwenden Sie den Befehl `delete-parameter-group`. Der durch `--parameter-group-name` angegebenen Parametergruppe dürfen eine Cluster zugeordnet sein und es darf sich nicht um eine Standard-Parametergruppe handeln, damit sie gelöscht wird.

Der folgende Beispielcode löscht die Parametergruppe `myRedis6x`.

Example

Für Linux, macOS oder Unix:

```
aws memorydb delete-parameter-group \  
  --parameter-group-name myRedis6x
```

Für Windows:

```
aws memorydb delete-parameter-group ^  
  --parameter-group-name myRedis6x
```

Weitere Informationen finden Sie unter [delete-parameter-group](#)

Löschen einer Parametergruppe (MemoryDB-API)

Verwenden Sie die Aktion, um eine Parametergruppe mithilfe der MemoryDB-API zu löschen. `DeleteParameterGroup` Der durch `ParameterGroupName` angegebenen Parametergruppe dürfen eine Cluster zugeordnet sein und es darf sich nicht um eine Standard-Parametergruppe handeln, damit sie gelöscht wird.

Example

Der folgende Beispielcode löscht die `MyRedis6X`-Parametergruppe.

```
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>
```

Weitere Informationen finden Sie unter [DeleteParameterGroup](#).

Redis-spezifische Parameter

Wenn Sie für Ihren Redis-Cluster keine Parametergruppe angeben, wird eine für Ihre Engine-Version geeignete Standard-Parametergruppe verwendet. Sie können die Werte von Parametern in der Standard-Parametergruppe nicht ändern. Sie können jedoch jederzeit eine benutzerdefinierte Parametergruppe erstellen und sie Ihrem Cluster zuordnen, solange die Werte von bedingungsabhängig veränderbaren Parametern in beiden Parametergruppen gleich sind. Weitere Informationen finden Sie unter [Erstellen einer Parametergruppe](#).

Themen

- [Parameteränderungen bei Redis 7](#)
- [Redis: 6 Parameter](#)
- [MemoryDB-Knotentyp-spezifische Parameter](#)

Parameteränderungen bei Redis 7

Note

MemoryDB hat eine Vorschauversion von [Vector Search](#) eingeführt, die eine neue unveränderliche Parametergruppe enthält. `default.memorydb-redis7.search.preview` Diese Parametergruppe ist in der MemoryDB-Konsole und beim Erstellen eines neuen vector-search-enabled Clusters mit dem CLI-Befehl [create-cluster](#) verfügbar. Die Vorabversion ist in den folgenden AWS Regionen verfügbar: USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Tokio) und Europa (Irland).

Parametergruppenfamilie: `memorydb_redis7`

Die in Redis 7 hinzugefügten Parameter lauten wie folgt.

Name	Details	Beschreibung
<code>latency-tracking</code>	<p>Gültige Werte: <code>yes</code>, <code>no</code></p> <p>Standard: <code>no</code></p> <p>Typ: Zeichenfolge</p>	Wenn diese Option auf „Ja“ festgelegt ist, werden die Latenzen pro Befehl protokolliert und die Perzentilverteilung über den Latenzstatistiken-Befehl <code>INFO</code> wird aktiviert. Ebenso werden die kumulativen Latenzverteilungen

Name	Details	Beschreibung
	<p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	<p>(Histogramme) über den LATENCY-Befehl exportiert.</p>
<code>hash-max-listpack-entries</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 512</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	<p>Die maximale Anzahl von Hash-Einträgen, damit der Datensatz komprimiert werden kann.</p>
<code>hash-max-listpack-value</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 64</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	<p>Der Schwellenwert der maximalen Anzahl von Hash-Einträgen, damit der Datensatz komprimiert werden kann.</p>

Name	Details	Beschreibung
<code>zset-max-listpack-entries</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 128</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	Die maximale Anzahl von Einträgen zu sortierten Sätzen, damit der Datensatz komprimiert werden kann.
<code>zset-max-listpack-value</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 64</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	Der Schwellenwert der maximalen Anzahl von Einträgen zu sortierten Sätzen, damit der Datensatz komprimiert werden kann.

Die in Redis 7 geänderten Parameter lauten wie folgt.

Name	Details	Beschreibung
<code>activeresharding</code>	<p>Anpassbar: no. In Redis 7 ist dieser Parameter standardmäßig ausgeblendet und aktiviert. Wenn Sie ihn deaktivieren möchten,</p>	Veränderbar war ja.

Name	Details	Beschreibung
	müssen Sie einen Support-Fall erstellen .	

Die in Redis 7 entfernten Parameter lauten wie folgt.

Name	Details	Beschreibung
<code>hash-max-ziplist-entries</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 512</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	Für die Darstellung kleiner Hash-Kodierungen <code>listpack</code> statt <code>ziplist</code> verwenden
<code>hash-max-ziplist-value</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 64</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	Für die Darstellung kleiner Hash-Kodierungen <code>listpack</code> statt <code>ziplist</code> verwenden
<code>zset-max-ziplist-entries</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 128</p> <p>Typ: Ganzzahl</p>	Verwenden Sie <code>listpack</code> statt <code>ziplist</code> für die Darstellung kleiner Hash-Kodierungen.

Name	Details	Beschreibung
	<p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	
<code>zset-max-ziplist-value</code>	<p>Zulässige Werte: 0+</p> <p>Standard: 64</p> <p>Typ: Ganzzahl</p> <p>Veränderbar: Ja</p> <p>Änderungen treten in Kraft: unmittelbar auf allen Knoten im Cluster</p>	Verwenden Sie <code>listpack</code> statt <code>ziplist</code> für die Darstellung kleiner Hash-Kodierungen.

Redis: 6 Parameter

Note

In der Redis-Engine-Version 6.2, als die R6GD-Node-Familie für die Verwendung mit [Daten-Tiering](#), `noeviction volatile-lru` und `allkeys-lru` max-Memory-Richtlinien eingeführt wurde, wurden Richtlinien für `r6gd`-Knotentypen unterstützt.

Parametergruppenfamilie: `memorydb_redis6`

Die in Redis 6 hinzugefügten Parameter lauten wie folgt.

Name	Details	Beschreibung
<code>maxmemory-policy</code>	Typ: STRING	Die Bereinigungsrichtlinie für Schlüssel, wenn die maximale Speichernutzung erreicht ist.

Name	Details	Beschreibung
	<p>Zulässige Werte: volatile-lru, allkeys-lru, volatile-lfu, allkeys-lfu, volatile-random, allkeys-random, volatile-ttl, noeviction</p> <p>Standard: noeviction</p>	<p>Weitere Informationen finden Sie unter Redis als LRU-Cache verwenden Redis als LRU-Cache verwenden.</p>
list-compress-depth	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 0</p>	<p>Die Komprimierungstiefe ist die Anzahl der quicklist ziplist-Knoten ab jeder Seite der Liste, die von der Komprimierung ausgeschlossen werden sollen. Anfang und Ende der Liste sind für schnelle push- und pop-Operationen immer dekomprimiert. Die Einstellungen sind:</p> <ul style="list-style-type: none"> • 0: Gesamte Komprimierung deaktivieren. • 1: Komprimierung jeweils einen Knoten nach innen vom Anfang und Ende starten. <p>[head]->node->node->...->node->[tail]</p> <p>Alle Knoten außer [head] und [tail] werden komprimiert. <ul style="list-style-type: none"> • 2: Komprimierung jeweils zwei Knoten nach innen vom Anfang und Ende starten. <p>[head]->[next]->node->node->...->node->[prev]->[tail]</p> <p>[head], [next], [prev], [tail] werden nicht komprimiert. Alle anderen Knoten werden komprimiert. <ul style="list-style-type: none"> • usw. </p></p>

Name	Details	Beschreibung
hll-sparse-max-bytes	<p>Typ: INTEGER</p> <p>Zulässige Werte: 1-16000</p> <p>Standard: 3000</p>	<p>HyperLogLog Limit für spärliche Repräsentations-Bytes. Das Limit umfasst den 16-Byte-Header. Wenn eine Darstellung HyperLogLog mit geringer Dichte diese Grenze überschreitet, wird sie in eine Darstellung mit hoher Dichte umgewandelt.</p> <p>Ein Wert größer als 16000 wird nicht empfohlen, da ab einem solchen Wert die dichte Repräsentation effizienter ist.</p> <p>Wir empfehlen einen Wert von etwa 3000, um die Vorteile der platzsparenden Kodierung nutzen zu können, ohne dabei PFADD zu stark zu verlangsamen, was bei der Sparse-Kodierung $O(N)$ ist. Der Wert kann auf ~ 10000 erhöht werden, wenn die CPU, aber der Speicherplatz keine Rolle spielt und der Datensatz aus vielen besteht, deren Kardinalität zwischen HyperLogLogs 0 und 15000 liegt.</p>
lfu-log-factor	<p>Typ: INTEGER</p> <p>Zulässige Werte: 1-</p> <p>Standard: 10</p>	<p>Der Protokollfaktor für die Erhöhung des Schlüsselzählers für die Richtlinie zur Räumung von LFUs.</p>
lfu-decay-time	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 1</p>	<p>Die Zeitspanne in Minuten, die benötigt wird, um den Schlüsselzähler für die Räumungsrichtlinie der LFU zu verringern.</p>

Name	Details	Beschreibung
<code>active-defrag-max-scan-fields</code>	Typ: INTEGER Zulässige Werte: 1-1000000 Standard: 1000	Maximale Anzahl von set/hash/zset/list-Feldern, die beim Hauptwörterbuchscan während der aktiven Defragmentierung verarbeitet werden.
<code>active-defrag-threshold-upper</code>	Typ: INTEGER Zulässige Werte: 1–100 Standard: 100	Maximaler Prozentsatz der Fragmentierung, bei dem der maximale Einsatz aufgewandt wird.
<code>client-output-buffer-limit-pubsub-hard-limit</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 33554432	Für Redis-publish/subscribe-Clients: Wenn der Ausgabepuffer eines Clients die angegebene Anzahl von Bytes erreicht, wird die Verbindung zum Client getrennt.
<code>client-output-buffer-limit-pubsub-soft-limit</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 8388608	Für Publish/Subscribe-Clients von Redis: Wenn der Ausgabepuffer eines Clients die angegebene Anzahl von Byte erreicht, wird die Verbindung zum Client unterbrochen, aber nur, wenn dieser Zustand andauert <code>client-output-buffer-limit-pubsub-soft-seconds</code> .
<code>client-output-buffer-limit-pubsub-soft-seconds</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 60	Für Redis-publish/subscribe-Clients: Wenn der Ausgabepuffer eines Clients länger als diese Anzahl von Sekunden bei <code>client-output-buffer-limit-pubsub-soft-limit</code> Bytes verbleibt, wird die Verbindung zum Client getrennt.

Name	Details	Beschreibung
timeout	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0,20-</p> <p>Standard: 0</p>	<p>Die Anzahl von Sekunden, die ein Knoten wartet, bevor eine Zeitüberschreitung eintritt. Die Werte sind:</p> <ul style="list-style-type: none"> • 0 — Trennen Sie niemals die Verbindung zu einem inaktiven Client. • 1-19 — ungültige Werte. • ≥ 20 — die Anzahl der Sekunden, die ein Knoten wartet, bevor er die Verbindung zu einem inaktiven Client trennt.
notify-keyspace-events	<p>Typ: STRING</p> <p>Zulässige Werte: NULL</p> <p>Standard: NULL</p>	<p>Die Keyspace-Ereignisse, über die Redis Pub/Sub-Clients informieren soll. Standardmäßig sind alle Benachrichtigungen deaktiviert.</p>
maxmemory-samples	<p>Typ: INTEGER</p> <p>Zulässige Werte: 1-</p> <p>Standard: 3</p>	<p>Bei least-recently-used (LRU) time-to-live (TTL) Berechnungen steht dieser Parameter für die Stichprobengröße der zu prüfenden Schlüssel. Redis wählt standardmäßig 3 Schlüssel aus und verwendet den Schlüssel, der zuletzt verwendet wurde.</p>
slowlog-max-len	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 128</p>	<p>Die maximale Länge des „Slow Log“ von Redis. Diese Länge ist unbegrenzt. Seien Sie sich nur bewusst, dass dadurch Speicherplatz verbraucht wird. Sie können den vom Slow Log belegten Speicher mit zurückgewinnen SLOWLOG RESET.</p>

Name	Details	Beschreibung
<code>activerehashing</code>	<p>Typ: STRING</p> <p>Zulässige Werte: ja, nein</p> <p>Standard: yes</p>	<p>Der Hashvorgang für die Haupt-Hash-Tabelle wird zehnmal pro Sekunde erneut durchgeführt. Jeder erneut durchgeführte Hashvorgang verbraucht 1 Millisekunde der CPU-Zeit.</p> <p>Es empfiehlt sich, diesen Wert gering zu halten. Wenn einem Cluster eine neue Parametergruppe zugewiesen wird, muss dieser Wert in der alten und in der neuen Parametergruppe identisch sein.</p>
<code>client-output-buffer-limit-normal-hard-limit</code>	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 0</p>	<p>Wenn der Ausgabepuffer eines Clients die angegebene Anzahl von Bytes erreicht, wird die Verbindung des Clients getrennt. Der Standard ist null (kein festes Limit).</p>
<code>client-output-buffer-limit-normal-soft-limit</code>	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 0</p>	<p>Wenn der Ausgabepuffer eines Clients die angegebene Anzahl von Bytes erreicht, wird die Verbindung des Clients getrennt, aber nur dann, wenn diese Bedingung <code>client-output-buffer-limit-normal-soft-seconds</code> lang andauert. Der Standard ist null (kein weiches Limit).</p>
<code>client-output-buffer-limit-normal-soft-seconds</code>	<p>Typ: INTEGER</p> <p>Zulässige Werte: 0-</p> <p>Standard: 0</p>	<p>Wenn der Ausgabepuffer eines Clients länger als die angegebene Anzahl von Sekunden bei <code>client-output-buffer-limit-normal-soft-limit</code> Bytes verbleibt, wird die Verbindung des Clients getrennt. Der Standard ist null (kein Zeitlimit).</p>

Name	Details	Beschreibung
<code>tcp-keepalive</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 300	Wenn dies auf einen Wert ungleich null (N) eingestellt wird, werden Knoten-Clients alle N Sekunden abgefragt, um sicherzustellen, dass sie noch verbunden sind. Bei der Standardinstellung „0“ finden keine solche Abfragen statt.
<code>active-defrag-cycle-min</code>	Typ: INTEGER Zulässige Werte: 1–75 Standard: 5	Minimaler Aufwand für die Defragmentierung als CPU-Prozentsatz.
<code>stream-node-max-bytes</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 4096	Die Struktur der Stream-Daten ist eine baumartige Struktur von Knoten, die mehrere Elemente im Innern codieren. Mit dieser Konfiguration legen Sie die maximale Größe (in Bytes) eines einzelnen Knotens in der Baumstruktur fest. Wird die Einstellung „0“ gewählt, ist die Größe des Baumknotens unbegrenzt.
<code>stream-node-max-entries</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 100	Die Struktur der Stream-Daten ist eine baumartige Struktur von Knoten, die mehrere Elemente im Innern codieren. Legen Sie mit dieser Konfiguration die maximale Anzahl der Elemente fest, die ein einzelner Knoten enthalten kann, bevor beim Anhängen neuer Stream-Einträge zu einem neuen Knoten gewechselt wird. Wenn der Wert auf 0 gesetzt ist, ist die Anzahl der Elemente im Baumknoten unbegrenzt.

Name	Details	Beschreibung
lazyfree-lazy- eviction	Typ: STRING Zulässige Werte: ja, nein Standard: no	Führen Sie bei Räumungen eine asynchrone Löschung durch.
active-defrag- ignore-bytes	Typ: INTEGER Zulässige Werte: 1048576- Standard: 104857600	Mindestmenge an Fragmentierungsresten für das Starten der aktiven Defragmentierung.
lazyfree-lazy- expire	Typ: STRING Zulässige Werte: ja, nein Standard: no	Führen Sie ein asynchrones Löschen abgelaufener Schlüssel durch.
active-defrag- threshold- lower	Typ: INTEGER Zulässige Werte: 1–100 Standard: 10	Mindestprozentsatz der Fragmentierung zum Starten der aktiven Defragmentierung.
active-defrag- cycle- max	Typ: INTEGER Zulässige Werte: 1–75 Standard: 75	Maximaler Aufwand für die Defragmentierung als CPU-Prozentsatz.
lazyfree-lazy- server- del	Typ: STRING Zulässige Werte: ja, nein Standard: no	Führt eine asynchrone Löschung bei Befehlen durch, die Werte aktualisieren.

Name	Details	Beschreibung
<code>slowlog-log-slower-than</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 10000	Die maximale Ausführungszeit in Mikrosekunden, die überschritten werden muss, damit der Befehl von der <code>slow log</code> Redis-Funktion protokolliert wird. Beachten Sie, dass eine negative Zahl das langsame Protokoll deaktiviert, während ein Wert von Null die Protokollierung aller Befehle erzwingt.
<code>hash-max-ziplist-entries</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 512	Bestimmt die für Hashes verwendete Speichermenge. Hashes mit weniger als der angegebenen Anzahl von Einträgen werden mit besonderer Codierung gespeichert, durch die Platz eingespart wird.
<code>hash-max-ziplist-value</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 64	Bestimmt die für Hashes verwendete Speichermenge. Hashes mit kleineren Einträgen als die angegebene Anzahl von Bytes werden mit besonderer Codierung gespeichert, durch die Platz eingespart wird.
<code>set-max-intset-entries</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 512	Bestimmt die für bestimmte Arten von Sätzen verwendete Speichermenge (Zeichenfolgen, die in Radix 10 Ganzzahlen im Bereich von signierten 64-Bit-Ganzzahlen sind). Solche Sätze mit weniger als der angegebenen Anzahl von Einträgen werden mit besonderer Codierung gespeichert, durch die Platz eingespart wird.

Name	Details	Beschreibung
<code>zset-max-ziplist-entries</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 128	Bestimmt die für sortierte Sätze verwendete Speichermenge. Sortierte Sätze mit weniger als der angegebenen Anzahl von Elementen werden mit besonderer Codierung gespeichert, durch die Platz eingespart wird.
<code>zset-max-ziplist-value</code>	Typ: INTEGER Zulässige Werte: 0- Standard: 64	Bestimmt die für sortierte Sätze verwendete Speichermenge. Sortierte Sätze mit kleineren Einträgen als die angegebene Anzahl von Bytes werden mit besonderer Codierung gespeichert, durch die Platz eingespart wird.
<code>tracking-table-max-keys</code>	Typ: INTEGER Zulässige Werte: 1-100000000 Standard: 1000000	<p>Um das clientseitige Caching zu unterstützen, unterstützt Redis das Nachverfolgen, welche Clients auf welche Schlüssel zugegriffen haben.</p> <p>Wenn der verfolgte Schlüssel geändert wird, werden Invalidierungsnachrichten an alle Clients gesendet, um ihnen mitzuteilen, dass ihre zwischengespeicherten Werte nicht mehr gültig sind. Mit diesem Wert können Sie die Obergrenze dieser Tabelle angeben.</p>
<code>acllog-max-len</code>	Typ: INTEGER Zulässige Werte: 1-10000 Standard: 128	Die maximale Anzahl von Einträgen im ACL-Protokoll.

Name	Details	Beschreibung
<code>active-expire-effort</code>	<p>Typ: INTEGER</p> <p>Zulässige Werte: 1—10</p> <p>Standard: 1</p>	<p>Redis löscht Schlüssel, die ihre Zeit überschritten haben, um von zwei Mechanismen zu leben. In einem wird auf einen Schlüssel zugegriffen und festgestellt, dass er abgelaufen ist. In der anderen, ein periodischer Job Proben Schlüssel und bewirkt, dass diejenigen, die ihre Zeit überschritten haben, ablaufen. Dieser Parameter definiert den Aufwand, den Redis zum Ablaufen von Elementen im periodischen Auftrag verwendet.</p> <p>Der Standardwert von 1 versucht zu vermeiden, dass mehr als 10 Prozent der abgelaufenen Schlüssel noch im Speicher sind. Außerdem wird versucht, mehr als 25 Prozent des gesamten Arbeitsspeichers zu verbrauchen und das System Latenz zu erhöhen. Sie können diesen Wert auf bis zu 10 erhöhen, um den Aufwand für ablaufende Schlüssel zu erhöhen. Der Kompromiss ist eine höhere CPU und eine potenziell höhere Latenz. Wir empfehlen den Wert 1, es sei denn, Sie sehen eine hohe Speicherauslastung und können eine Erhöhung der CPU-Auslastung tolerieren.</p>
<code>lazyfree-lazy-user-del</code>	<p>Typ: STRING</p> <p>Zulässige Werte: ja, nein</p> <p>Standard: no</p>	<p>Gibt an, ob sich das Standardverhalten des DEL Befehls genauso verhält wie UNLINK.</p>

Name	Details	Beschreibung
<code>activedefrag</code>	Typ: STRING Zulässige Werte: ja, nein Standard: no	Defragmentierung des aktiven Speichers aktiviert.
<code>maxclients</code>	Typ: INTEGER Zulässige Werte: 65000 Standard: 65000	Die maximale Anzahl von Clients, die zu jedem beliebigen Zeitpunkt angeschlossen sein können. Nicht veränderbar.
<code>client-query-buffer-limit</code>	Typ: INTEGER Zulässige Werte: 1048576-1073741824 Standard: 1073741824	Maximale Größe eines einzelnen Client-Abfragepuffers. Die Änderung erfolgt sofort.
<code>proto-max-bulk-len</code>	Typ: INTEGER Zulässige Werte: 1048576-536870912 Standard: 536870912	Maximale Größe einer einzelnen Elementanforderung. Die Änderung erfolgt sofort.

MemoryDB-Knotentyp-spezifische Parameter

Obwohl die meisten Parameter über einen einzelnen Wert verfügen, ist bei einigen Parametern der jeweilige Wert vom verwendeten Knotentyp abhängig. Die folgende Tabelle zeigt den Standardwert für die für jeden Knotentyp `maxmemory`. Der Wert `maxmemory` ist die maximale Anzahl von Bytes, die für Ihre Verwendung, für Daten oder für andere Zwecke auf dem Knoten verfügbar sind.

Knotentyp	Maxmemory
db.r7g.large	14037181030
db.r7g.xlarge	28261849702
db.r7g.2xlarge	56711183565
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.klein	1471026299
db.t4g.medium	3317862236

Note

Alle MemoryDB-Instance-Typen müssen in einer Amazon Virtual Private Cloud Cloud-VPC erstellt werden.

Tutorial: Konfiguration einer Lambda-Funktion für den Zugriff auf MemoryDB in einer Amazon VPC

In diesem Tutorial erfahren Sie, wie Sie:

- Erstellen Sie einen MemoryDB-Cluster in Ihrer standardmäßigen Amazon Virtual Private Cloud (Amazon VPC) in der Region US-East-1.
- Erstellen Sie eine Lambda-Funktion für den Zugriff auf den Cluster. Wenn Sie die Lambda-Funktion erstellen, geben Sie Subnetz-IDs in Ihrer Amazon VPC und in einer VPC-Sicherheitsgruppe an, um der Lambda-Funktion den Zugriff auf Ressourcen in Ihrer VPC zu ermöglichen. Zur Veranschaulichung in diesem Tutorial generiert die Lambda-Funktion eine UUID, schreibt sie in den Cluster und ruft sie aus dem Cluster ab.
- Rufen Sie die Lambda-Funktion manuell auf und überprüfen Sie, ob sie auf den Cluster in Ihrer VPC zugegriffen hat.
- Bereinigen Sie die Lambda-Funktion, den Cluster und die IAM-Rolle, die für dieses Tutorial eingerichtet wurden.

Themen

- [Schritt 1: Erstellen eines Clusters](#)
- [Schritt 2: Erstellen einer Lambda-Funktion](#)
- [Schritt 3: Testen der Lambda -Funktion](#)
- [Schritt 4: Aufräumen \(optional\)](#)

Schritt 1: Erstellen eines Clusters

Gehen Sie folgendermaßen vor, um einen Cluster zu erstellen.

Themen

- [Schritt 1.1: Erstellen Sie einen Cluster](#)

- [Schritt 1.2: Kopieren Sie den Cluster-Endpunkt](#)
- [Schritt 1.3: IAM-Rolle erstellen](#)
- [Schritt 1.4: Erstellen Sie eine Zugriffskontrollliste \(ACL\)](#)

Schritt 1.1: Erstellen Sie einen Cluster

In diesem Schritt erstellen Sie in Ihrem Konto mithilfe der AWS Command Line Interface (CLI) einen Cluster in der Standard-Amazon-VPC in der Region us-east-1. Informationen zum Erstellen eines Clusters mithilfe der MemoryDB-Konsole oder API finden Sie unter [Schritt 1: Erstellen eines Clusters](#)

```
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
```

Beachten Sie, dass für das Feld „Status“ der Wert CREATING festgelegt ist. Es kann einige Minuten dauern, bis MemoryDB die Erstellung Ihres Clusters abgeschlossen hat.

Schritt 1.2: Kopieren Sie den Cluster-Endpunkt

Stellen Sie mit dem Befehl sicher, dass MemoryDB die Erstellung des Clusters abgeschlossen hat.

`describe-clusters`

```
aws memorydb describe-clusters \
--cluster-name cluster-01
```

Kopieren Sie die Cluster-Endpunktadresse, die in der Ausgabe angezeigt wird. Sie benötigen diese Adresse, wenn Sie das Bereitstellungspaket für Ihre Lambda-Funktion erstellen.

Schritt 1.3: IAM-Rolle erstellen

1. Erstellen Sie, wie unten dargestellt, ein Dokument mit den IAM-Vertrauensrichtlinien für Ihre Rolle, sodass Ihr Konto die neue Rolle übernehmen kann. Speichern Sie die Richtlinie in einer Datei namens `trust-policy.json`. Achten Sie darauf, `account_id 123456789012` in dieser Richtlinie durch Ihre `account_id` zu ersetzen.

```
{
  "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"
  }
]
}

```

- Erstellen Sie ein IAM-Richtliniendokument wie im Folgenden dargestellt. Speichern Sie die Richtlinie in einer Datei namens `policy.json`. Achten Sie darauf, die `Account_ID` 123456789012 in dieser Richtlinie durch Ihre Account-ID zu ersetzen.

```

{
  "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"
      ]
    }
  ]
}

```

- Erstellen Sie eine IAM-Rolle.

```

aws iam create-role \
--role-name "memorydb-iam-auth-app" \
--assume-role-policy-document file://trust-policy.json

```

- Erstellen Sie die IAM-Richtlinie.

```

aws iam create-policy \
--policy-name "memorydb-allow-all" \

```

```
--policy-document file://policy.json
```

5. Fügen Sie die IAM-Richtlinie an die Rolle an. Achten Sie darauf, die Account_ID 123456789012 in dieser Policy-ARN durch Ihre Account-ID zu ersetzen.

```
aws iam attach-role-policy \  
  --role-name "memorydb-iam-auth-app" \  
  --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

Schritt 1.4: Erstellen Sie eine Zugriffskontrollliste (ACL)

1. Erstellen Sie einen neuen IAM-fähigen Benutzer.

```
aws memorydb create-user \  
  --user-name iam-user-01 \  
  --authentication-mode Type=iam \  
  --access-string "on ~* +@all"
```

2. Erstellen Sie eine ACL und fügen Sie sie dem Cluster hinzu.

```
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
```

Schritt 2: Erstellen einer Lambda-Funktion

Gehen Sie wie folgt vor, um eine Lambda-Funktion zu erstellen.

Themen

- [Schritt 2.1: Das Bereitstellungspaket erstellen](#)
- [Schritt 2.2: Eine IAM-Rolle erstellen \(Ausführungsrolle\)](#)
- [Schritt 2.3: Aktualisieren des Bereitstellungspakets \(Erstellen der Lambda-Funktion\)](#)

Schritt 2.1: Das Bereitstellungspaket erstellen

In diesem Tutorial stellen wir Beispielcode in Python für Ihre Lambda-Funktion bereit.

Python

Der folgende Python-Beispielcode liest und schreibt ein Element in Ihren MemoryDB-Cluster. Kopieren Sie den Code in eine Datei und speichern Sie diese mit dem Namen `app.py`. Achten Sie darauf, den `cluster_endpoint` Wert im Code durch die Endpunktadresse zu ersetzen, die Sie in Schritt 1.2 kopiert haben.

```
from typing import Tuple, Union
from urllib.parse import ParseResult, urlencode, urlunparse

import boto3.session
import redis
from boto3.model import ServiceId
from boto3.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 = boto3.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"
```

Dieser Code verwendet die `redis-py` Python-Bibliothek, um Elemente in Ihren Cluster einzufügen und abzurufen. Dieser Code dient `dazucachetools`, generierte IAM-Auth-Token 15 Minuten lang zwischenspeichern. Gehen Sie wie folgt vor `cachetools`, um ein Bereitstellungspaket mit `redis-py` und zu erstellen.

Erstellen Sie in Ihrem Projektverzeichnis, das die `app.py` Quellcode-Datei enthält, ein Ordnerpaket, in dem die `cachetools` Bibliotheken `redis-py` und installiert werden sollen.

```
mkdir package
```

Installieren `redis-py` und `cachetools` verwenden Sie Pip.

```
pip install --target ./package redis
pip install --target ./package cachetools
```

Erstellen Sie eine `.zip`-Datei, die die Bibliotheken `redis-py` und `cachetools` enthält. Führen Sie unter Linux und macOS den folgenden Befehl aus. Verwenden Sie in Windows Ihr bevorzugtes ZIP-Hilfsprogramm, um eine `.zip`-Datei mit den `cachetools` Bibliotheken `redis-py` und im Stammverzeichnis zu erstellen.

```
cd package
zip -r ../my_deployment_package.zip
```

Fügen Sie den Funktionscode in die ZIP-Datei ein. Führen Sie unter Linux oder macOS den folgenden Befehl aus: Verwenden Sie in Windows Ihr bevorzugtes ZIP-Programm, um `app.py` zum Stammverzeichnis Ihrer ZIP-Datei hinzuzufügen.

```
cd ..
zip my_deployment_package.zip app.py
```

Schritt 2.2: Eine IAM-Rolle erstellen (Ausführungsrolle)

Hängen Sie die AWS verwaltete Richtlinie mit dem Namen der Rolle `AWSLambdaVPCAccessExecutionRole` an.

```
aws iam attach-role-policy \
```

```
--role-name "memorydb-iam-auth-app" \  
--policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCLambdaAccessExecutionRole"
```

Schritt 2.3: Aktualisieren des Bereitstellungspakets (Erstellen der Lambda-Funktion)

In diesem Schritt erstellen Sie die Lambda-Funktion (AccessMemoryDB) mit dem Befehl AWS CLI `create-function`.

Führen Sie in dem Projektverzeichnis, das die ZIP-Datei Ihres Bereitstellungspakets enthält, den folgenden `create-function` Lambda-CLI-Befehl aus.

Verwenden Sie für die Rollenoption den ARN der Ausführungsrolle, die Sie in Schritt 2.2 erstellt haben. Geben Sie für die `vpc-config` durch Kommas getrennte Listen der Subnetze Ihrer Standard-VPC und die Sicherheitsgruppen-ID Ihrer Standard-VPC ein. Sie finden diese Werte in der Amazon-VPC-Konsole. Um die Subnetze Ihrer Standard-VPC zu finden, wählen Sie Ihre VPCs und dann die Standard-VPC Ihres AWS Kontos aus. Um die Sicherheitsgruppe für diese VPC zu finden, gehen Sie zu Sicherheit und wählen Sie Sicherheitsgruppen aus. Stellen Sie sicher, dass die Region `us-east-1` ausgewählt ist.

```
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
```

Schritt 3: Testen der Lambda -Funktion

In diesem Schritt rufen Sie die Lambda-Funktion manuell mit dem Befehl `invoke` auf. Wenn die Lambda-Funktion ausgeführt wird, generiert sie eine UUID und schreibt sie in den ElastiCache Cache, den Sie in Ihrem Lambda-Code angegeben haben. Die Lambda-Funktion ruft das Element dann aus dem Cache ab.

1. Rufen Sie die Lambda-Funktion (AccessMemoryDB) mit dem Befehl AWS Lambda `invoke` auf.

```
aws lambda invoke \  

```

```
--function-name AccessMemoryDB \  
--region us-east-1 \  
output.txt
```

2. Stellen Sie sicher, dass die Lambda-Funktion erfolgreich ausgeführt wurde:

- Überprüfen Sie die Datei "output.txt".
- Überprüfen Sie die Ergebnisse in CloudWatch Logs, indem Sie die CloudWatch Konsole öffnen und die Protokollgruppe für Ihre Funktion auswählen (/aws/lambda/). AccessRedis Die Ausgabe dieses Protokollstreams sollte ähnlich wie folgt aussehen:

```
Success: Inserted 826e70c5f4d2478c8c18027125a3e01e. Fetched  
826e70c5f4d2478c8c18027125a3e01e from MemoryDB.
```

- Überprüfen Sie die Ergebnisse in der Konsole. AWS Lambda

Schritt 4: Aufräumen (optional)

Gehen Sie zum Aufräumen wie folgt vor.

Themen

- [Schritt 4.1: Lambda-Funktion löschen](#)
- [Schritt 4.2: Löschen Sie den MemoryDB-Cluster](#)
- [Schritt 4.3: Entfernen Sie die IAM-Rolle und -Richtlinien](#)

Schritt 4.1: Lambda-Funktion löschen

```
aws lambda delete-function \  
--function-name AccessMemoryDB
```

Schritt 4.2: Löschen Sie den MemoryDB-Cluster

Löschen Sie den Cluster.

```
aws memorydb delete-cluster \  
--cluster-name cluster-01
```

Benutzer und ACL entfernen.

```
aws memorydb delete-user \  
  --user-id iam-user-01  
  
aws memorydb delete-acl \  
  --acl-name iam-acl-01
```

Schritt 4.3: Entfernen Sie die IAM-Rolle und -Richtlinien

```
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"
```

Vektor-Suche

Diese Funktion befindet sich in der Vorschauversion für MemoryDB for Redis und kann sich ändern.

Die Vektorsuche für MemoryDB erweitert die Funktionalität von MemoryDB. Die Vektorsuche kann in Verbindung mit vorhandenen MemoryDB-Funktionen verwendet werden. Anwendungen, die die Vektorsuche nicht verwenden, sind von ihrer Präsenz nicht betroffen. Die Vorschau der Vektorsuche ist ab MemoryDB 7.1 in den folgenden Regionen verfügbar: USA Ost (Nord-Virginia und Ohio), USA West (Oregon), EU (Irland) und Asien-Pazifik (Tokio).

Die Vektorsuche für Amazon MemoryDB for Redis vereinfacht Ihre Anwendungsarchitektur und bietet gleichzeitig eine Hochgeschwindigkeits-Vektorsuche. Die Vektorsuche für MemoryDB ist ideal für Anwendungsfälle, bei denen Spitzenleistung und Skalierbarkeit die wichtigsten Auswahlkriterien sind. Sie können Ihre vorhandenen MemoryDB-Daten oder die Redis-API verwenden, um Anwendungsfälle für maschinelles Lernen und generative KI zu erstellen, wie z. B. Generierung mit erweitertem Abruf, Erkennung von Anomalien, Abruf von Dokumenten und Empfehlungen in Echtzeit.

Themen

- [Überblick über die Vektorsuche](#)
- [Funktionen und Einschränkungen der Vektorsuche](#)
- [Anwendungsfälle](#)
- [Mit dem AWS Management Console](#)
- [Verwenden von AWS Command Line Interface](#)
- [Befehle für die Vektorsuche](#)

Überblick über die Vektorsuche

Diese Funktion befindet sich in der Vorschauversion für MemoryDB for Redis und kann sich ändern.

Die Vektorsuche basiert auf der Erstellung, Pflege und Verwendung von Indizes. Jede Vektorsuchoperation spezifiziert einen einzelnen Index und ihre Operation ist auf diesen Index beschränkt, d. h. Operationen an einem Index werden von Operationen an einem anderen Index nicht beeinflusst. Mit Ausnahme der Operationen zum Erstellen und Löschen von Indizes können jederzeit beliebig viele Operationen für jeden Index ausgeführt werden, was bedeutet, dass auf Clusterebene mehrere Operationen mit mehreren Indizes gleichzeitig ausgeführt werden können.

Einzelne Indizes sind benannte Objekte, die in einem eindeutigen Namespace existieren, der sich von den anderen Redis-Namespace unterscheidet: Schlüssel, Funktionen usw. Jeder Index ähnelt konzeptionell einer herkömmlichen Datenbanktabelle, da er in zwei Dimensionen strukturiert ist: Spalten und Zeilen. Jede Zeile in der Tabelle entspricht einem Redis-Schlüssel. Jede Spalte im Index entspricht einem Element oder Teil dieses Schlüssels. In diesem Dokument sind die Begriffe Schlüssel, Zeile und Datensatz identisch und werden synonym verwendet. In ähnlicher Weise sind die Begriffe Spalte, Feld, Pfad und Element im Wesentlichen identisch und werden auch synonym verwendet.

Es gibt keine speziellen Befehle zum Hinzufügen, Löschen oder Ändern indizierter Daten. Vielmehr aktualisieren die vorhandenen JSON Befehle HASH oder Befehle, die einen Schlüssel ändern, der sich in einem Index befindet, auch automatisch den Index.

Themen

- [Indizes und der Redis-Keyspace](#)
- [Typen von Indexfeldern](#)
- [Vektor-Index-Algorithmen](#)
- [Ausdruck für eine Vektor-Suchanfrage](#)
- [Befehl INFO](#)
- [Sicherheit bei der Vektorsuche](#)

Indizes und der Redis-Keyspace

Indizes werden über eine Teilmenge des Redis-Schlüsselraums erstellt und verwaltet. Bei mehreren Indizes können unzusammenhängende oder überlappende Teilmengen des Redis-Schlüsselraums ohne Einschränkung ausgewählt werden. Der Schlüsselraum für jeden Index wird durch eine Liste von Schlüsselpräfixen definiert, die bei der Erstellung des Indexes bereitgestellt werden. Die Liste der Präfixe ist optional, und wenn sie weggelassen wird, ist der gesamte Redis-Schlüsselraum Teil dieses Indexes. Indizes werden auch so eingegeben, dass sie nur Schlüssel abdecken, die einen passenden

Typ haben. Derzeit werden nur JSON- und HASH-Indizes unterstützt. Ein HASH-Index indexiert nur HASH-Schlüssel, die in seiner Präfixliste enthalten sind, und in ähnlicher Weise indexiert ein JSON-Index nur JSON-Schlüssel, die in seiner Präfixliste enthalten sind. Schlüssel in der Schlüsselraum-Präfixliste eines Indexes, die nicht den angegebenen Typ haben, werden ignoriert und wirken sich nicht auf Suchvorgänge aus.

Wenn ein HASH- oder JSON-Befehl einen Schlüssel ändert, der sich innerhalb eines Schlüsselraums eines Indexes befindet, wird dieser Index aktualisiert. Dieser Prozess beinhaltet das Extrahieren der deklarierten Felder für jeden Index und das Aktualisieren des Index mit dem neuen Wert. Der Aktualisierungsprozess wird in einem Hintergrundthread durchgeführt, was bedeutet, dass die Indizes erst irgendwann mit ihren Schlüsselrauminhalten konsistent sind. Somit wird das Einfügen oder Aktualisieren eines Schlüssels für einen kurzen Zeitraum nicht in den Suchergebnissen sichtbar sein. In Zeiten hoher Systemlast und/oder starker Datenmutation kann die Sichtbarkeitsverzögerung länger werden.

Die Erstellung eines Index ist ein mehrstufiger Prozess. Der erste Schritt besteht darin, den Befehl [FT.CREATE](#) auszuführen, der den Index definiert. Bei erfolgreicher Ausführung einer Erstellung wird automatisch der zweite Schritt eingeleitet — das Backfilling. Der Backfill-Prozess läuft in einem Hintergrundthread und durchsucht den Redis-Schlüsselbereich nach Schlüsseln, die sich in der Präfixliste des neuen Indexes befinden. Jeder gefundene Schlüssel wird dem Index hinzugefügt. Schließlich wird der gesamte Schlüsselraum gescannt, wodurch der Indexerstellungsprozess abgeschlossen ist. Beachten Sie, dass während der Ausführung des Backfill-Prozesses Mutationen von indizierten Schlüsseln zulässig sind, es keine Einschränkungen gibt und der Index-Backfill-Prozess erst abgeschlossen wird, wenn alle Schlüssel ordnungsgemäß indexiert sind. Abfrageoperationen, die versucht werden, während ein Index aufgefüllt wird, sind nicht zulässig und werden mit einem Fehler beendet. Der Abschluss des Backfill-Vorgangs kann anhand der Ausgabe des `FT.INFO` Befehls für diesen Index ('backfill_status') ermittelt werden.

Typen von Indexfeldern

Jedes Feld (Spalte) eines Indexes hat einen bestimmten Typ, der bei der Indexerstellung deklariert wird, sowie eine Position innerhalb eines Schlüssels. Bei HASH-Schlüsseln ist der Speicherort der Feldname innerhalb des HASH. Bei JSON-Schlüsseln ist der Speicherort eine JSON-Pfadbeschreibung. Wenn ein Schlüssel geändert wird, werden die mit den deklarierten Feldern verknüpften Daten extrahiert, in den deklarierten Typ konvertiert und im Index gespeichert. Wenn die Daten fehlen oder nicht erfolgreich in den deklarierten Typ konvertiert werden können, wird dieses Feld aus dem Index weggelassen. Es gibt vier Arten von Feldern, die im Folgenden erklärt werden:

- Zahlenfelder enthalten eine einzelne Zahl. Bei JSON-Feldern müssen die numerischen Regeln für JSON-Nummern befolgt werden. Bei HASH wird erwartet, dass das Feld den ASCII-Text einer Zahl enthält, die im Standardformat für Fest- oder Fließkommazahlen geschrieben ist. Unabhängig von der Darstellung innerhalb des Schlüssels wird dieses Feld zur Speicherung im Index in eine 64-Bit-Fließkommazahl konvertiert. Zahlenfelder können mit dem Bereichssuchoperator verwendet werden. Da die zugrunde liegenden Zahlen mit ihren Genauigkeitsbeschränkungen in Fließkommazahlen gespeichert werden, gelten die üblichen Regeln für numerische Vergleiche für Fließkommazahlen.
- Tag-Felder enthalten null oder mehr Tag-Werte, die als einzelne UTF-8-Zeichenfolge codiert sind. Die Zeichenfolge wird mithilfe eines Trennzeichens (Standard ist ein Komma, kann aber überschrieben werden) in Tagwerte zerlegt, wobei führende und nachfolgende Leerzeichen entfernt werden. In einem einzigen Tag-Feld können beliebig viele Tag-Werte enthalten sein. Tag-Felder können verwendet werden, um Abfragen nach der Äquivalenz von Tag-Werten zu filtern, wobei entweder Groß- und Kleinschreibung berücksichtigt wird oder nicht.
- Textfelder enthalten einen Blob von Bytes, die nicht UTF-8-kompatibel sein müssen. Textfelder können verwendet werden, um Abfrageergebnisse mit anwendungsrelevanten Werten zu versehen. Zum Beispiel eine URL oder der Inhalt eines Dokuments usw.
- Vektorfelder enthalten einen Zahlenvektor, der auch als Einbettung bezeichnet wird. Vektorfelder unterstützen die Suche nach dem K-Nearest Neighbor (KNN) von Vektoren fester Größe unter Verwendung eines bestimmten Algorithmus und einer bestimmten Entfernungsmetrik. Bei HASH-Indizes sollte das Feld den gesamten Vektor enthalten, der im Binärformat codiert ist (Little-Endian IEEE 754). Bei JSON-Schlüsseln sollte der Pfad auf ein mit Zahlen gefülltes Array der richtigen Größe verweisen. Beachten Sie, dass bei der Verwendung eines JSON-Arrays als Vektorfeld die interne Darstellung des Arrays innerhalb des JSON-Schlüssels in das für den ausgewählten Algorithmus erforderliche Format konvertiert wird, wodurch der Speicherverbrauch und die Genauigkeit reduziert werden. Nachfolgende Lesevorgänge mit den JSON-Befehlen ergeben den reduzierten Genauigkeitswert.

Vektor-Index-Algorithmen

Es stehen zwei Vektorindex-Algorithmen zur Verfügung:

- Flach — Der Flat-Algorithmus ist eine lineare Brute-Force-Verarbeitung aller Vektoren im Index, die exakte Antworten innerhalb der Grenzen der Genauigkeit der Entfernungsberechnungen liefert. Aufgrund der linearen Verarbeitung des Index können die Laufzeiten für diesen Algorithmus bei großen Indizes sehr hoch sein.

- HNSW (Hierarchical Navigable Small Worlds) — Der HNSW-Algorithmus ist eine Alternative, die im Austausch für wesentlich kürzere Ausführungszeiten eine Annäherung an die richtige Antwort liefert. Der Algorithmus wird durch drei Parameter gesteuert, und. M EF_CONSTRUCTION EF_RUNTIME Die ersten beiden Parameter werden bei der Indexerstellung angegeben und können nicht geändert werden. Der EF_RUNTIME Parameter hat einen Standardwert, der bei der Indexerstellung angegeben wird, aber danach bei jedem einzelnen Abfragevorgang überschrieben werden kann. Diese drei Parameter wirken zusammen, um den Speicher- und CPU-Verbrauch bei Aufnahme- und Abfragevorgängen auszugleichen und die Qualität der Annäherung an eine exakte KNN-Suche (bekannt als Recall-Ratio) zu steuern.

Beide Vektorsuchalgorithmen (Flat und HNSW) unterstützen einen optionalen Parameter. INITIAL_CAP Wenn dieser Parameter angegeben ist, weist er den Indizes vorab Speicher zu, was zu einem geringeren Speicherverwaltungsaufwand und höheren Vektoraufnahmeraten führt.

Vektorsuchalgorithmen wie HNSW können das Löschen oder Überschreiben zuvor eingefügter Vektoren möglicherweise nicht effizient handhaben. Die Verwendung dieser Operationen kann zu einem übermäßigen Speicherverbrauch des Indexes und/oder zu einer Verschlechterung der Abrufqualität führen. Die Neuindizierung ist eine Methode zur Wiederherstellung einer optimalen Speichernutzung und/oder eines optimalen Speicherabrufs.

Ausdruck für eine Vektor-Suchanfrage

Die Befehle [FT.SEARCH](#) und [FT.AGGREGATE](#) erfordern einen Abfrageausdruck. Dieser Ausdruck ist ein einzelner Zeichenkettenparameter, der aus einem oder mehreren Operatoren besteht. Jeder Operator verwendet ein Feld im Index, um eine Teilmenge der Schlüssel im Index zu identifizieren. Mehrere Operatoren können mithilfe von booleschen Kombinatoren sowie Klammern kombiniert werden, um den gesammelten Satz von Schlüsseln (oder die Ergebnismenge) weiter zu erweitern oder einzuschränken.

Platzhalter

Der Platzhalteroperator, das Sternchen (*), entspricht allen Schlüsseln im Index.

Numerischer Bereich

Der numerische Bereichsoperator hat die folgende Syntax:

```
<range-search> ::= '@' <numeric-field-name> ':' '[' <bound> <bound> ']'  
<bound> ::= <number> | '(' <number>
```

```
<number> ::= <integer> | <fixed-point> | <floating-point> | 'Inf' | '-Inf' | '+Inf'
```

Bei `< numeric-field-name >` muss es sich um ein deklariertes Feld vom Typ `NUMERIC`. Standardmäßig ist die Grenze inklusiv, aber eine führende offene Klammer `[` (`(`) kann verwendet werden, um eine Grenze exklusiv zu machen. Die Bereichssuche kann mithilfe `Inf` von `+Inf` oder `-Inf` als eine der Grenzen in einen einzelnen relationalen Vergleich (`<`, `<=`, `>`, `>=`) umgewandelt werden. Unabhängig vom angegebenen numerischen Format (Ganzzahl, Festkomma, Gleitkomma, Unendlich) wird die Zahl für Vergleiche in 64-Bit-Gleitkommazahlen umgewandelt, wodurch die Genauigkeit entsprechend reduziert wird.

Example Beispiele

```
@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
```

Tag-Vergleich

Der Tag-Vergleichsoperator hat die folgende Syntax:

```
<tag-search> ::= '@' <tag-field-name> ':' '{' <tag> [ '|' <tag> ]* '}'
```

Wenn eines der Tags im Operator mit einem der Tags im Tag-Feld des Datensatzes übereinstimmt, wird der Datensatz in die Ergebnismenge aufgenommen. Das von dem entworfene Feld `<tag-field-name>` muss ein Feld des mit `type` deklarierten Index sein. TAG Beispiele für einen Tag-Vergleich sind:

```
@tag-field:{ atag }
@tag-field: { tag1 | tag2 }
```

Boolesche Kombinationen

Die Ergebnismengen eines numerischen Operators oder eines Tag-Operators können mithilfe der booleschen Logik kombiniert werden: `und/oder`. Klammern können verwendet werden, um Operatoren zu gruppieren und/oder die Reihenfolge der Auswertung zu ändern. Die Syntax boolescher Logikoperatoren lautet:

```

<expression> ::= <phrase> | <phrase> '|' <expression> | '(' <expression> ')'
<phrase> ::= <term> | <term> <phrase>
<term> ::= <range-search> | <tag-search> | '*'

```

Mehrere Begriffe, die zu einer Phrase zusammengefasst sind, sind „und“ -ed. Mehrere Phrasen, die mit einem senkrechten Strich (|) kombiniert werden, stehen für „oder“.

Vektor-Suche

Der Vektorsuchoperator führt eine Suche nach einem Vektorfeldindex durch, der nach dem K-Nearest Nearest Neighbor gesucht wird. Die Syntax einer Vektorsuche lautet:

```

<vector-search> ::= <expression> '=>[KNN' <k> '@' <vector-field-name> '$' <parameter-name> <modifiers> ']'
<modifiers> ::= [ 'EF_RUNTIME' <integer> ] [ 'AS' <distance-field-name>]

```

Die Vektorsuche wird nur auf die Vektoren angewendet, <expression> die den Kriterien entsprechen. Dabei kann es sich um eine beliebige Kombination der oben definierten Operatoren handeln: Platzhalter, Bereichssuche, Tag-Suche und/oder boolesche Kombinationen davon.

- <k> ist eine Ganzzahl, die die Anzahl der zurückzugebenden Nachbarvektoren angibt.
- <vector-field-name> muss ein deklariertes Feld vom Typ angeben VECTOR.
- <parameter-name> field gibt einen der Einträge für die PARAM Tabelle des FT.AGGREGATE Befehls FT.SEARCH or an. Dieser Parameter ist der Referenzvektorwert für Entfernungsberechnungen. Der Wert des Vektors wird in den PARAM Wert im Little-Endian-Binärformat IEEE 754 codiert (genauso codiert wie bei einem HASH-Vektorfeld)
- Bei Vektorindizes des Typs HNSW kann die optionale EF_RUNTIME Klausel verwendet werden, um den Standardwert des Parameters zu überschreiben, der bei der Indexerstellung festgelegt wurde. EF_RUNTIME
- Die optionale Option <distance-field-name> stellt einen Feldnamen für die Ergebnismenge bereit, der die berechnete Entfernung zwischen dem Referenzvektor und dem gefundenen Schlüssel enthält.

Befehl INFO

Die Vektorsuche erweitert den Redis [INFO-Befehl](#) um mehrere zusätzliche Abschnitte mit Statistiken und Zählern. Eine Anfrage zum Abrufen des Abschnitts SEARCH ruft alle der folgenden Abschnitte ab:

search_memory Abschnitt

Name	Beschreibung
search_used_memory_bytes	Anzahl der in allen Suchdatenstrukturen verbrauchten Speicherbytes
search_used_memory_human	Für Menschen lesbare Version von oben

search_index_stats Abschnitt

Name	Beschreibung
search_number_of_indexes	Anzahl der erstellten Indizes
search_num_fulltext_indexes	Anzahl der Nicht-Vektor-Felder in allen Indizes
search_num_vector_indexes	Anzahl der Vektorfelder in allen Indizes
search_num_hash_indexes	Anzahl der Indizes für Schlüssel vom Typ HASH
search_num_json_indexes	Anzahl der Indizes für Schlüssel vom Typ JSON
search_total_indexed_keys	Gesamtzahl der Schlüssel in allen Indizes
search_total_indexed_vectors	Gesamtzahl der Vektoren in allen Indizes
search_total_indexed_hash_keys	Gesamtzahl der Schlüssel des Typs HASH in allen Indizes
search_total_indexed_json_keys	Gesamtzahl der Schlüssel vom Typ JSON in allen Indizes
search_total_index_size	Von allen Indizes verwendete Bytes
search_total_fulltext_index_size	Bytes, die von Indexstrukturen verwendet werden, die keine Vektoren sind

Name	Beschreibung
search_total_vector_index_size	Von Vektorindexstrukturen verwendete Bytes
search_max_index_lag_ms	Verzögerung bei der Aufnahme während der letzten Aktualisierung des Aufnahme-Batches

search_ingestion Abschnitt

Name	Beschreibung
search_background_indexing_status	Status der Einnahme. NO_ACTIVITY bedeutet untätig. Andere Werte deuten darauf hin, dass sich Schlüssel im Prozess der Aufnahme befinden.
search_ingestion_paused	Außer beim Neustart sollte dies immer „nein“ sein.

search_backfill Abschnitt

Note

Einige der in diesem Abschnitt dokumentierten Felder sind nur sichtbar, wenn gerade ein Backfill im Gange ist.

Name	Beschreibung
search_num_active_backfills	Anzahl der aktuellen Backfill-Aktivitäten
search_backfills_pausiert	Außer wenn nicht genügend Speicher zur Verfügung steht, sollte dies immer „nein“ sein.
search_highest_backfill_progress_percentage	% Abschluss (0-100) der am häufigsten abgeschlossenen Auffüllung

Name	Beschreibung
search_lowest_backfill_progress_percentage	% Abschluss (0-100) der am wenigsten abgeschlossenen Auffüllung

search_query Abschnitt

Name	Beschreibung
search_num_active_queries	Anzahl der Befehle und, die derzeit ausgeführt werden FT .SEARCH FT .AGGREGATE

Sicherheit bei der Vektorsuche

Die Sicherheitsmechanismen von [Redis ACL \(Access Control Lists\)](#) für Befehle und Datenzugriff wurden erweitert, um die Suchfunktion zu kontrollieren. Die ACL-Steuerung einzelner Suchbefehle wird vollständig unterstützt. Eine neue ACL-Kategorie `@search`, wird bereitgestellt, und viele der vorhandenen Kategorien (`@fast@read`, `@write`, usw.) wurden aktualisiert, um die neuen Befehle aufzunehmen. Suchbefehle ändern keine Schlüsseldaten, was bedeutet, dass die bestehende ACL-Maschinerie für den Schreibzugriff erhalten bleibt. Die Zugriffsregeln für HASH- und JSON-Operationen werden durch das Vorhandensein eines Indexes nicht verändert; auf diese Befehle wird weiterhin die normale Zugriffskontrolle auf Schlüsselebene angewendet.

Der Zugriff auf Suchbefehle mit einem Index wird ebenfalls über Redis ACL gesteuert. Zugriffsprüfungen werden auf der Ebene des gesamten Indexes durchgeführt, nicht auf der Ebene pro Schlüssel. Das bedeutet, dass einem Benutzer nur dann Zugriff auf einen Index gewährt wird, wenn dieser Benutzer berechtigt ist, auf alle möglichen Schlüssel in der Schlüsselraumpräfixliste dieses Indexes zuzugreifen. Mit anderen Worten, der tatsächliche Inhalt eines Indexes steuert den Zugriff nicht. Vielmehr ist es der theoretische Inhalt eines Indexes, wie er in der Präfixliste definiert ist, der für die Sicherheitsüberprüfung verwendet wird. Es kann leicht passieren, dass ein Benutzer Lese- und/oder Schreibzugriff auf einen Schlüssel hat, aber nicht auf einen Index zugreifen kann, der diesen Schlüssel enthält. Beachten Sie, dass nur Lesezugriff auf den Schlüsselraum erforderlich ist, um einen Index zu erstellen oder zu verwenden — das Vorhandensein oder Fehlen von Schreibzugriff wird nicht berücksichtigt.

Weitere Informationen zur Verwendung von ACLs mit MemoryDB finden Sie unter [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#).

Funktionen und Einschränkungen der Vektorsuche

Diese Funktion befindet sich in der Vorschauversion von MemoryDB for Redis und kann sich ändern.

Verfügbarkeit der Vektorsuche

Die MemoryDB-Konfiguration mit aktivierter Vektorsuche wird auf den Knotentypen R6g, R7g und T4g unterstützt und ist in den folgenden AWS Regionen verfügbar: USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Tokio) und Europa (Irland).

Parametrische Einschränkungen

Die folgende Tabelle zeigt die Grenzwerte für verschiedene Vektor-Suchelemente in der Vorschau:

Item	Maximaler Wert
Anzahl der Dimensionen in einem Vektor	32768
Anzahl der Indizes, die erstellt werden können	10
Anzahl der Felder in einem Index	50
Anzahl gleichzeitiger FT.CREATE-Backfill-Operationen für den Index	1
FT.SEARCH- und FT.AGGREGATE TIMEOUT-Klausel (MilliSeconds)	60000
Anzahl der Pipeline-Stufen im Befehl FT.AGGREGATE	32
Anzahl der Felder in der FT.AGGREGATE LOAD-Klausel	1024

Item	Maximaler Wert
Anzahl der Felder in der FT.AGGREGATE GROUPBY-Klausel	16
Anzahl der Felder in der FT.AGGREGATE SORTBY-Klausel	16
Anzahl der Parameter in der FT.AGGREGATE PARAM-Klausel	32
HNSW M-Parameter	512
HNSW EF_KONSTRUKTIONSPARAMETER	4096
HNSW EF_RUNTIME-Parameter	4096

Skalierungsgrenzen

Die Vektorsuche für MemoryDB ist derzeit auf einen einzelnen Shard beschränkt und die horizontale Skalierung wird nicht unterstützt. Die Vektorsuche unterstützt die vertikale Skalierung und die Skalierung von Replikaten.

Betriebliche Einschränkungen

Persistenz und Backfilling von Indizes

In der Vorschau der Vektorsuche wird die Definition von Indizes beibehalten, nicht jedoch deren Inhalt. Daher erfordert jede betriebliche Anforderung oder jedes Ereignis, das zum Starten oder Neustarten eines Knotens führt, dass alle Indizes anhand ihrer Definitionen und Quellschlüsseldaten neu erstellt werden. Der Neuaufbauprozess wird automatisch eingeleitet, sobald alle Daten wiederhergestellt wurden — es ist keine Benutzeraktion erforderlich, um ihn zu initiieren. Die Wiederherstellung wird als Backfill-Vorgang ausgeführt, sobald die Daten wiederhergestellt sind. Dies entspricht funktionell der automatischen Ausführung eines [FT.CREATE-Befehls](#) durch das System für jeden definierten Index. Beachten Sie, dass der Knoten für Anwendungsoperationen verfügbar ist, sobald die Daten wiederhergestellt sind, aber wahrscheinlich noch bevor das Auffüllen des Index abgeschlossen ist. Das bedeutet, dass Backfill (s) wieder für Anwendungen sichtbar werden.

Beispielsweise können Suchbefehle, die Backfill-Indizes verwenden, zurückgewiesen werden. Weitere Informationen zum Backfilling finden Sie unter [Überblick über die Vektorsuche](#)

Der Abschluss des Index-Backfills wird nicht zwischen einem Primär- und einem Replikat synchronisiert. Dieser Mangel an Synchronisation kann für Anwendungen unerwartet sichtbar werden. Daher wird empfohlen, dass Anwendungen den Abschluss des Backfill-Vorgangs für Primärdateien und alle Replikate überprüfen, bevor sie Suchvorgänge einleiten.

Import/Export von Snapshots und Live-Migration

Das Vorhandensein von Suchindizes in einer RDB-Datei schränkt die kompatible Übertragbarkeit dieser Daten ein. Das in der Vorschau-Edition definierte Format der Indizes wird nur von einem anderen Preview-Edition-Cluster verstanden. Daher können RDB-Dateien mit Suchindizes nur zwischen MemoryDB-Clustern übertragen oder von diesen genutzt werden, deren Vorschau aktiviert ist.

RDB-Dateien, die keine Indizes enthalten, sind auf diese Weise jedoch nicht eingeschränkt. Somit können Daten innerhalb eines Vorschau-Clusters in Nicht-Vorschau-Cluster exportiert werden, indem die Indizes vor dem Export gelöscht werden.

Speicherverbrauch

Die aktuelle Implementierung von Vektorindizes verbraucht etwa doppelt so viel Speicher wie die Implementierung der allgemeinen Verfügbarkeit.

Nicht genügend Speicher beim Auffüllen

Ähnlich wie bei Redis-Schreiboperationen unterliegt ein Index-Backfill Einschränkungen. out-of-memory Wenn der Redis-Speicher voll ist, während ein Backfill läuft, werden alle Backfills angehalten. Wenn Speicher verfügbar wird, wird der Backfill-Vorgang wieder aufgenommen. Es ist auch möglich, zu löschen und zu indizieren, wenn das Auffüllen aufgrund von Speichermangel unterbrochen wird.

Transaktionen

Die Befehle `FT.CREATE`, `FT.DROPINDEX`, `FT.ALIASADD`, `FT.ALIASDEL`, und `FT.ALIASUPDATE` können nicht in einem Transaktionskontext ausgeführt werden, d. h. nicht innerhalb eines `MULTI/EXEC`-Blocks oder innerhalb eines `LUA`- oder `FUNCTION`-Skripts.

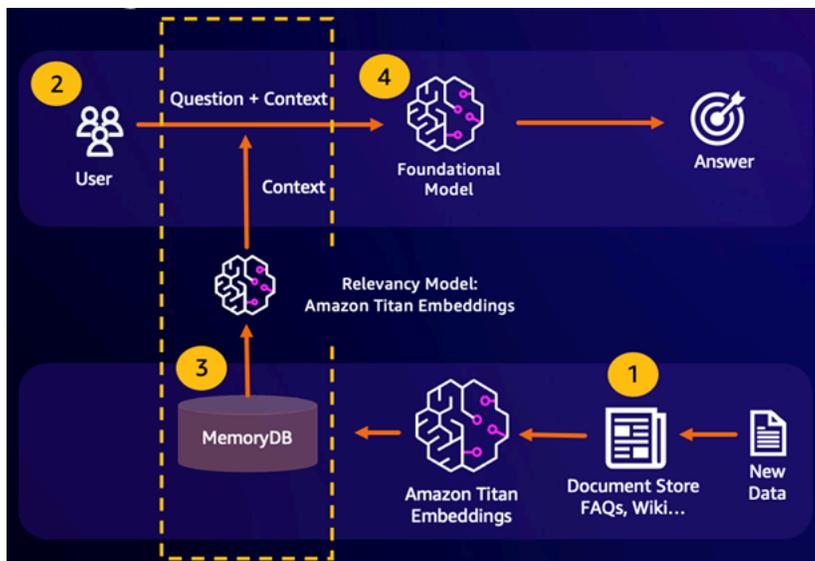
Anwendungsfälle

Diese Funktion befindet sich in der Vorschauversion für MemoryDB for Redis und kann sich ändern.

Im Folgenden sind Anwendungsfälle der Vektorsuche aufgeführt.

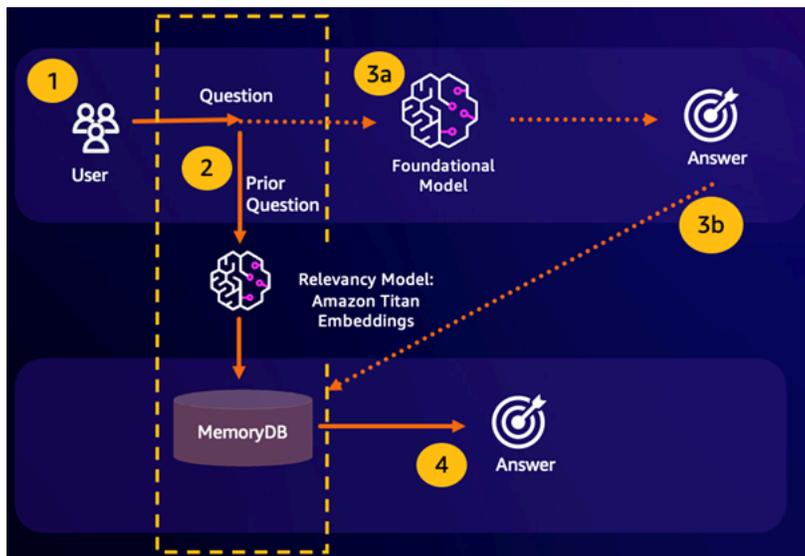
Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) nutzt die Vektorsuche, um relevante Passagen aus einem großen Datenkorpus abzurufen und so ein großes Sprachmodell (LLM) zu erweitern. Konkret bettet ein Encoder den Eingabekontext und die Suchanfrage in Vektoren ein und verwendet dann die ungefähre Suche nach dem nächsten Nachbarn, um semantisch ähnliche Passagen zu finden. Diese abgerufenen Passagen werden mit dem ursprünglichen Kontext verkettet, um dem LLM zusätzliche relevante Informationen zur Verfügung zu stellen, damit der Benutzer eine genauere Antwort erhält.



Foundation Model (FM) -Pufferspeicher

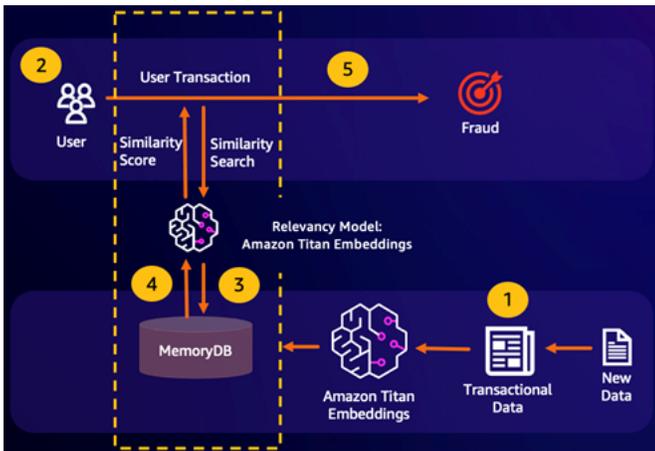
Beim Pufferspeicher (Foundation Model, FM) handelt es sich um ein Verfahren zur Senkung der Rechenkosten, indem frühere FM-Ergebnisse gespeichert werden. Durch die Wiederverwendung früherer Ergebnisse aus früheren Inferenzen, anstatt sie neu zu berechnen, reduziert der FM-Pufferspeicher den Rechenaufwand für die Inferenz durch die FMs. Dieser FM-Pufferspeicher ermöglicht es großen Sprachmodellen, schneller und kostengünstiger zu reagieren, da die FM-Servicegebühren anfallen.



- **Semantischer Suchtreffer** — Wenn die Anfrage eines Kunden aufgrund eines definierten Ähnlichkeitswerts einer vorherigen Frage semantisch ähnlich ist, gibt der FM-Pufferspeicher (MemoryDB) die Antwort auf die vorherige Frage in Schritt 4 zurück und ruft das FM nicht in Schritt 3 auf. Dadurch werden die Latenz und die damit verbundenen Kosten des Foundation Model (FM) vermieden und der Kunde erhält ein schnelleres Erlebnis.
- **Fehlende semantische Suche** — Wenn die Anfrage eines Kunden aufgrund eines definierten Ähnlichkeitswerts einer vorherigen Anfrage semantisch nicht ähnlich ist, ruft ein Kunde den FM an, um dem Kunden in Schritt 3a eine Antwort zu senden. Die vom FM generierte Antwort wird dann als Vektor in MemoryDB für future Abfragen gespeichert (Schritt 3b), um die FM-Kosten für semantisch ähnliche Fragen zu minimieren. In diesem Ablauf würde Schritt 4 nicht aufgerufen werden, da es für die ursprüngliche Abfrage keine semantisch ähnliche Frage gab.

Betrugserkennung

Die Betrugserkennung, eine Form der Anomalieerkennung, stellt gültige Transaktionen als Vektoren dar und vergleicht gleichzeitig die Vektordarstellungen neuer Nettotransaktionen. Betrug wird aufgedeckt, wenn diese Netto-Neutransaktionen eine geringe Ähnlichkeit mit den Vektoren aufweisen, die die gültigen Transaktionsdaten darstellen. Auf diese Weise kann Betrug aufgedeckt werden, indem normales Verhalten modelliert wird, anstatt zu versuchen, jeden möglichen Betrugsfall vorherzusagen. MemoryDB ermöglicht es Unternehmen, dies in Zeiten mit hohem Durchsatz zu tun, mit minimalen Fehlalarmen und Latenz im einstelligen Millisekundenbereich.



Andere Anwendungsfälle

- Empfehlungsmaschinen können Nutzer nach ähnlichen Produkten oder Inhalten suchen, indem sie Artikel als Vektoren darstellen. Die Vektoren werden durch die Analyse von Attributen und Mustern erstellt. Auf der Grundlage von Benutzermustern und -attributen können Benutzern neue unsichtbare Objekte empfohlen werden, indem die ähnlichsten Vektoren gefunden werden, die bereits positiv bewertet wurden und auf den Benutzer abgestimmt sind.
- Dokumentensuchmaschinen stellen Textdokumente als dichte Zahlenvektoren dar und erfassen so semantische Bedeutungen. Bei der Suche wandelt die Suchmaschine eine Suchabfrage in einen Vektor um und findet Dokumente, deren Vektoren der Anfrage am ähnlichsten sind. Dabei wird die ungefähre Suche nach dem nächsten Nachbarn verwendet. Dieser Ansatz zur Vektorähnlichkeit ermöglicht den Abgleich von Dokumenten anhand ihrer Bedeutung und nicht nur anhand übereinstimmender Stichwörter.

Mit dem AWS Management Console

Diese Funktion befindet sich in der Vorschauversion für MemoryDB for Redis und kann sich ändern.

Um einen Cluster zu erstellen, für den die Vektorsuche in der Konsole aktiviert ist, müssen Sie die Vektorsuche in den Cluster-Einstellungen aktivieren. Die Vektorsuche ist für MemoryDB for Redis Version 7.1 und eine Konfiguration mit einem einzelnen Shard verfügbar.

Cluster settings

Enable vector search - *public preview* [Info](#)
You can store vector embeddings and perform vector searches.

i The preview for vector search is compatible with MemoryDB for Redis version 7.1 and a single shard configuration. Vector search and these configurations cannot be modified after creation. We recommend you do not enable this for production clusters.

Weitere Informationen zur Verwendung der Vektorsuche mit dem finden Sie unter [AWS Management Console](#) [Einen Cluster erstellen \(Konsole\)](#)

Verwenden von AWS Command Line Interface

Diese Funktion befindet sich in der Vorschauversion für MemoryDB for Redis und kann sich ändern.

Um einen MemoryDB-Cluster mit aktivierter Vektorsuche zu erstellen, können Sie den Befehl MemoryDB [create-cluster](#) verwenden, indem Sie eine unveränderliche Parametergruppe `default.memorydb-redis7.search.preview` übergeben, um den Vorschaumodus für Vektorsuchfunktionen zu aktivieren.

```
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
```

Befehle für die Vektorsuche

Im Folgenden finden Sie eine Liste der unterstützten Befehle für die Vektorsuche.

Themen

- [FT.CREATE](#)

- [FT.SEARCH](#)
- [FT.AGGREGATE](#)
- [FT.DROPINDEX](#)
- [FT.INFO](#)
- [FT._LISTE](#)
- [FT.ALIASADD](#)
- [FT.ALIASDEL](#)
- [FT.ALIASUPDATE](#)
- [FT._ALIASLISTE](#)
- [FT.CONFIG ABRUFEN](#)
- [FT.CONFIG-HILFE](#)
- [FT.CONFIG GESETZT](#)
- [FT.PROFILE](#)
- [FT.ERLÄUTERN](#)
- [FT.EXPLAINCLI](#)

FT.CREATE

Erzeugt einen Index und initiiert ein Auffüllen dieses Indexes. Weitere Informationen zur Indexkonstruktion finden Sie unter [Überblick über die Vektorsuche](#).

Syntax

```
FT.CREATE <index-name>
ON HASH | JSON
[PREFIX <count> <prefix1> [<prefix2>...]]
SCHEMA
(<field-identifizier> [AS <alias>]
  NUMERIC
| TAG [SEPARATOR <sep>] [CASESENSITIVE]
| TEXT
| VECTOR [HNSW|FLAT] <attr_count> [<attribute_name> <attribute_value>])
)+
```

Schema

- **Feldbezeichner:**
 - Bei Hash-Schlüsseln ist die Feld-ID ein Feldname.
 - Bei JSON-Schlüsseln ist die Feld-ID ein JSON-Pfad.

Weitere Informationen finden Sie unter [Typen von Indexfeldern](#).

- **Feldtypen:**
 - TAG: Weitere Informationen finden Sie unter [Tags](#).
 - NUMERISCH: Das Feld enthält eine Zahl.
 - TEXT: Das Feld enthält einen beliebigen Datenblock.
 - VECTOR: Vektorfeld, das die Vektorsuche unterstützt.
 - Algorithmus — kann HNSW (Hierarchical Navigable Small World) oder FLAT (Brute Force) sein.
 - `attr_count`— Anzahl der Attribute, die als Algorithmuskonfiguration übergeben werden, dazu gehören sowohl Namen als auch Werte.
 - `{attribute_name} {attribute_value}`— Algorithmusspezifische Schlüssel/Wert-Paare, die die Indexkonfiguration definieren.

Für den FLAT-Algorithmus sind die Attribute:

Erforderlich:

- DIM — Anzahl der Dimensionen im Vektor.
- DISTANCE_METRIC — Kann einer von [L2 | IP | COSINE] sein.
- TYPE — Vektortyp. Der einzige unterstützte Typ ist `FLOAT32`.

Optional:

- INITIAL_CAP — Die anfängliche Vektorkapazität im Index beeinflusst die Größe der Speicherzuweisung des Indexes.

Für den HNSW-Algorithmus sind die Attribute:

Erforderlich:

- TYPE — Vektortyp. Der einzige unterstützte Typ ist `FLOAT32`.
- DIM — Vektordimension, angegeben als positive Ganzzahl. Maximum: 32768

- DISTANCE_METRIC — Kann einer von [L2 | IP | COSINE] sein.

Optional:

- `INITIAL_CAP` — Die anfängliche Vektorkapazität im Index beeinflusst die Größe der Speicherzuweisung des Indexes. Der Standardwert ist 1024.
- `M` — Anzahl der maximal zulässigen ausgehenden Kanten für jeden Knoten im Diagramm in jeder Ebene. Auf Ebene Null beträgt die maximale Anzahl ausgehender Kanten 2^m . Die Standardeinstellung ist 16, das Maximum ist 512.
- `EF_CONSTRUCTION` — steuert die Anzahl der Vektoren, die bei der Indexerstellung untersucht werden. Höhere Werte für diesen Parameter verbessern die Rückrufrate auf Kosten längerer Indexierungszeiten. Der Standardwert ist 200. Der Maximalwert ist 4096.
- `EF_RUNTIME` — steuert die Anzahl der Vektoren, die bei Abfrageoperationen untersucht werden. Höhere Werte für diesen Parameter können zu einem besseren Abruf führen, allerdings auf Kosten längerer Abfragezeiten. Der Wert dieses Parameters kann für jede Abfrage außer Kraft gesetzt werden. Standardwert: 10. Der Höchstwert ist 4096.

Ergebnis

Gibt eine einfache Zeichenfolge zurück: OK-Meldung oder Fehlerantwort.

Beispiele

Note

Im folgenden Beispiel werden Argumente verwendet, die für [redis-cli](#) typisch sind, wie z. B. das Entfernen von Anführungszeichen und das Entfernen von Escapes von Daten, bevor sie an Redis gesendet werden. Um andere Programmiersprachenclients (Python, Ruby, C# usw.) zu verwenden, befolgen Sie die Regeln dieser Umgebungen für den Umgang mit Zeichenketten und Binärdaten. [Weitere Informationen zu unterstützten Clients finden Sie unter Tools to Build On AWS](#)

Example 1: Erstellen Sie einige Indizes

Erstellen Sie einen Index für Vektoren der Größe 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
```

Erstellen Sie einen 6-dimensionalen JSON-Index mit dem HNSW-Algorithmus:

```
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 Beispiel 2: Füllen Sie einige Daten aus

Die folgenden Befehle sind so formatiert, dass sie als Argumente für das Redis-CLI-Terminalprogramm ausgeführt werden können. Entwickler, die Programmiersprachen-Clients (wie Python, Ruby, C# usw.) verwenden, müssen die Regeln ihrer Umgebung für den Umgang mit Zeichenketten und Binärdaten befolgen.

Einige Hash- und JSON-Daten erstellen:

```
HSET hash:0 vec "\x00\x00\x00\x00\x00\x00\x00\x00"
HSET hash:1 vec "\x00\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]}'
```

Beachten Sie Folgendes:

- Die Schlüssel der Hash- und JSON-Daten haben die Präfixe ihrer Indexdefinitionen.
- Die Vektoren befinden sich an den entsprechenden Pfaden der Indexdefinitionen.
- Die Hash-Vektoren werden als Hex-Daten eingegeben, während die JSON-Daten als Zahlen eingegeben werden.
- Die Vektoren haben die entsprechenden Längen, die zweidimensionalen Hash-Vektoreinträge haben Hex-Daten im Wert von zwei Fließkommazahlen, die sechsdimensionalen JSON-Vektoreinträge haben sechs Zahlen.

Example Beispiel 3: Löschen Sie einen Index und erstellen Sie ihn neu

```
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
```


- RETURN: Diese Klausel identifiziert, welche Felder eines Schlüssels zurückgegeben werden. Die optionale AS-Klausel für jedes Feld überschreibt den Namen des Felds im Ergebnis. Es können nur Felder angegeben werden, die für diesen Index deklariert wurden.
- LIMIT: <offset><count>: Diese Klausel ermöglicht die Paginierung, da nur die Schlüssel zurückgegeben werden, die den Offset- und Count-Werten entsprechen. Wenn diese Klausel weggelassen wird, wird standardmäßig „LIMIT 0 10“ verwendet, d. h. es werden nur maximal 10 Schlüssel zurückgegeben.
- PARAMS: Zweimal so viele Schlüssel-Wert-Paare. Parameter-Schlüssel/Wert-Paare können innerhalb des Abfrageausdrucks referenziert werden. Weitere Informationen finden Sie unter Abfrageausdruck für die [Vektorsuche](#).
- COUNT: Diese Klausel unterdrückt die Rückgabe des Inhalts von Schlüsseln, es wird nur die Anzahl der Schlüssel zurückgegeben. Dies ist ein Alias für „LIMIT 0 0“.

Ergebnis

Gibt ein Array oder eine Fehlerantwort zurück.

- Wenn der Vorgang erfolgreich abgeschlossen wird, wird ein Array zurückgegeben. Das erste Element ist die Gesamtzahl der Schlüssel, die der Abfrage entsprechen. Die übrigen Elemente sind Paare aus Schlüsselname und Feldliste. Die Feldliste ist ein weiteres Array, das Paare von Feldnamen und Werten umfasst.
- Wenn der Index gerade wieder aufgefüllt wird, gibt der Befehl sofort eine Fehlerantwort zurück.
- Wenn das Timeout erreicht ist, gibt der Befehl eine Fehlerantwort zurück.

Beispiel: Führen Sie einige Suchanfragen durch

Note

Das folgende Beispiel verwendet Argumente, die für [redis-cli](#) typisch sind, wie z. B. das Entfernen von Anführungszeichen und das Entfernen von Escapes von Daten, bevor sie an Redis gesendet werden. Um andere Programmiersprachenclients (Python, Ruby, C# usw.) zu verwenden, befolgen Sie die Regeln dieser Umgebungen für den Umgang mit Zeichenketten und Binärdaten. [Weitere Informationen zu unterstützten Clients finden Sie unter Tools to Build On AWS](#)

angegebenen Reihenfolge angewendet, wobei die Ausgabe einer Klausel die Eingabe der nächsten Klausel speist.

- In der obigen Syntax ist eine „Eigenschaft“ entweder ein Feld, das im Befehl [FT.CREATE](#) für diesen Index deklariert wurde, ODER die Ausgabe einer vorherigen APPLY-Klausel oder REDUCE-Funktion.
- Die LOAD-Klausel ist auf das Laden von Feldern beschränkt, die im Index deklariert wurden. „LOAD *“ lädt alle im Index deklarierten Felder.
- Die folgenden Reducer-Funktionen werden unterstützt: COUNT, COUNT_DISTINCTISH, SUM, MIN, MAX, AVG, STDDEV, QUANTILE, TOLIST, FIRST_VALUE und RANDOM_SAMPLE. [Weitere Informationen finden Sie unter Aggregationen](#)
- LIMIT <offset><count>: Speichert Datensätze ab <offset>und bis zu<count>, alle anderen Datensätze werden verworfen.
- PARAMS: Zweimal so viele Schlüssel-Wert-Paare. Parameter-Schlüssel/Wert-Paare können innerhalb des Abfrageausdrucks referenziert werden. Weitere Informationen finden Sie unter Abfrageausdruck für die [Vektorsuche](#).

Ergebnis

Gibt ein Array oder eine Fehlerantwort zurück.

- Wenn der Vorgang erfolgreich abgeschlossen wird, wird ein Array zurückgegeben. Das erste Element ist eine Ganzzahl ohne besondere Bedeutung (sollte ignoriert werden). Die verbleibenden Elemente sind die Ergebnisse, die von der letzten Stufe ausgegeben wurden. Jedes Element ist ein Array von Feldnamen- und Wertepaaren.
- Wenn der Index gerade wieder aufgefüllt wird, gibt der Befehl sofort eine Fehlerantwort zurück.
- Wenn das Timeout erreicht ist, gibt der Befehl eine Fehlerantwort zurück.

FT.DROPINDEX

Löscht einen Index. Die Indexdefinition und der zugehörige Inhalt werden gelöscht. Redis-Schlüssel sind davon nicht betroffen.

Syntax

```
FT.DROPINDEX <index-name>
```

Ergebnis

Gibt eine einfache Zeichenfolge, eine OK-Nachricht oder eine Fehlerantwort zurück.

FT.INFO

Syntax

```
FT.INFO <index-name>
```

Die Ausgabe der FT.INFO-Seite ist ein Array von Schlüssel-Wert-Paaren, wie in der folgenden Tabelle beschrieben:

Schlüssel	Werttyp	Beschreibung
index_name	Zeichenfolge	Name des Indexes
Erstellungszeitstempel	Ganzzahl	Zeitstempel der Erstellungszeit im UNIX-Stil
Schlüsseltyp	Zeichenfolge	HASH oder JSON
key_prefixes	Zeichenfolgen-Array	Wichtige Präfixe für diesen Index
fields	Reihe von Feldinformationen	Felder dieses Indexes
Space_Usage	Ganzzahl	Von diesem Index verwendete Speicherbytes
fullext_space_usage	Ganzzahl	Speicherbytes, die von Feldern verwendet werden, die keine Vektoren sind
vector_space_usage	Ganzzahl	Von Vektorfeldern verwendete Speicherbytes
num_docs	Ganzzahl	Anzahl der Schlüssel, die derzeit im Index enthalten sind

Schlüssel	Werttyp	Beschreibung
num_indexed_vectors	Ganzzahl	Anzahl der Vektoren, die derzeit im Index enthalten sind
current_lag	Ganzzahl	Aktuelle Verzögerung bei der Aufnahme (Millisekunden)
Backfill_Status	Zeichenfolge	Einer von: Abgeschlossen, InProgres Pausiert oder Fehlgeschlagen

In der folgenden Tabelle werden die Informationen für jedes Feld beschrieben:

Schlüssel	Werttyp	Beschreibung
Bezeichner	Zeichenfolge	Name des Feldes
Feldname	Zeichenfolge	Hash-Mitgliedsname oder JSON-Pfad
Typ	Zeichenfolge	einer der folgenden Werte: Numerisch, Tag, Text oder Vektor
option	Zeichenfolge	ignore

Wenn das Feld vom Typ Vector ist, werden je nach Algorithmus zusätzliche Informationen angezeigt.

Für den HNSW-Algorithmus:

Schlüssel	Werttyp	Beschreibung
Algorithmus	Zeichenfolge	HNSW
data_type	Zeichenfolge	SCHWEBEN 32
distanz_metrisch	Zeichenfolge	einer von: L2, IP oder Cosine

Schlüssel	Werttyp	Beschreibung
initial_capacity	Ganzzahl	Anfangsgröße des Vektorfeld dindexes
aktuelle_Kapazität	Ganzzahl	Aktuelle Größe des Vektorfeld dindexes
maximum_edges	Ganzzahl	M-Parameter bei der Erstellung
ef_construction	Ganzzahl	EF_CONSTRUCTION-Parameter bei der Erstellung
ef_runtime	Ganzzahl	EF_RUNTIME-Parameter bei der Erstellung

Für den FLAT-Algorithmus:

Schlüssel	Werttyp	Beschreibung
Algorithmus	Zeichenfolge	WOHNUNG
data_type	Zeichenfolge	SCHWEBEN 32
distanz_metrisch	Zeichenfolge	einer von: L2, IP oder Cosine
initial_capacity	Ganzzahl	Anfangsgröße des Vektorfeld dindexes
aktuelle_Kapazität	Ganzzahl	Aktuelle Größe des Vektorfeld dindexes

FT. _LISTE

Listet alle Indizes auf.

Syntax

```
FT._LIST
```

Ergebnis

Gibt ein Array von Indexnamen zurück

FT.ALIASADD

Fügen Sie einen Alias für einen Index hinzu. Der neue Aliasname kann überall verwendet werden, wo ein Indexname erforderlich ist.

Syntax

```
FT.ALIASADD <alias> <index-name>
```

Ergebnis

Gibt eine einfache Zeichenfolge, eine OK-Nachricht oder eine Fehlerantwort zurück.

FT.ALIASDEL

Löscht einen vorhandenen Alias für einen Index.

Syntax

```
FT.ALIASDEL <alias>
```

Ergebnis

Gibt eine einfache Zeichenfolge, eine OK-Nachricht oder eine Fehlerantwort zurück.

FT.ALIASUPDATE

Aktualisieren Sie einen vorhandenen Alias so, dass er auf einen anderen physischen Index verweist. Dieser Befehl wirkt sich nur auf future Verweise auf den Alias aus. Derzeit laufende Operationen (FT.SEARCH, FT.AGGREGATE) sind von diesem Befehl nicht betroffen.

Syntax

```
FT.ALIASUPDATE <alias> <index>
```

Ergebnis

Gibt eine einfache OK-Meldung oder eine Fehlerantwort zurück.

FT._ALIASLISTE

Listet die Index-Aliase auf.

Syntax

```
FT._ALIASLIST
```

Ergebnis

Gibt ein Array zurück, das der Größe der Anzahl der aktuellen Aliase entspricht. Jedes Element des Arrays ist das Alias-Indexpaar.

FT.CONFIG ABRUFEN

Gibt den Wert des TIMEOUT-Parameters zurück.

Syntax

```
FT.CONFIG GET [* | <timeout>]
```

Ergebnis

Gibt den Wert des TIMEOUT-Parameters zurück.

FT.CONFIG-HILFE

Ruft Informationen über den TIMEOUT-Parameter ab.

Syntax

```
FT.CONFIG HELP [* | <timeout>]
```

Ergebnis

Gibt Informationen zum TIMEOUT-Parameter zurück

FT.CONFIG GESETZT

Stellen Sie den TIMEOUT-Parameter ein. Der Standardwert ist 10.000 Millisekunden.

Note

Bei konfigurierbaren Parameternamen wird nicht zwischen Groß- und Kleinschreibung unterschieden.

Syntax

```
FT.CONFIG SET <timeout> <value>
```

Ergebnis

Gibt den Wert der TIMEOUT-Einstellung zurück.

FT.PROFILE

Führen Sie eine Abfrage aus und geben Sie Profilinformatoren zu dieser Abfrage zurück.

Syntax

```
FT.PROFILE  
  
<index>  
SEARCH | AGGREGATE  
[LIMITED]  
QUERY <query ....>
```

Ergebnis

Ein Array mit zwei Elementen. Das erste Element ist das Ergebnis des FT.AGGREGATE Befehls FT.SEARCH oder, für den ein Profil erstellt wurde. Das zweite Element besteht aus einer Reihe von Leistungs- und Profilinformatoren.

FT.ERLÄUTERN

Analysieren Sie eine Abfrage und geben Sie Informationen darüber zurück, wie diese Abfrage analysiert wurde.

Syntax

```
FT.EXPLAIN <index> <query>
```

Ergebnis

Eine Zeichenfolge, die die analysierten Ergebnisse enthält.

FT.EXPLAINCLI

Entspricht dem Befehl FT.EXPLAIN, außer dass die Ergebnisse in einem anderen Format angezeigt werden, das mit der Redis-CLI nützlicher ist.

Syntax

```
FT.EXPLAINCLI <index> <query>
```

Ergebnis

Eine Zeichenfolge, die die analysierten Ergebnisse enthält.

Sicherheit in MemoryDB for Redis

Die Sicherheit in der Cloud hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die zur Erfüllung der Anforderungen von Organisationen entwickelt wurden, für die Sicherheit eine kritische Bedeutung hat.

Sicherheit ist eine übergreifende Verantwortlichkeit zwischen AWS und Ihnen. Das [Modell der übergreifenden Verantwortlichkeit](#) beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud selbst – AWS ist dafür verantwortlich, die Infrastruktur zu schützen, mit der AWS-Services in der AWS Cloud ausgeführt werden. AWS stellt Ihnen außerdem Services bereit, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der [AWS-Compliance-Programme](#) regelmäßig. Informationen zu den Compliance-Programmen, die für MemoryDB gelten, finden Sie unter Im Rahmen des Compliance-Programms zugelassene [AWS-Services Im Rahmen des Compliance-Programms](#) zugelassene [-Services Im Rahmen des Compliance-Programms](#) [AWS -Services](#).
- Sicherheit in der Cloud – Ihr Verantwortungsumfang wird durch den AWS-Service bestimmt, den Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation erläutert, wie das Modell der geteilten Verantwortung bei der Verwendung von MemoryDB for Redis zum Tragen kommt. Es zeigt Ihnen, wie Sie MemoryDB konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS -Services nutzen können, die Ihnen helfen, Ihre MemoryDB-Ressourcen zu überwachen und zu sichern.

Inhalt

- [Datenschutz in MemoryDB for Redis](#)
- [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#)
- [Protokollierung und Überwachung](#)
- [Konformitätsprüfung für MemoryDB for Redis](#)
- [Sicherheit der Infrastruktur in Amazon MemoryDB in Amazon MemoryDB in Amazon MemoryDB für Redis](#)
- [Richtlinie für den Datenverkehr zwischen Netzwerken](#)
- [Service-Updates in MemoryDB für Redis](#)

Datenschutz in MemoryDB for Redis

Das [Modell der AWS gemeinsamen Verantwortung](#) und gilt für den Datenschutz in. Wie in diesem Modell beschrieben, ist AWS verantwortlich für den Schutz der globalen Infrastruktur, in der die gesamte AWS Cloud ausgeführt wird. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter [Häufig gestellte Fragen zum Datenschutz](#). Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag [AWS-Modell der geteilten Verantwortung und in der DSGVO](#) im AWS-Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, AWS-Konto-Anmeldeinformationen zu schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einzurichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden zu schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor Authentifizierung (MFA).
- Verwenden Sie SSL/TLS für die Kommunikation mit AWS-Ressourcen. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit AWS CloudTrail ein.
- Verwenden Sie AWS-Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen in AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff auf AWS über eine Befehlszeilenschnittstelle oder über eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter [Federal Information Processing Standard \(FIPS\) 140-2](#).

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit der Konsole, der API oder den AWS SDKs arbeiten oder diese anderweitig AWS-Services verwenden. AWS CLI Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Datensicherheit in MemoryDB für Redis

Um Ihre Daten zu schützen, bieten MemoryDB for Redis und Amazon EC2 Mechanismen zum Schutz vor unbefugtem Zugriff auf Ihre Daten auf dem Server.

MemoryDB bietet auch Verschlüsselungsfunktionen für Daten auf Clustern:

- Bei der Verschlüsselung während der Übertragung werden Ihre Daten bei der Verschiebung von einem Ort an den anderen, z. B. zwischen Knoten in Ihrem Cluster oder zwischen einem Cluster und Ihrer Anwendung, verschlüsselt.
- Bei der Verschlüsselung im Ruhezustand werden das Transaktionsprotokoll und Ihre Daten auf der Festplatte bei Snapshot-Vorgängen verschlüsselt.

Sie können sie auch verwenden [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#), um den Benutzerzugriff auf Ihre Cluster zu kontrollieren.

Themen

- [In MemoryDB gespeicherte Verschlüsselung in MemoryDB](#)
- [Verschlüsselung während der Übertragung \(TLS\) in MemoryDB](#)
- [Benutzer mit Access Control Lists \(ACLs\) authentifizieren](#)
- [Authentifizieren mit IAM](#)

In MemoryDB gespeicherte Verschlüsselung in MemoryDB

Zur Gewährleistung der Sicherheit Ihrer Daten bieten MemoryDB for Redis und Amazon S3 verschiedene Möglichkeiten, den Zugriff auf die Daten in Ihren Clustern zu beschränken.

Weitere Informationen erhalten Sie unter [MemoryDB und Amazon VPC](#) und [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#).

Die Verschlüsselung im Ruhezustand ist immer aktiviert, um die Datensicherheit durch Verschlüsselung persistenter Daten zu erhöhen. Es verschlüsselt die folgenden Aspekte:

- Daten im Transaktionsprotokoll
- Festplatte bei Synchronisierungs-, Snapshot- und Swap-Vorgängen
- In Amazon S3 gespeicherte

MemoryDB bietet eine standardmäßige (vom Dienst verwaltete) Verschlüsselung im [AWSKey-Management Service \(KMS\)](#).

In der Data-Tiering-fähigen fähigen Klasse gespeicherte Daten werden standardmäßig immer verschlüsselt.

Weitere Informationen über Verschlüsselung während der Übertragung finden Sie unter [Verschlüsselung während der Übertragung \(TLS\) in MemoryDB](#).

Themen

- [Verwendung von vom Kunden verwalteten Schlüsseln von AWS KMS](#)
- [Weitere Informationen finden Sie unter:](#)

Verwendung von vom Kunden verwalteten Schlüsseln von AWS KMS

MemoryDB unterstützt symmetrische, vom Kunden verwaltete Root-Schlüssel (KMS-Schlüssel) für die Verschlüsselung im Ruhezustand. Kundenverwaltete KMS-Schlüssel sind Verschlüsselungsschlüssel, die Sie erstellen, besitzen und in Ihrem AWS-Konto verwalten. Weitere Informationen finden Sie unter [dem Entwicklerhandbuch AWS](#) zum -Schlüsselverwaltungsdienst. Die Schlüssel müssen in AWS -KMS erstellt werden, bevor Sie mit MemoryDB verwendet werden können.

Wie Sie AWS-KMS-Root-Schlüssel erstellen können, erfahren Sie unter [Erstellen von Schlüsseln](#) im AWS-Entwicklerhandbuch zum Schlüsselverwaltungsdienst.

S3-Bucket kopieren, der mit Standardverschlüsselung unter Verwendung eines KMS-Schlüssels eingerichtet ist, oder eine Verschlüsselungsoption in der Datei selbst ändern.

- Außerdem können Sie vom Kunden verwaltete Schlüssel verwenden, um manuell erstellte Snapups zu verschlüsseln, die keine vom Kunden verwalteten Schlüssel für die Verschlüsselung verwenden. Bei Amazon S3 gespeicherten wird mit einem KMS-Schlüssel verschlüsselt, auch wenn die Daten im ursprünglichen Cluster nicht verschlüsselt sind.

Bei der Wiederherstellung aus einem Snapshot können Sie aus den verfügbaren Verschlüsselungsoptionen wählen, ähnlich den Verschlüsselungsoptionen, die beim Erstellen eines neuen Clusters verfügbar sind.

- Wenn Sie den Schlüssel löschen oder den Schlüssel [deaktivieren](#) und die Erteilung des Schlüssels, den Sie zum Verschlüsseln eines Clusters verwendet haben, [widerrufen](#), kann der Cluster nicht wiederhergestellt werden. Das bedeutet, dass er nach einem Hardwareausfall nicht geändert oder wiederhergestellt werden kann. AWS KMS löscht Root-Schlüssel erst nach einer Wartezeit von mindestens sieben Tagen. Nach dem Löschen des Schlüssels können Sie einen anderen vom Kunden verwalteten Schlüssel verwenden, um einen Snapups für Archivierungszwecke zu erstellen.
- Bei der automatischen Schlüsselrotation werden die Eigenschaften der AWS -KMS-Root-Schlüssel beibehalten, sodass sich die Rotation nicht auf Ihre Fähigkeit zum Zugriff auf Ihre MemoryDB-Daten auswirkt. Für verschlüsselte MemoryDB-Cups ist keine manuelle Schlüsselrotation möglich, bei der ein neuer Stammschlüssel erstellt und alle Verweise auf den alten Schlüssel aktualisiert werden. Weitere Informationen finden Sie unter [Rotieren von Kundenstammschlüsseln](#) im Entwicklerhandbuch AWS zum -Schlüsselverwaltungsdienst.
- Die Verschlüsselung eines MemoryDB-Cups mit einem KMS-Schlüssel erfordert eine Genehmigung pro Cluster. Dieser Zuschuss wird während der gesamten Lebensdauer des Clusters verwendet. Zusätzlich wird bei der Snapshot-Erstellung ein Grant pro Snapshot verwendet. Dieser Zuschuss wird zurückgezogen, sobald der Snapshot erstellt wurde.
- Weitere Informationen zu AWS -KMS-Genehmigungen und Grenzwerten finden Sie unter [Kontingente](#) im Entwicklerhandbuch AWS zum -Schlüsselverwaltungsdienst.

Weitere Informationen finden Sie unter:

- [Verschlüsselung während der Übertragung \(TLS\) in MemoryDB](#)
- [MemoryDB und Amazon VPC](#)
- [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#)

Verschlüsselung während der Übertragung (TLS) in MemoryDB

Um Ihre Daten zu schützen, bieten MemoryDB for Redis und Amazon EC2 Mechanismen zum Schutz vor unbefugtem Zugriff auf Ihre Daten auf dem Server. Durch die Bereitstellung von Verschlüsselungsfunktionen während der Übertragung bietet Ihnen MemoryDB ein Tool, mit dem Sie Ihre Daten schützen können, wenn sie von einem Ort zum anderen übertragen werden. Sie können beispielsweise Daten von einem primären Knoten auf einen Read Replica-Knoten innerhalb eines Clusters oder zwischen Ihrem Cluster und Ihrer Anwendung verschieben.

Themen

- [Übersicht über die Verschlüsselung während der Übertragung](#)
- [Weitere Informationen finden Sie auch unter](#)

Übersicht über die Verschlüsselung während der Übertragung

Die Verschlüsselung bei der Übertragung mit MemoryDB for Redis ist eine Funktion, die die Sicherheit Ihrer Daten an den anfälligsten Stellen erhöht, d. h. wenn sie von einem Ort zum anderen übertragen werden.

Die MemoryDB-Verschlüsselung bei der Übertragung implementiert die folgenden Funktionen:

- Verschlüsselte Verbindungen — Sowohl die Server- als auch die Client-Verbindungen sind mit Transport Layer Security (TLS) verschlüsselt.
- Verschlüsselte Replikation – Daten, die zwischen einem Primärknoten und Replikationsknoten übertragen werden, sind verschlüsselt.
- Serverauthentifizierung – Clients können die Verbindung zum richtigen Server authentifizieren.

Ab dem 20.07.2023 ist TLS 1.2 die unterstützte Mindestversion für neue und bestehende Cluster. Verwenden Sie diesen [Link](#), um mehr über TLS 1.2 zu erfahren unter. AWS

Weitere Informationen zum Herstellen einer Verbindung zu MemoryDB-Clustern finden Sie unter. [Mit Redis-Cli eine Verbindung zu MemoryDB-Knoten herstellen](#)

Weitere Informationen finden Sie auch unter

- [In MemoryDB gespeicherte Verschlüsselung in MemoryDB](#)
- [Authentifizieren von Benutzern mit Zugriffskontrolllisten \(ACLs\)](#)
- [MemoryDB und Amazon VPC](#)
- [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#)

Benutzer mit Access Control Lists (ACLs) authentifizieren

Sie können Benutzer mit Zugriffskontrolllisten (ACLs) authentifizieren.

Mit ACLs können Sie den Clusterzugriff steuern, indem Sie Benutzer gruppieren. Diese Zugriffskontrolllisten dienen dazu, den Zugriff auf Cluster zu organisieren.

Mit ACLs erstellen Sie Benutzer und weisen ihnen mithilfe einer Zugriffszeichenfolge bestimmte Berechtigungen zu, wie im nächsten Abschnitt beschrieben. Sie weisen die Benutzer Zugriffskontrolllisten zu, denen eine bestimmte Rolle zugewiesen ist (Administratoren, Personalabteilung), die dann in einem oder mehreren MemoryDB-Clustern bereitgestellt werden. Auf diese Weise können Sie Sicherheitsgrenzen zwischen Clients einrichten, die denselben MemoryDB-Cluster oder dieselben MemoryDB-Cluster verwenden, und verhindern, dass Clients gegenseitig auf die Daten zugreifen.

ACLs wurden entwickelt, um die Einführung von [Redis ACL in Redis 6](#) zu unterstützen. Wenn Sie ACLs mit Ihrem MemoryDB-Cluster verwenden, gibt es einige Einschränkungen:

- Sie können keine Passwörter in einer Zugriffsfolge angeben. Sie legen Passwörter mit [CreateUser](#) oder Aufrufen fest. [UpdateUser](#)
- Für Benutzerrechte übergeben Sie `on` und `off` als Teil der Zugriffszeichenfolge. Wenn nichts in der Zugriffszeichenfolge angegeben ist, wird der Benutzer zugewiesen `off` und hat keine Zugriffsrechte für den Cluster.
- Sie können keine verbotenen Befehle verwenden. Wenn Sie einen verbotenen Befehl angeben, wird eine Ausnahme ausgelöst. Eine Liste dieser Befehle finden Sie unter [Beschränkte Redis-Befehle](#).

- Sie können den `reset`-Befehl als Teil einer Zugriffszeichenfolge nicht benutzen. Sie geben Passwörter mit API-Parametern an, und MemoryDB verwaltet Passwörter. Daher können Sie `reset` nicht nutzen, da es alle Kennwörter für einen Benutzer entfernen würde.
- Redis 6 führt den [ACL-LISTE](#)-Befehl ein. Dieser Befehl gibt eine Liste der Benutzer zusammen mit den ACL-Regeln zurück, die auf jeden Benutzer angewendet wurden. MemoryDB unterstützt den `ACL LIST` Befehl, bietet jedoch keine Unterstützung für Passwort-Hashes wie Redis. Mit MemoryDB können Sie den [DescribeUsers](#) Vorgang verwenden, um ähnliche Informationen abzurufen, einschließlich der in der Zugriffszeichenfolge enthaltenen Regeln. Ruft jedoch [DescribeUsers](#) kein Benutzerkennwort ab.

[Andere schreibgeschützte Befehle, die von MemoryDB unterstützt werden, sind ACL WHOAMI, ACL USERS und ACL CAT.](#) MemoryDB unterstützt keine anderen schreibbasierten ACL-Befehle.

Die Verwendung von ACLs mit MemoryDB wird im Folgenden ausführlicher beschrieben.

Themen

- [Spezifizieren von Berechtigungen mithilfe einer Zugriffszeichenfolge](#)
- [Funktionen für die Vektorsuche](#)
- [Anwenden von ACLs auf einen Cluster für MemoryDB](#)

Spezifizieren von Berechtigungen mithilfe einer Zugriffszeichenfolge

Um Berechtigungen für einen MemoryDB-Cluster anzugeben, erstellen Sie eine Zugriffszeichenfolge und weisen sie einem Benutzer zu, indem Sie entweder das oder verwenden. AWS CLI AWS Management Console

Zugriffszeichenfolgen werden als eine Liste von durch Leerzeichen getrennten Regeln definiert, die für den Benutzer angewendet werden. Sie definieren, welche Befehle ein Benutzer ausführen kann und welche Schlüssel ein Benutzer benutzen kann. Um einen Befehl auszuführen, muss ein Benutzer Zugriff auf den ausgeführten Befehl und alle Schlüssel haben, auf die mit dem Befehl zugegriffen wird. Regeln werden kumulativ von links nach rechts angewendet, und eine einfachere Zeichenfolge kann anstelle der angegebenen verwendet werden, wenn die angegebene Zeichenfolge Redundanzen enthält.

Weitere Informationen zur Syntax der ACL-Regeln finden Sie unter [ACL](#).

Im folgenden Beispiel wird durch die Zugriffszeichenfolge ein aktiver Benutzer dargestellt, der Zugriff auf alle verfügbaren Schlüssel und Befehle hat.

```
on ~* &* +@all
```

Die Syntax der Zugriffszeichenfolge wird wie folgt verteilt:

- `on` – Der Benutzer ist ein aktiver Benutzer.
- `~*` – Zugriff auf alle verfügbaren Schlüssel ist erlaubt.
- `&*` – Zugriff wird auf alle Pubsub-Kanäle gewährt.
- `+@all` – Zugriff auf alle verfügbaren Befehle ist erlaubt.

Die vorgenannten Einstellungen sind am wenigsten restriktiv. Sie können diese Einstellungen ändern und sie sicherer zu machen.

Das folgende Beispiel zeigt einen Benutzer, der nur Lesezugriff auf Schlüssel hat, die mit dem Schlüsselbereich "app:." beginnen

```
on ~app:.* -@all +@read
```

Sie können diese Berechtigungen weiter verfeinern, indem Sie die Befehle auflisten, auf die der Benutzer zugreifen kann:

`+command1` – Der Zugriff des Benutzers auf Befehle ist auf *command1* beschränkt.

`+@category` – Der Zugriff des Benutzers auf Befehle ist auf eine Kategorie von Befehlen beschränkt.

Informationen zum Zuweisen einer Zugriffszeichenfolge zu einem Benutzer finden Sie unter [Benutzer und Zugriffskontrolllisten mit der Konsole und CLI erstellen](#).

Wenn Sie einen vorhandenen Workload zu MemoryDB migrieren, können Sie die Zugriffszeichenfolge durch einen Aufruf abrufen `ACL LIST`, ohne den Benutzer und alle Passwort-Hashes.

Funktionen für die Vektorsuche

Note

Diese Funktion befindet sich in der Vorschauversion von MemoryDB for Redis und kann sich ändern.

Denn [Vektor-Suche](#) alle Suchbefehle gehören zur @search Kategorie und zu den vorhandenen Kategorien und wurden aktualisiert @read@write, sodass @fast @slow sie Suchbefehle enthalten. Wenn ein Benutzer keinen Zugriff auf eine Kategorie hat, hat er auch keinen Zugriff auf Befehle innerhalb der Kategorie. Wenn der Benutzer beispielsweise keinen Zugriff auf hat@search, kann er keine suchbezogenen Befehle ausführen.

Die folgende Tabelle zeigt die Zuordnung von Suchbefehlen zu den entsprechenden Kategorien.

VSS-Befehle	@read	@write	@fast	@slow
FT.CREATE		Y	Y	
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.ALIASDELETE		Y	Y	
FT.ALIASUPDATE		Y	Y	

VSS-Befehle	@read	@write	@fast	@slow
FT._ALIAS LIST	Y			Y
FT.EXPLAIN	Y		Y	
FT.EXPLAINCLI	Y		Y	
FT.CONFIG	Y		Y	

Anwenden von ACLs auf einen Cluster für MemoryDB

Um MemoryDB-ACLs zu verwenden, gehen Sie wie folgt vor:

1. Erstellung eines oder mehrerer Benutzer.
2. Erstellen Sie eine ACL und fügen Sie Benutzer zur Liste hinzu.
3. Weisen Sie die ACL einem Cluster zu.

Diese Schritte werden im Folgenden detailliert beschrieben.

Themen

- [Benutzer und Zugriffskontrolllisten mit der Konsole und CLI erstellen](#)
- [Verwaltung von Zugriffskontrolllisten mit der Konsole und CLI](#)
- [Zuweisung von Zugriffskontrolllisten zu Clustern](#)

Benutzer und Zugriffskontrolllisten mit der Konsole und CLI erstellen

Die Benutzerinformationen für ACLs-Benutzer sind ein Benutzername und optional ein Passwort und eine Zugriffszeichenfolge. Die Zugriffszeichenfolge stellt die Berechtigungsstufe für Schlüssel und Befehle bereit. Der Name ist für den Benutzer eindeutig und wird an die Engine übergeben.

Stellen Sie sicher, dass die von Ihnen bereitgestellten Benutzerberechtigungen dem Zweck der ACL entsprechen. Wenn Sie beispielsweise eine ACL mit dem Namen `erstellenAdministrators`, sollte für jeden Benutzer, den Sie zu dieser Gruppe hinzufügen, die Zugriffszeichenfolge auf vollen Zugriff auf Tasten und Befehle gesetzt sein. Für Benutzer in einer `e-commerce` ACL können Sie ihre Zugriffszeichenfolgen auf schreibgeschützten Zugriff festlegen.

MemoryDB konfiguriert automatisch einen Standardbenutzer pro Konto mit einem Benutzernamen `"default"`. Es wird keinem Cluster zugeordnet, es sei denn, es wird ausdrücklich zu einer ACL hinzugefügt. Sie können diesen Benutzer nicht löschen oder ändern. Dieser Benutzer ist für die Kompatibilität mit dem Standardverhalten früherer Redis-Versionen gedacht und hat eine Zugriffszeichenfolge, die es ihm erlaubt, alle Befehle aufzurufen und auf alle Schlüssel zuzugreifen.

Für jedes Konto, das den Standardbenutzer enthält, wird eine unveränderliche „Open-Access“-ACL erstellt. Dies ist die einzige ACL, bei der der Standardbenutzer Mitglied sein kann. Wenn Sie einen Cluster erstellen, müssen Sie eine ACL auswählen, die dem Cluster zugeordnet werden soll. Sie haben zwar die Möglichkeit, die „Open-Access“-ACL für den Standardbenutzer anzuwenden, wir empfehlen jedoch dringend, eine ACL für Benutzer zu erstellen, deren Berechtigungen auf ihre Geschäftsanforderungen beschränkt sind.

Cluster, für die TLS nicht aktiviert ist, müssen die „Open-Access“-ACL verwenden, um eine offene Authentifizierung zu ermöglichen.

ACLs können ohne Benutzer erstellt werden. Eine leere ACL hätte keinen Zugriff auf einen Cluster und kann nur TLS-fähigen Clustern zugeordnet werden.

Beim Erstellen eines Benutzers können Sie bis zu zwei Passwörter einrichten. Wenn Sie ein Passwort ändern, werden alle bestehenden Verbindungen zu Clustern beibehalten.

Beachten Sie insbesondere die folgenden Einschränkungen bei Benutzerkennwörtern, wenn Sie ACLs für MemoryDB verwenden:

- Passwörter müssen 16-128 druckbare Zeichen enthalten.
- Folgende nicht-alphanumerische Zeichen sind nicht zulässig: `,` `"` `'` `/` `@`.

Verwalten von Benutzern mit der Konsole und dem CLI

Einen Benutzer erstellen (Konsole)

Um Benutzer auf der Konsole zu erstellen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Benutzer aus.
3. Wählen Sie Benutzer erstellen
4. Geben Sie auf der Seite „Benutzer erstellen“ einen Namen ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
5. Unter Passwörter können Sie bis zu zwei Passwörter eingeben.
 6. Geben Sie unter Zugriffszeichenfolge eine Zugriffszeichenfolge ein. Die Zugriffszeichenfolge legt die Berechtigungsstufe fest, welche Schlüssel und Befehle für den Benutzer erlaubt sind.
 7. Für Tags können Sie optional Tags anwenden, um Ihre Benutzer zu suchen und zu filtern oder Ihre AWS Kosten nachzuverfolgen.
 8. Wählen Sie Erstellen.

Einen Benutzer mit dem erstellen AWS CLI

So erstellen Sie einen Benutzer mit der CLI

- Verwenden Sie den Befehl [create-user](#), um einen Benutzer zu erstellen.

Für Linux, macOS oder Unix:

```
aws memorydb create-user \  
  --user-name user-name-1 \  
  --access-string "~objects:* ~items:* ~public:*" \  
  --authentication-mode \  
  --passwords "password1 password2" \  
  --access-string-arn arn:aws:iam::123456789012:accesskeypolicy/PolicyName
```

```
Passwords="abc",Type=password
```

Für Windows:

```
aws memorydb create-user ^
--user-name user-name-1 ^
--access-string "~objects:* ~items:* ~public:*" ^
--authentication-mode \
    Passwords="abc",Type=password
```

Einen Benutzer ändern (Konsole)

Um Benutzer auf der Konsole zu ändern

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Benutzer aus.
3. Wählen Sie das Optionsfeld neben dem Benutzer, den Sie ändern möchten, und wählen Sie dann Aktionen -> Ändern
4. Wenn Sie ein Passwort ändern möchten, wählen Sie das Optionsfeld Passwörter ändern. Beachten Sie, dass Sie, wenn Sie zwei Passwörter haben, beide eingeben müssen, wenn Sie eines davon ändern.
5. Wenn Sie die Zugriffszeichenfolge aktualisieren, geben Sie die neue ein.
6. Wählen Sie Ändern aus.

Einen Benutzer ändern mit AWS CLI

So ändern Sie einen Benutzer mit dem CLI

1. Verwenden Sie den Befehl [update-user](#), um einen Benutzer zu ändern.
2. Wenn ein Benutzer geändert wird, werden die mit dem Benutzer verknüpften Zugriffskontrolllisten zusammen mit allen Clustern, die der ACL zugeordnet sind, aktualisiert. Alle vorhandenen Verbindungen werden gewartet. Im Folgenden sind einige Beispiele aufgeführt.

Für Linux, macOS oder Unix:

```
aws memorydb update-user \  
  --user-name user-name-1 \  
  --access-string "~objects:* ~items:* ~public:*
```

Für Windows:

```
aws memorydb update-user ^  
  --user-name user-name-1 ^  
  --access-string "~objects:* ~items:* ~public:*
```

Benutzerdetails anzeigen (Konsole)

Um Benutzerdetails auf der Konsole anzuzeigen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Benutzer aus.
3. Wählen Sie den Benutzer unter Benutzername aus, oder verwenden Sie das Suchfeld, um den Benutzer zu finden.
4. Unter Benutzereinstellungen können Sie die Zugriffszeichenfolge, die Anzahl der Passwörter, den Status und den Amazon-Ressourcennamen (ARN) des Benutzers überprüfen.
5. Unter Zugriffskontrolllisten (ACL) können Sie überprüfen, zu welcher ACL der Benutzer gehört.
6. Unter Tags können Sie alle mit dem Benutzer verknüpften Tags überprüfen.

Benutzerdetails anzeigen mit dem AWS CLI

Verwenden Sie den Befehl [describe-users](#), um Details eines Benutzers anzuzeigen.

```
aws memorydb describe-users \  
  --user-name my-user-name
```

Einen Benutzer löschen (Konsole)

Um Benutzer auf der Konsole zu löschen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Benutzer aus.
3. Wählen Sie das Optionsfeld neben dem Benutzer, den Sie ändern möchten, und wählen Sie dann Aktionen -> Löschen
4. Geben Sie zur Bestätigung den Text delete in das Bestätigungsfeld ein und wählen Sie dann Löschen.
5. Wenn Sie den Vorgang abbrechen möchten, klicken Sie auf Cancel (Abbrechen).

Löschen eines Benutzers mit dem AWS CLI

So löschen Sie einen Benutzer mit dem CLI

- Verwenden Sie den Befehl [delete-user](#), um einen Benutzer zu löschen.

Das Konto wird gelöscht und aus allen Zugriffskontrolllisten entfernt, zu denen es gehört. Im Folgenden wird ein Beispiel gezeigt.

Für Linux, macOS oder Unix:

```
aws memorydb delete-user \  
--user-name user-name-2
```

Für Windows:

```
aws memorydb delete-user ^  
--user-name user-name-2
```

Verwaltung von Zugriffskontrolllisten mit der Konsole und CLI

Sie können Zugriffskontrolllisten erstellen, um den Zugriff von Benutzern auf einen oder mehrere Cluster zu organisieren und zu kontrollieren, wie im Folgenden gezeigt.

Gehen Sie wie folgt vor, um Zugriffskontrolllisten mithilfe der Konsole zu verwalten.


```
--user-names "user-name-1" "user-name-2"
```

Für Windows:

```
aws memorydb create-acl ^  
--acl-name "new-acl-1" ^  
--user-names "user-name-1" "user-name-2"
```

Ändern einer Zugriffskontrollliste (ACL) (Konsole)

Um eine Zugriffskontrollliste mit der Konsole zu ändern

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Access Control Lists (ACL) aus.
3. Wählen Sie die ACL aus, die Sie ändern möchten, und klicken Sie dann auf Ändern
4. Führen Sie auf der Seite Ändern unter Ausgewählte Benutzer einen der folgenden Schritte aus:
 - a. Erstellen Sie einen neuen Benutzer, indem Sie Benutzer erstellen wählen, der der ACL hinzugefügt werden soll.
 - b. Fügen Sie Benutzer hinzu oder entfernen Sie sie, indem Sie „Verwalten“ wählen und dann im Dialogfeld „Benutzer verwalten“ Benutzer auswählen oder deren Auswahl aufheben und dann „Auswählen“ wählen.
5. Geben Sie auf der Seite Zugriffskontrollliste (ACL) erstellen einen ACL-Namen ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
6. Führen Sie unter Ausgewählte Benutzer einen der folgenden Schritte aus:
 - a. Erstellen Sie einen neuen Benutzer, indem Sie Benutzer erstellen wählen
 - b. Fügen Sie Benutzer hinzu, indem Sie Verwalten und dann Benutzer im Dialogfeld Benutzer verwalten auswählen und dann Auswählen auswählen.

7. Wählen Sie Ändern, um Ihre Änderungen zu speichern, oder Abbrechen, um sie zu verwerfen.

Ändern einer Zugriffskontrollliste (ACL) mithilfe der AWS CLI

So ändern Sie eine ACL, indem Sie neue Benutzer hinzufügen oder aktuelle Mitglieder mit der CLI entfernen

- Verwenden Sie den Befehl [update-acl](#), um eine ACL zu ändern.

Für Linux, macOS oder Unix:

```
aws memorydb update-acl --acl-name new-acl-1 \  
--user-names-to-add user-name-3 \  
--user-names-to-remove user-name-2
```

Für 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

Alle offenen Verbindungen eines Benutzers, der aus einer ACL entfernt wurde, werden mit diesem Befehl beendet.

Details zur Zugriffskontrollliste (ACL) anzeigen (Konsole)

Um ACL-Details auf der Konsole anzuzeigen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Access Control Lists (ACL) aus.
3. Wählen Sie die ACL unter ACL-Name aus oder verwenden Sie das Suchfeld, um die ACL zu finden.
4. Unter Benutzer können Sie die Liste der Benutzer überprüfen, die der ACL zugeordnet sind.

5. Unter Assoziierte Cluster können Sie überprüfen, zu welchem Cluster die ACL gehört.
6. Unter Tags können Sie alle mit der ACL verknüpften Tags überprüfen.

Zugriffskontrolllisten (ACL) anzeigen mit dem AWS CLI

Verwenden Sie den Befehl [describe-acls](#), um Details einer ACL anzuzeigen.

```
aws memorydb describe-acls \  
  --acl-name test-group
```

Löschen einer Zugriffskontrollliste (ACL) (Konsole)

Um Zugriffskontrolllisten mit der Konsole zu löschen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/memorydb/>.](#)
2. Wählen Sie im linken Navigationsbereich Access Control Lists (ACL) aus.
3. Wählen Sie die ACL aus, die Sie ändern möchten, und klicken Sie dann auf Löschen
4. Geben delete Sie auf der Seite Löschen das Bestätigungsfeld ein und wählen Sie Löschen oder Abbrechen, um das Löschen der ACL zu verhindern.

Die ACL selbst, nicht die Benutzer, die zur Gruppe gehören, wird gelöscht.

Löschen einer Zugriffskontrollliste (ACL) mit dem AWS CLI

So löschen Sie eine ACL mit der CLI

- Verwenden Sie den Befehl [delete-acl](#), um eine ACL zu löschen.

Für Linux, macOS oder Unix:

```
aws memorydb delete-acl /  
  --acl-name
```

Für Windows:

```
aws memorydb delete-acl ^  
  --acl-name
```

Die vorhergehenden Beispiele geben die folgende Antwort zurück.

```
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"
}
```

Zuweisung von Zugriffskontrolllisten zu Clustern

Nachdem Sie eine ACL erstellt und Benutzer hinzugefügt haben, besteht der letzte Schritt bei der Implementierung von ACLs darin, die ACL einem Cluster zuzuweisen.

Zuweisung von Zugriffskontrolllisten zu Clustern mithilfe der Konsole

Informationen zum Hinzufügen einer ACL zu einem Cluster mithilfe der finden Sie AWS Management Console unter [Einen MemoryDB-Cluster erstellen](#).

Zuweisen von Zugriffskontrolllisten zu Clustern mithilfe der AWS CLI

Der folgende AWS CLI Vorgang erstellt einen Cluster mit aktivierter Verschlüsselung bei der Übertragung (TLS) und dem `acl-name` Parameter mit dem Wert `my-acl-name`. Ersetzen Sie die Subnetzgruppe `subnet-group` durch eine vorhandene Subnetzgruppe.

Hauptparameter

- **--engine-version**— Muss 6.2 sein.
- **--tls-enabled**— Wird für die Authentifizierung und für die Zuordnung einer ACL verwendet.
- **--acl-name**— Dieser Wert stellt Zugriffskontrolllisten bereit, die sich aus Benutzern mit bestimmten Zugriffsberechtigungen für den Cluster zusammensetzen.

Für Linux, macOS oder 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"
```

Für 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"
```

Der folgende AWS CLI Vorgang ändert einen Cluster mit aktivierter Verschlüsselung bei der Übertragung (TLS) und dem `acl-name` Parameter mit dem Wert `new-acl-2`.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name cluster-1 \  
  --acl-name "new-acl-2"
```

Für Windows:

```
aws memorydb update-cluster ^  
  --cluster-name cluster-1 ^  
  --acl-name "new-acl-2"
```

Authentifizieren mit IAM

Themen

- [Übersicht](#)

- [Einschränkungen](#)
- [Setup](#)
- [Herstellen von Verbindungen](#)

Übersicht

Mit der IAM-Authentifizierung können Sie eine Verbindung mit MemoryDB mithilfe von AWS IAM-Identitäten authentifizieren, wenn Ihr Cluster für die Verwendung der Redis-Version 7 oder höher konfiguriert ist. Auf diese Weise können Sie Ihr Sicherheitsmodell stärken und viele administrative Sicherheitsaufgaben vereinfachen. Mit der IAM-Authentifizierung können Sie eine detaillierte Zugriffskontrolle für jeden einzelnen MemoryDB-Cluster und jeden MemoryDB-Benutzer konfigurieren und den Prinzipien der geringsten Berechtigung folgen. Die IAM-Authentifizierung für MemoryDB Redis funktioniert, indem ein kurzlebiges IAM-Authentifizierungstoken anstelle eines langlebigen MemoryDB-Benutzerpassworts im Redis-Befehl bereitgestellt wird. AUTH HELLO Weitere Informationen zum IAM-Authentifizierungstoken finden Sie im Allgemeinen Referenzhandbuch von AWS beim [Signiervorgang für Signature Version 4](#) und im folgenden Codebeispiel.

Sie können IAM-Identitäten und die zugehörigen Richtlinien verwenden, um den Redis-Zugriff weiter einzuschränken. Sie können Benutzern von ihren Verbundidentitätsanbietern auch direkt Zugriff auf MemoryDB-Cluster gewähren.

Um AWS IAM mit MemoryDB zu verwenden, müssen Sie zuerst einen MemoryDB-Benutzer erstellen, bei dem der Authentifizierungsmodus auf IAM festgelegt ist. Anschließend können Sie eine IAM-Identität erstellen oder wiederverwenden. Die IAM-Identität benötigt eine zugehörige Richtlinie, um die `memorydb:Connect` Aktion dem MemoryDB-Cluster und dem MemoryDB-Benutzer zuzuweisen. Nach der Konfiguration können Sie ein IAM-Authentifizierungstoken mit den AWS-Anmeldeinformationen des IAM-Benutzers oder der IAM-Rolle erstellen. Schließlich müssen Sie das kurzlebige IAM-Authentifizierungstoken als Passwort in Ihrem Redis-Client angeben, wenn Sie eine Verbindung mit Ihrem MemoryDB-Clusterknoten herstellen. Ein Redis-Client mit Unterstützung für Anbieter von Anmeldeinformationen kann die temporären Anmeldeinformationen automatisch für jede neue Verbindung generieren. MemoryDB führt die IAM-Authentifizierung für Verbindungsanfragen von IAM-fähigen MemoryDB-Benutzern durch und validiert die Verbindungsanfragen mit IAM.

Einschränkungen

Bei der Verwendung der IAM-Authentifizierung gelten die folgenden Einschränkungen:

- Die IAM-Authentifizierung ist verfügbar, wenn Sie die Redis-Engine Version 7.0 oder höher verwenden.
- Das IAM-Authentifizierungstoken ist für 15 Minuten gültig. Für langlebige Verbindungen empfehlen wir die Verwendung eines Redis-Clients, der eine Schnittstelle zu einem Anbieter von Anmeldeinformationen unterstützt.
- Eine IAM-authentifizierte Verbindung mit MemoryDB wird nach 12 Stunden automatisch getrennt. Die Verbindung kann um 12 Stunden verlängert werden, indem ein AUTH- oder HELLO-Befehl mit einem neuen IAM-Authentifizierungstoken gesendet wird.
- Die IAM-Authentifizierung wird in MULTI EXEC-Befehlen nicht unterstützt.
- Derzeit unterstützt die IAM-Authentifizierung nicht alle globalen Bedingungskontextschlüssel. Weitere Informationen über globale Bedingungskontextschlüssel finden Sie unter [Globale Bedingungskontextschlüssel](#) finden Sie unter [AWS Globale Bedingungskontextschlüssel](#) finden Sie unter Globale Bedingungskontextschlüssel, finden Sie unter

Setup

So richten Sie die IAM-Authentifizierung ein:

1. Erstellen eines -Clusters

```
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. Erstellen Sie, wie unten dargestellt, ein Dokument mit den IAM-Vertrauensrichtlinien für Ihre Rolle, sodass Ihr Konto die neue Rolle übernehmen kann. Speichern Sie die Richtlinie in einer Datei namens trust-policy.json.

```
{  
  "Version": "2012-10-17",  
  "Statement": {  
    "Effect": "Allow",  
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },  
    "Action": "sts:AssumeRole"  
  }  
}
```

```
}

```

- Erstellen Sie ein IAM-Richtliniendokument wie im Folgenden dargestellt. Speichern Sie die Richtlinie in einer Datei namens `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"
      ]
    }
  ]
}
```

- Erstellen Sie eine IAM-Rolle.

```
aws iam create-role \
  --role-name "memorydb-iam-auth-app" \
  --assume-role-policy-document file://trust-policy.json
```

- Erstellen Sie die IAM-Richtlinie.

```
aws iam create-policy \
  --policy-name "memorydb-allow-all" \
  --policy-document file://policy.json
```

- Fügen Sie die IAM-Richtlinie an die Rolle an.

```
aws iam attach-role-policy \
  --role-name "memorydb-iam-auth-app" \
  --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

- Erstellen Sie einen neuen IAM-fähigen Benutzer.

```
aws memorydb create-user \
  --user-name iam-user-01 \
```

```
--authentication-mode Type=iam \  
--access-string "on ~* +@all"
```

8. Erstellen Sie eine

```
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
```

Herstellen von Verbindungen

Verbinden mit Token als Passwort

Sie müssen zuerst das kurzlebige IAM-Authentifizierungstoken mithilfe einer [vorab signierten AWS-SigV4-Anfrage](#) generieren. Danach geben Sie, wie im folgenden Beispiel dargestellt, das IAM-Authentifizierungstoken als Passwort an, wenn Sie eine Verbindung mit einem MemoryDB-Cluster herstellen.

```
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();
```

Im Folgenden finden Sie die Definition für `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());
}
}

```

Verbinden mit Anbieter von Anmeldeinformationen

Der folgende Code zeigt, wie Sie sich mit MemoryDB mithilfe des Anbieters für Anmeldeinformationen zur IAM-Authentifizierung authentifizieren.

```

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();
```

Im Folgenden finden Sie ein Beispiel für einen Lettuce-Redis-Cluster-Client, der IAM AuthTokenRequest in einen Anbieter von Anmeldeinformationen verpackt, um bei Bedarf automatisch temporäre Anmeldeinformationen zu generieren.

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

Identitäts- und Zugriffsmanagement in MemoryDB für Redis

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um MemoryDB-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- [Zielgruppe](#)
- [Authentifizierung mit Identitäten](#)
- [Verwalten des Zugriffs mit Richtlinien](#)
- [Wie funktioniert MemoryDB for Redis mit IAM](#)
- [Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)
- [Fehlerbehebung bei Identität und Zugriff auf MemoryDB for Redis](#)
- [Zugriffskontrolle](#)
- [Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre MemoryDB-Ressourcen](#)

Zielgruppe

Die Art und Weise, wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in MemoryDB ausführen.

Dienstbenutzer — Wenn Sie den MemoryDB-Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr MemoryDB-Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Wenn Sie auf eine Funktion in MemoryDB nicht zugreifen können, finden Sie weitere Informationen unter.

[Fehlerbehebung bei Identität und Zugriff auf MemoryDB for Redis](#)

Dienstadministrator — Wenn Sie in Ihrem Unternehmen für die MemoryDB-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf MemoryDB. Es ist Ihre Aufgabe, zu bestimmen, auf welche Funktionen und Ressourcen von MemoryDB Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM mit MemoryDB verwenden kann, finden Sie unter. [Wie funktioniert MemoryDB for Redis mit IAM](#)

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien zur Verwaltung des Zugriffs auf MemoryDB schreiben können. Beispiele für identitätsbasierte MemoryDB-Richtlinien, die Sie in IAM verwenden können, finden Sie unter.

[Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich mit Ihren Identitätsdaten anmelden. AWS Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter [So melden Sie sich bei Ihrem an AWS-Konto](#) im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, mit denen Sie Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch signieren können. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu [signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen](#) signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter [Multi-Faktor-Authentifizierung](#) im AWS IAM Identity Center - Benutzerhandbuch und [Verwenden der Multi-Faktor-Authentifizierung \(MFA\) in AWS](#) im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter [Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine

Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter [Was ist IAM Identity Center?](#) im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein [IAM-Benutzer](#) ist eine Identität innerhalb Ihres Unternehmens AWS-Konto, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter [Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Eine [IAM-Gruppe](#) ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter [Erstellen eines IAM-Benutzers \(anstatt einer Rolle\)](#) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine [IAM-Rolle](#) ist eine Identität innerhalb Ihres Unternehmens AWS-Konto, die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen [wechseln](#). Sie können eine Rolle übernehmen, indem Sie eine AWS

CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter [Verwenden von IAM-Rollen](#) im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- **Verbundbenutzerzugriff** – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.
- **Temporäre IAM-Benutzerberechtigungen** – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- **Kontoübergreifender Zugriff** – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zum Unterschied zwischen Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie unter [Kontenübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch](#).
- **Serviceübergreifender Zugriff** — Einige verwenden Funktionen in anderen. AWS-Services AWS-Services Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
- **Forward Access Sessions (FAS)** — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über

Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).

- **Servicerolle** – Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Auf Amazon EC2 ausgeführte Anwendungen** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter [Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden](#) im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter [Erstellen einer IAM-Rolle \(anstatt eines Benutzers\)](#) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter [Übersicht über JSON-Richtlinien](#) im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console, der AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter [Auswahl zwischen verwalteten und eingebundenen Richtlinien](#) im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen

in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter [Zugriffskontrollliste \(ACL\) – Übersicht](#) (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- **Berechtigungsgrenzen** – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter [Berechtigungsgrenzen für IAM-Entitäten](#) im IAM-Benutzerhandbuch.
- **Service Control Policies (SCPs)** — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos

Weitere Informationen zu Organizations und SCPs finden Sie unter [Funktionsweise von SCPs](#) im AWS Organizations -Benutzerhandbuch.

- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter [Sitzungsrichtlinien](#) im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter [Bewertungslogik für Richtlinien](#).

Wie funktioniert MemoryDB for Redis mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf MemoryDB zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen für die Verwendung mit MemoryDB verfügbar sind.

IAM-Funktionen, die Sie mit MemoryDB for Redis verwenden können

IAM-Feature	MemoryDB-Unterstützung
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Ja

IAM-Feature	MemoryDB-Unterstützung
ABAC (Tags in Richtlinien)	Ja
Temporäre Anmeldeinformationen	Ja
Hauptberechtigungen	Ja
Servicerollen	Ja
Service-verknüpfte Rollen	Ja

Einen allgemeinen Überblick darüber, wie MemoryDB und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter [AWS Dienste, die mit IAM funktionieren](#).

Identitätsbasierte Richtlinien für MemoryDB

Unterstützt Richtlinien auf Identitätsbasis.	Ja
--	----

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der [IAM-Referenz für JSON-Richtlinienelemente](#) im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für MemoryDB

Beispiele für identitätsbasierte MemoryDB-Richtlinien finden Sie unter [Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)

Ressourcenbasierte Richtlinien in MemoryDB

Unterstützt ressourcenbasierte Richtlinien	Nein
--	------

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM-Administrator des vertrauenswürdigen Kontos auch der Prinzipalentsität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter [Kontenübergreifender Ressourcenzugriff in IAM](#) im IAM-Benutzerhandbuch.

Richtlinienaktionen für MemoryDB

Unterstützt Richtlinienaktionen	Ja
---------------------------------	----

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang

gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der MemoryDB-Aktionen finden Sie unter [Von MemoryDB for Redis definierte Aktionen in der Service Authorization Reference](#).

Richtlinienaktionen in MemoryDB verwenden vor der Aktion das folgende Präfix:

```
MemoryDB
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [  
  "MemoryDB:action1",  
  "MemoryDB:action2"  
]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort Describe beginnen, einschließlich der folgenden Aktion:

```
"Action": "MemoryDB:Describe*"
```

Beispiele für identitätsbasierte MemoryDB-Richtlinien finden Sie unter [Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)

Richtlinienressourcen für MemoryDB

Unterstützt Richtlinienressourcen

Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein `Resource` oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen [Amazon-Ressourcennamen \(ARN\)](#) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*" 
```

Eine Liste der MemoryDB-Ressourcentypen und ihrer ARNs finden Sie unter [Von MemoryDB for Redis definierte Ressourcen in der Service Authorization Reference](#). Informationen dazu, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter [Von MemoryDB für Redis definierte Aktionen](#).

Beispiele für identitätsbasierte MemoryDB-Richtlinien finden Sie unter [Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)

Bedingungsschlüssel für Richtlinien für MemoryDB

Unterstützt servicespezifische Richtlinienbedingungsschlüssel	Ja
---	----

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition block`) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die [Bedingungsoperatoren](#) verwenden, z. B. `ist gleich` oder `kleiner als`, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen AND-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, wertet die

Bedingung mithilfe einer logischen OR Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter [IAM-Richtlinienelemente: Variablen und Tags](#) im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter [Kontextschlüssel für AWS globale Bedingungen](#) im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte MemoryDB-Richtlinien finden Sie unter [Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis](#)

Verwenden von Bedingungsschlüssel

Sie können Bedingungen angeben, die bestimmen, wie eine IAM-Richtlinie wirksam wird. In MemoryDB können Sie das Condition Element einer JSON-Richtlinie verwenden, um Schlüssel im Anforderungskontext mit Schlüsselwerten zu vergleichen, die Sie in Ihrer Richtlinie angeben. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#).

Eine Liste der MemoryDB-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für [MemoryDB for Redis in der Service Authorization Reference](#).

Eine Liste der globalen Bedingungsschlüssel finden Sie unter [AWS Globale Bedingungskontextschlüssel](#).

Festlegung von Bedingungen: Verwenden von Bedingungsschlüsseln

Um eine differenzierte Steuerung zu implementieren, können Sie eine IAM-Berechtigungsrichtlinie schreiben, die Bedingungen für die Steuerung einzelner Parameter bei bestimmten Anfragen festlegt. Anschließend können Sie die Richtlinie auf IAM-Benutzer, -Gruppen oder -Rollen anwenden, die Sie mit der IAM-Konsole erstellen.

Um eine Bedingung anzuwenden, fügen Sie die Bedingungsinformationen der IAM-Richtlinienanweisung hinzu. Um beispielsweise die Erstellung eines MemoryDB-Clusters mit deaktiviertem TLS zu verbieten, können Sie in Ihrer Richtlinienerklärung die folgende Bedingung angeben.

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

```

"Statement": [
  {
    "Effect": "Deny",
    "Action": [
      "memorydb:CreateCluster"
    ],
    "Resource": [
      "*"
    ],
    "Condition": {
      "Bool": {
        "memorydb:TLSEnabled": "false"
      }
    }
  }
]
}

```

Weitere Informationen zum Tagging finden Sie unter [Markierung Ihrer MemoryDB-Ressourcen](#)

Weitere Informationen zur Verwendung von Richtlinienbedingungsoperatoren finden Sie unter [MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen](#).

Beispielrichtlinien: Verwenden von Bedingungen für die differenzierte Parameterkontrolle

In diesem Abschnitt werden Beispielrichtlinien für die Implementierung einer differenzierten Zugriffskontrolle für die zuvor aufgeführten MemoryDB-Parameter beschrieben.

1. memoryDB:TlsEnabled — Geben Sie an, dass Cluster nur mit aktiviertem TLS erstellt werden.

```

{
  "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`: — Geben Sie an, dass die Benutzer mit einem bestimmten Authentifizierungsmodus (z. B. IAM) erstellt werden können.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "memorydb:Createuser"
      ],
      "Resource": [
        "arn:aws:memorydb:*:*:user/*"
      ],
      "Condition": {
        "StringEquals": {
          "memorydb:UserAuthenticationMode": "iam"
        }
      }
    }
  ]
}

```

In Fällen, in denen Sie auf „Verweigern“ basierende Richtlinien festlegen, wird empfohlen, den [StringEqualsIgnoreCase](#) Operator zu verwenden, um unabhängig vom jeweiligen Fall alle Anrufe mit einem bestimmten Benutzerauthentifizierungsmodus zu vermeiden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "memorydb:CreateUser"
      ],
      "Resource": "*",
      "Condition": {
        "StringEqualsIgnoreCase": {
          "memorydb:UserAuthenticationMode": "password"
        }
      }
    }
  ]
}
```

Zugriffskontrolllisten (ACLs) in MemoryDB

Unterstützt ACLs

Ja

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Attributbasierte Zugriffskontrolle (ABAC) mit MemoryDB

Unterstützt ABAC (Tags in Richtlinien)

Ja

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In werden diese Attribute als Tags AWS bezeichnet. Sie

können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungelement einer [Richtlinie Tag-Informationen](#) an, indem Sie die Schlüssel `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter [Was ist ABAC?](#) im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe [Attributbasierte Zugriffskontrolle \(ABAC\)](#) verwenden im IAM-Benutzerhandbuch.

Temporäre Anmeldeinformationen mit MemoryDB verwenden

Unterstützt temporäre Anmeldeinformationen	Ja
--	----

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services [funktionieren AWS-Services](#), [finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM](#).

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter [Wechseln zu einer Rolle \(Konsole\)](#) im IAM-Benutzerhandbuch.

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden

AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter [Temporäre Sicherheitsanmeldeinformationen in IAM](#).

Serviceübergreifende Prinzipalberechtigungen für MemoryDB

Unterstützt Forward Access Sessions (FAS)	Ja
---	----

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).

Servicerollen für MemoryDB

Unterstützt Servicerollen	Ja
---------------------------	----

Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.

Warning

Durch das Ändern der Berechtigungen für eine Servicerolle kann die MemoryDB-Funktionalität beeinträchtigt werden. Bearbeiten Sie Servicerollen nur, wenn MemoryDB Sie dazu anleitet.

Dienstbezogene Rollen für MemoryDB

Unterstützt serviceverknüpfte Rollen Ja

Eine dienstverknüpfte Rolle ist eine Art von Servicerolle, die mit einer Service-Verknüpfung ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter [AWS -Services, die mit IAM funktionieren](#). Suchen Sie in der Tabelle nach einem Service mit einem Yes in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

Beispiele für identitätsbasierte Richtlinien für MemoryDB for Redis

Standardmäßig sind Benutzer und Rollen nicht berechtigt, MemoryDB-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Einzelheiten zu den von MemoryDB definierten Aktionen und Ressourcentypen, einschließlich des Formats der ARNs für jeden Ressourcentyp, finden Sie unter [Aktionen, Ressourcen und Bedingungsschlüssel für MemoryDB for Redis in der Service Authorization Reference](#).

Themen

- [Bewährte Methoden für Richtlinien](#)
- [Verwenden der MemoryDB-Konsole](#)
- [Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer](#)

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand MemoryDB-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) oder [AWS -verwaltete Richtlinien für Auftrags-Funktionen](#) im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter [Richtlinien und Berechtigungen in IAM](#) im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter [Richtlinienvvalidierung zum IAM Access Analyzer](#) im IAM-Benutzerhandbuch.

- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter [Konfigurieren eines MFA-geschützten API-Zugriffs](#) im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter [Bewährte Methoden für die Sicherheit in IAM](#) im IAM-Benutzerhandbuch.

Verwenden der MemoryDB-Konsole

Um auf die MemoryDB for Redis-Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den MemoryDB-Ressourcen in Ihrem aufzulisten und anzuzeigen. AWS-Konto Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. AWS Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die MemoryDB-Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die MemoryDB ConsoleAccess - oder ReadOnly AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter [Hinzufügen von Berechtigungen zu einem Benutzer](#) im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der OR-API. AWS CLI 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": "*"
  }
]
}

```

Fehlerbehebung bei Identität und Zugriff auf MemoryDB for Redis

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit MemoryDB und IAM auftreten können.

Themen

- [Ich bin nicht berechtigt, eine Aktion in MemoryDB auszuführen](#)
- [Ich bin nicht berechtigt, iam auszuführen: PassRole](#)
- [Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine MemoryDB-Ressourcen ermöglichen](#)

Ich bin nicht berechtigt, eine Aktion in MemoryDB auszuführen

Wenn Ihnen AWS Management Console mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion auszuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort bereitgestellt hat.

Der folgende Beispielfehler tritt auf, wenn der `mateojackson`-Benutzer versucht, die Konsole zum Anzeigen von Details zu einer fiktiven `my-example-widget`-Ressource zu verwenden, jedoch nicht über MemoryDB: `GetWidget`-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
MemoryDB: GetWidget on resource: my-example-widget
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion `my-example-widget` auf die Ressource MemoryDB: `GetWidget` zugreifen zu können.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht autorisiert sind, die `iam:PassRole` Aktion auszuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an MemoryDB übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in MemoryDB auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren Administrator. AWS Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine MemoryDB-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob MemoryDB diese Funktionen unterstützt, finden Sie unter [Wie funktioniert MemoryDB for Redis mit IAM](#)
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter [Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.](#)
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie [AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.](#)
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter [Gewähren von Zugriff für extern authentifizierte Benutzer \(Identitätsverbund\)](#) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie im IAM-Benutzerhandbuch unter [Kontenübergreifender Ressourcenzugriff in IAM.](#)

Zugriffskontrolle

Sie können über gültige Anmeldeinformationen verfügen, um Ihre Anfragen zu authentifizieren, aber ohne die entsprechenden Berechtigungen können Sie keine MemoryDB for Redis-Ressourcen erstellen oder darauf zugreifen. Sie benötigen beispielsweise Berechtigungen, um einen MemoryDB-Cluster zu erstellen.

In den folgenden Abschnitten wird beschrieben, wie Sie Berechtigungen für MemoryDB for Redis verwalten. Wir empfehlen Ihnen, zunächst die Übersicht zu lesen.

- [Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre MemoryDB-Ressourcen](#)
- [Verwenden von identitätsbasierten Richtlinien \(IAM-Richtlinien\) für MemoryDB for Redis](#)

Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre MemoryDB-Ressourcen

Jede AWS Ressource gehört einem AWS Konto, und die Berechtigungen zum Erstellen oder Zugreifen auf eine Ressource werden durch Berechtigungsrichtlinien geregelt. Ein Kontoadministrator kann Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anfügen. Darüber hinaus unterstützt MemoryDB for Redis auch das Anhängen von Berechtigungsrichtlinien an Ressourcen.

Note

Ein Kontoadministrator (oder Administratorbenutzer) ist ein Benutzer mit Administratorrechten. Weitere Informationen finden Sie unter [Bewährte Methoden für IAM](#) im IAM-Benutzerhandbuch.

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in: AWS IAM Identity Center

Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter [Erstellen eines Berechtigungssatzes](#) im AWS IAM Identity Center -Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter [Erstellen einer Rolle für einen externen Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch.

- IAM-Benutzer:

- Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter [Erstellen einer Rolle für einen IAM-Benutzer](#) im IAM-Benutzerhandbuch.
- (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter [Hinzufügen von Berechtigungen zu einem Benutzer \(Konsole\)](#) im IAM-Benutzerhandbuch.

Themen

- [Ressourcen und Operationen von MemoryDB für Redis](#)
- [Grundlegendes zum Eigentum an Ressourcen](#)

- [Verwalten des Zugriffs auf Ressourcen](#)
- [Verwenden von identitätsbasierten Richtlinien \(IAM-Richtlinien\) für MemoryDB for Redis](#)
- [Berechtigungen auf Ressourcenebene](#)
- [Verwenden von serviceverknüpften Rollen für Amazon MemoryDB for Redis](#)
- [AWSverwaltete Richtlinien für MemoryDB for Redis](#)
- [MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen](#)

Ressourcen und Operationen von MemoryDB für Redis

In MemoryDB for Redis ist die primäre Ressource ein Cluster.

Diesen Ressourcen sind eindeutige Amazon-Ressourcennamen (ARN) zugeordnet, wie im Folgenden gezeigt.

Note

Damit Berechtigungen auf Ressourcenebene wirksam sind, sollte der Ressourcenname in der ARN-Zeichenfolge in Kleinbuchstaben geschrieben werden.

Ressourcentyp	ARN-Format
Benutzer	<i>arn:aws:memorydb: us-east-1:123456789012:user/user1</i>
Zugriffskontrollliste (ACL)	<i>arn:aws:memorydb: us-east-1:123456789012:acl/myacl</i>
Cluster	<i>arn:aws:memorydb: us-east-1:123456789012:cluster/mein-Cluster</i>
Snapshot	<i>arn:aws:memorydb: us-east-1:123456789012:snapshot/mein-snapshot</i>

Ressourcentyp	ARN-Format
Parametergruppe	<i>arn:aws:memorydb: us-east-1:123456789012:parametergruppe/my-parameter-group</i>
Subnetzgruppe	<i>arn:aws:memorydb: us-east-1:123456789012:subnetzgruppe/my-subnet-group</i>

MemoryDB bietet eine Reihe von Operationen für die Arbeit mit MemoryDB-Ressourcen. [Eine Liste der verfügbaren Operationen finden Sie unter MemoryDB for Redis Actions.](#)

Grundlegendes zum Eigentum an Ressourcen

Ein Ressourcenbesitzer ist das AWS Konto, das die Ressource erstellt hat. Das heißt, der Ressourcenbesitzer ist das AWS Konto der Prinzipalidentität, die die Anforderung authentifiziert, mit der die Ressource erstellt wird. Eine Prinzipalidentität kann das Root-Konto, ein IAM-Benutzer oder eine IAM-Rolle sein. Die Funktionsweise wird anhand der folgenden Beispiele deutlich:

- Angenommen, Sie verwenden die Root-Kontoanmeldeinformationen Ihres AWS Kontos, um einen Cluster zu erstellen. In diesem Fall ist Ihr AWS Konto der Eigentümer der Ressource. In MemoryDB ist die Ressource der Cluster.
- Angenommen, Sie erstellen in Ihrem AWS Konto einen IAM-Benutzer und gewähren diesem Benutzer die Erlaubnis, einen Cluster zu erstellen. In diesem Fall kann der Benutzer einen Cluster erstellen. Ihr AWS Konto, zu dem der Benutzer gehört, besitzt jedoch die Clusterressource.
- Angenommen, Sie erstellen in Ihrem AWS Konto eine IAM-Rolle mit Berechtigungen zum Erstellen eines Clusters. In diesem Fall kann jeder, der die Rolle übernehmen kann, einen Cluster erstellen. Ihr AWS Konto, zu dem die Rolle gehört, besitzt die Clusterressource.

Verwalten des Zugriffs auf Ressourcen

Eine Berechtigungsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Im folgenden Abschnitt werden die verfügbaren Optionen zum Erstellen von Berechtigungsrichtlinien erläutert.

Note

In diesem Abschnitt wird die Verwendung von IAM im Kontext von MemoryDB for Redis beschrieben. Er enthält keine detaillierten Informationen über den IAM-Service. Eine umfassende IAM-Dokumentation finden Sie unter [Was ist IAM?](#) im IAM-Benutzerhandbuch. Für Informationen über die Syntax und Beschreibungen von [AWS -IAM-Richtlinien](#) lesen Sie die IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

An eine IAM-Identität angefügte Richtlinien werden als identitätsbasierte Richtlinien (oder IAM-Richtlinien) bezeichnet. An Ressourcen angehängte Richtlinien werden als ressourcenbasierte Richtlinien bezeichnet.

Themen

- [Identitätsbasierte Richtlinien \(IAM-Richtlinien\)](#)
- [Angaben der Richtlinienelemente: Aktionen, Effekte, Ressourcen und Prinzipale](#)
- [Angaben von Bedingungen in einer Richtlinie](#)

Identitätsbasierte Richtlinien (IAM-Richtlinien)

Richtlinien können IAM-Identitäten angefügt werden. Sie können z. B. Folgendes tun:

- Anfügen einer Berechtigungsrichtlinie zu einem Benutzer oder einer Gruppe in Ihrem Konto – Ein Kontoadministrator kann eine Berechtigungsrichtlinie verwenden, die einem bestimmten Benutzer zugeordnet ist, um Berechtigungen zu erteilen. In diesem Fall sind die Berechtigungen für diesen Benutzer vorgesehen, um eine MemoryDB-Ressource zu erstellen, z. B. einen Cluster, eine Parametergruppe oder eine Sicherheitsgruppe.
- Einer Rolle eine Berechtigungsrichtlinie zuweisen (kontoübergreifende Berechtigungen gewähren) – Sie können einer IAM-Rolle eine identitätsbasierte Berechtigungsrichtlinie zuweisen, um kontoübergreifende Berechtigungen zu erteilen. Der Administrator in Konto A kann beispielsweise wie folgt eine Rolle erstellen, um einem anderen Konto (z. B. AWS Konto B) oder einem AWS Dienst kontoübergreifende Berechtigungen zu gewähren:
 1. Der Administrator von Konto A erstellt eine IAM-Rolle und fügt ihr eine Berechtigungsrichtlinie an, die Berechtigungen für Ressourcen in Konto A erteilt.
 2. Der Administrator von Konto A weist der Rolle eine Vertrauensrichtlinie zu, die Konto B als den Prinzipal identifiziert, der die Rolle übernehmen kann.

3. Der Administrator von Konto B kann dann die Berechtigungen zur Übernahme der Rolle an alle Benutzer in Konto B delegieren. Auf diese Weise können Benutzer in Konto B Ressourcen in Konto A erstellen oder darauf zugreifen. In einigen Fällen möchten Sie einem AWS Dienst möglicherweise Berechtigungen zur Übernahme der Rolle erteilen. Zum Support dieses Ansatzes kann es sich beim Prinzipal in der Vertrauensrichtlinie auch um einen AWS -Service-Prinzipal handeln.

Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter [Zugriffsverwaltung](#) im IAM-Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für eine Richtlinie, die es einem Benutzer ermöglicht, die DescribeClusters Aktion für Ihr AWS Konto durchzuführen. MemoryDB unterstützt auch die Identifizierung bestimmter Ressourcen mithilfe der Ressourcen-ARNs für API-Aktionen. (Dieser Ansatz heißt auch Ressourcenebenen-Berechtigungen.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "DescribeClusters",
    "Effect": "Allow",
    "Action": [
      "memorydb:DescribeClusters"],
    "Resource": resource-arn
  ]
}
```

Weitere Informationen zur Verwendung identitätsbasierter Richtlinien mit MemoryDB finden Sie unter [Verwenden von identitätsbasierten Richtlinien \(IAM-Richtlinien\) für MemoryDB for Redis](#) Weitere Informationen zu Benutzern, Gruppen, Rollen und Berechtigungen finden Sie unter [Identitäten \(Benutzer, Gruppen und Rollen\)](#) im IAM-Benutzerhandbuch.

Angeben der Richtlinienelemente: Aktionen, Effekte, Ressourcen und Prinzipale

[Für jede MemoryDB for Redis-Ressource \(siehe Ressourcen und Operationen von MemoryDB für Redis\) definiert der Dienst eine Reihe von API-Operationen \(siehe Aktionen\).](#) Um Berechtigungen für diese API-Operationen zu gewähren, definiert MemoryDB eine Reihe von Aktionen, die Sie in einer Richtlinie angeben können. Für die MemoryDB-Clusterressource sind beispielsweise die

folgenden Aktionen definiert: `CreateCluster`, und `DeleteCluster` `DescribeClusters` Für das Durchführen einer API-Operation können Berechtigungen für mehrere Aktionen erforderlich sein.

Grundlegende Richtlinienelemente:

- **Ressource** – In einer Richtlinie wird der Amazon-Ressourcenname (ARN) zur Identifizierung der Ressource verwendet, für die die Richtlinie gilt. Weitere Informationen finden Sie unter [Ressourcen und Operationen von MemoryDB für Redis](#).
- **Aktion** – Mit Aktionsschlüsselwörtern geben Sie die Ressourcenoperationen an, die Sie zulassen oder verweigern möchten. Je nach Angabe gewährt oder verweigert die `memorydb:CreateCluster` Berechtigung dem Benutzer beispielsweise die Berechtigung `Effect`, den Vorgang MemoryDB for Redis auszuführen. `CreateCluster`
- **Auswirkung** – Die von Ihnen festgelegte Auswirkung, wenn der Benutzer die jeweilige Aktion anfordert – entweder „allow“ (Zugriffserlaubnis) oder „deny“ (Zugriffsverweigerung). Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten ("Allow"), wird er automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So können Sie zum Beispiel sicherstellen, dass ein Benutzer nicht auf die Ressource zugreifen kann, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird.
- **Prinzipal** – In identitätsbasierten Richtlinien (IAM-Richtlinien) ist der Benutzer, dem die Richtlinie zugewiesen ist, automatisch der Prinzipal. In ressourcenbasierten Richtlinien müssen Sie den Benutzer, das Konto, den Service oder die sonstige Entität angeben, die die Berechtigungen erhalten soll (gilt nur für ressourcenbasierte Richtlinien).

Weitere Informationen zur Syntax und zu Beschreibungen von IAM-Richtlinien finden Sie in der [AWS -IAM-Richtlinienreferenz](#) im IAM-Benutzerhandbuch.

Eine Tabelle mit allen API-Aktionen von MemoryDB für Redis finden Sie unter [MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen](#)

Angeben von Bedingungen in einer Richtlinie

Beim Erteilen von Berechtigungen können Sie mithilfe der IAM-Richtliniensyntax die Bedingungen angeben, unter denen die Richtlinie wirksam werden soll. Beispielsweise kann festgelegt werden, dass eine Richtlinie erst ab einem bestimmten Datum gilt. Weitere Informationen zum Angeben von Bedingungen in einer Richtliniensyntax finden Sie im Thema [Bedingung](#) im IAM Benutzerhandbuch.

Verwenden von identitätsbasierten Richtlinien (IAM-Richtlinien) für MemoryDB for Redis

In diesem Thema finden Sie Beispiele für identitätsbasierte Richtlinien, in denen ein Kontoadministrator den IAM-Identitäten (Benutzer, Gruppen und Rollen) Berechtigungsrichtlinien anfügen kann.

Important

Wir empfehlen Ihnen, zuerst die Themen zu lesen, in denen die grundlegenden Konzepte und Optionen zur Verwaltung des Zugriffs auf MemoryDB for Redis-Ressourcen erläutert werden. Weitere Informationen finden Sie unter [Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre MemoryDB-Ressourcen](#).

Dieses Thema besteht aus folgenden Abschnitten:

- [Für die Verwendung der MemoryDB for Redis-Konsole sind Berechtigungen erforderlich](#)
- [AWS-verwaltete \(vordefinierte\) Richtlinien für MemoryDB for Redis](#)
- [Beispiele für vom Kunden verwaltete Richtlinien](#)

Dies ist ein Beispiel für eine Berechtigungsrichtlinie.

```
{
  "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"
  }
]
```

```
]
}
```

Die Richtlinie enthält zwei Anweisungen:

- Die erste Anweisung gewährt Berechtigungen für die MemoryDB for Redis-Aktionen (`memorydb:CreateCluster`, `memorydb:UpdateCluster`) auf jedem `memorydb:DescribeClusters`, der dem Konto gehört.
- Die zweite Anweisung erteilt Berechtigungen für die IAM-Aktion (`iam:PassRole`) für den IAM-Rollenamen am Ende des Resource-Wertes.

Das Element `Principal` ist in der Richtlinie nicht angegeben, da in identitätsbasierten Richtlinien die Angabe des Prinzipals als Empfänger der Berechtigung nicht erforderlich ist. Wenn Sie einem Benutzer eine Richtlinie zuweisen, ist der Benutzer automatisch der Prinzipal. Wird die Berechtigungsrichtlinie einer IAM-Rolle angefügt, erhält der in der Vertrauensrichtlinie der Rolle angegebene Prinzipal die Berechtigungen.

Eine Tabelle mit allen API-Aktionen von MemoryDB für Redis und den Ressourcen, für die sie gelten, finden Sie unter [MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen](#)

Für die Verwendung der MemoryDB for Redis-Konsole sind Berechtigungen erforderlich

Die Referenztabelle für Berechtigungen listet die API-Operationen von MemoryDB für Redis auf und zeigt die erforderlichen Berechtigungen für jeden Vorgang. Weitere Informationen zu MemoryDB-API-Vorgängen finden Sie unter [MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen](#)

Um die MemoryDB for Redis-Konsole zu verwenden, gewähren Sie zunächst Berechtigungen für zusätzliche Aktionen, wie in der folgenden Berechtigungsrichtlinie beschrieben.

```
{
  "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": "*"
  }
]
}

```

Die MemoryDB-Konsole benötigt diese zusätzlichen Berechtigungen aus den folgenden Gründen:

- Berechtigungen für die MemoryDB-Aktionen ermöglichen es der Konsole, MemoryDB-Ressourcen im Konto anzuzeigen.
- Die Konsole benötigt Berechtigungen für die ec2-Aktionen zur Abfrage von Amazon EC2, damit sie Availability Zones, VPCs, Sicherheitsgruppen und Kontoattribute anzeigen kann.
- Die Berechtigungen für cloudwatch Aktionen ermöglichen es der Konsole, CloudWatch Amazon-Metriken und -Alarmer abzurufen und sie in der Konsole anzuzeigen.
- Die Berechtigungen für sns-Aktionen ermöglichen es der Konsole, Themen und Abonnements von Amazon Simple Notification Service (Amazon SNS) abzurufen und in der Konsole anzuzeigen.

Beispiele für vom Kunden verwaltete Richtlinien

Wenn Sie keine Standardrichtlinie verwenden und eine benutzerdefinierte verwaltete Richtlinie verwenden möchten, stellen Sie eine sicher. Sie sollten entweder die Genehmigung zum Aufrufen von `iam:createServiceLinkedRole` haben (weitere Informationen finden Sie unter [Beispiel 4: Erlauben Sie einem Benutzer, die IAM-API aufzurufen CreateServiceLinkedRole](#)). Oder Sie hätten eine mit dem Service verknüpfte MemoryDB-Rolle erstellen sollen.

In Kombination mit den Mindestberechtigungen, die für die Verwendung der MemoryDB for Redis-Konsole erforderlich sind, gewähren die Beispielrichtlinien in diesem Abschnitt zusätzliche Berechtigungen. Die Beispiele sind auch relevant für die AWS SDKs und die AWS CLI. Weitere Informationen darüber, welche Berechtigungen für die Verwendung der MemoryDB-Konsole erforderlich sind, finden Sie unter [Für die Verwendung der MemoryDB for Redis-Konsole sind Berechtigungen erforderlich](#).

Anweisungen zum Einrichten von IAM-Benutzern und -Gruppen finden Sie unter [Erstellen Ihrer ersten IAM-Benutzer- und -Administratorengruppe](#) im IAM-Benutzerhandbuch.

Important

Testen Sie Ihre IAM-Richtlinien immer gründlich, bevor Sie sie in der Produktion verwenden. Einige MemoryDB-Aktionen, die einfach erscheinen, können andere Aktionen erfordern, um sie zu unterstützen, wenn Sie die MemoryDB-Konsole verwenden. `memorydb:CreateCluster` Gewährt beispielsweise Berechtigungen zum Erstellen von MemoryDB-Clustern. Um diesen Vorgang auszuführen, verwendet die MemoryDB-Konsole jedoch eine Reihe von `List` AND-Aktionen, um `Describe` Konsolenlisten aufzufüllen.

Beispiele

- [Beispiel 1: Erlauben Sie einem Benutzer nur Lesezugriff auf MemoryDB-Ressourcen](#)
- [Beispiel 2: Erlauben Sie einem Benutzer, allgemeine MemoryDB-Systemadministrationsaufgaben auszuführen](#)
- [Beispiel 3: Erlauben Sie einem Benutzer den Zugriff auf alle MemoryDB-API-Aktionen](#)
- [Beispiel 4: Erlauben Sie einem Benutzer, die IAM-API aufzurufen `CreateServiceLinkedRole`](#)

Beispiel 1: Erlauben Sie einem Benutzer nur Lesezugriff auf MemoryDB-Ressourcen

Die folgende Richtlinie gewährt Berechtigungen für MemoryDB-Aktionen, die es einem Benutzer ermöglichen, Ressourcen aufzulisten. Gewöhnlich ordnen Sie diese Art von Berechtigungsrichtlinie einer Gruppe von Managern zu.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "MemDBUnrestricted",
    "Effect": "Allow",
    "Action": [
      "memorydb:Describe*",
      "memorydb:List*"
    ],
    "Resource": "*"
  ]
}
```

Beispiel 2: Erlauben Sie einem Benutzer, allgemeine MemoryDB-Systemadministratortasken auszuführen

Zu den allgemeinen Systemadministratortasken gehört das Ändern von Clustern, Parametern und Parametergruppen. Ein Systemadministrator möchte möglicherweise auch Informationen über die MemoryDB-Ereignisse abrufen. Die folgende Richtlinie gewährt einem Benutzer Berechtigungen zur Ausführung von MemoryDB-Aktionen für diese allgemeinen Systemadministratortasken. Normalerweise ordnen Sie diese Art Berechtigungsrichtlinie der Systemadministratortengruppe zu.

```
{
  "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": "*"
  }
]
```

Beispiel 3: Erlauben Sie einem Benutzer den Zugriff auf alle MemoryDB-API-Aktionen

Die folgende Richtlinie ermöglicht einem Benutzer den Zugriff auf alle MemoryDB-Aktionen. Es wird empfohlen, diese Art von Berechtigungsrichtlinie nur einem Administratorbenutzer zu gewähren.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "MDBAllowAll",
    "Effect": "Allow",
    "Action": [
      "memorydb:*"
    ],
    "Resource": "*"
  }
]
```

```
}
```

Beispiel 4: Erlauben Sie einem Benutzer, die IAM-API aufzurufen `CreateServiceLinkedRole`

Die folgende Richtlinie ermöglicht dem Benutzer den Aufruf der `IAM-CreateServiceLinkedRole`-API. Wir empfehlen, dass Sie dem Benutzer, der mutative MemoryDB-Operationen aufruft, diese Art von Berechtigungsrichtlinie gewähren.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CreateSLRAllows",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWS ServiceName": "memorydb.amazonaws.com"
        }
      }
    }
  ]
}
```

Berechtigungen auf Ressourcenebene

Sie können den Umfang der Berechtigungen einschränken, indem Sie Ressourcen in einer IAM-Richtlinie festlegen. Viele AWS CLI API-Aktionen unterstützen einen Ressourcentyp, der je nach Verhalten der Aktion variiert. Jede IAM-Richtlinienanweisung erteilt die Berechtigung für eine Aktion, die auf eine Ressource ausgeführt wird. Wenn die Aktion nicht auf eine benannte Ressource wirkt oder wenn Sie die Erlaubnis erteilen, die Aktion auf allen Ressourcen durchzuführen, ist der Wert der Ressource in der Richtlinie ein Platzhalter (*). Für viele API-Aktionen können Sie die Ressourcen einschränken, die ein Benutzer ändern kann. Hierzu geben Sie den Amazon-Ressourcennamen (ARN) einer Ressource oder ein ARN-Muster an, das mehreren Ressourcen entspricht. Zum Einschränken von Berechtigungen nach Ressource bestimmen Sie die Ressource unter Angabe des ARN.

ARN-Format für MemoryDB-Ressourcen

Note

Damit Berechtigungen auf Ressourcenebene wirksam sind, sollte der Ressourcenname in der ARN-Zeichenfolge in Kleinbuchstaben geschrieben werden.

- *Benutzer* – `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/mein-cluster`
- *Schnappschuss* – `arn:aws:memorydb: us-east- 1:123456789012:snapshot/mein-snapshot`
- *Parametergruppe* – `arn:aws:memorydb: us-east- 1:123456789012:parametergroup/ my-parameter-group`
- *Subnetzgruppe* – `arn:aws:memorydb: us-east- 1:123456789012:subnetgroup/ my-subnet-group`

Beispiele

- [Beispiel 1: Erlauben Sie einem Benutzer vollen Zugriff auf bestimmte MemoryDB-Ressourcentypen](#)
- [Beispiel 2: Verweigern Sie einem Benutzer den Zugriff auf einen Cluster.](#)

Beispiel 1: Erlauben Sie einem Benutzer vollen Zugriff auf bestimmte MemoryDB-Ressourcentypen

Die folgende Richtlinie erlaubt ausdrücklich den angegebenen `account-id` Vollzugriff auf alle Ressourcen vom Typ Subnetzgruppe, Sicherheitsgruppe und 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/*"
  ]
}
```

Beispiel 2: Verweigern Sie einem Benutzer den Zugriff auf einen Cluster.

Im folgenden Beispiel wird der angegebene `account-id` Zugriff auf einen bestimmten Cluster explizit verweigert.

```
{
  "Sid": "Example2",
  "Effect": "Deny",
  "Action": "memorydb:*",
  "Resource": [
    "arn:aws:memorydb:us-east-1:account-id:cluster/name"
  ]
}
```

Verwenden von serviceverknüpften Rollen für Amazon MemoryDB for Redis

[Amazon MemoryDB for Redis verwendet AWS Identity and Access Management \(IAM\) serviceverknüpfte Rollen.](#) Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit einem AWS Service verknüpft ist, z. B. Amazon MemoryDB for Redis. Servicebezogene Rollen von Amazon MemoryDB for Redis sind von Amazon MemoryDB for Redis vordefiniert. Sie enthalten alle Berechtigungen, die der Dienst benötigt, um AWS -Dienste im Namen Ihrer Cluster aufzurufen.

Eine serviceverknüpfte Rolle erleichtert die Einrichtung von Amazon MemoryDB for Redis, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Die Rollen sind bereits in Ihrem AWS Konto vorhanden, sind jedoch mit Amazon MemoryDB for Redis-Anwendungsfällen verknüpft und verfügen über vordefinierte Berechtigungen. Nur Amazon MemoryDB for Redis kann diese Rollen übernehmen, und nur diese Rollen können die vordefinierte Berechtigungsrichtlinie verwenden. Sie können die Rollen nur nach dem Löschen der zugehörigen Ressourcen löschen. Dadurch werden Ihre Amazon MemoryDB for Redis-Ressourcen geschützt, da Sie nicht versehentlich die erforderlichen Berechtigungen für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpften Rollen unterstützen, finden Sie unter [AWS -Services, die mit IAM funktionieren](#). Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Inhalt

- [Servicebezogene Rollenberechtigungen für Amazon MemoryDB for Redis](#)
- [Erstellen einer serviceverknüpften Rolle \(IAM\)](#)

- [Erstellen einer serviceverknüpften Rolle \(IAM-Konsole\)](#)
- [Erstellen einer serviceverknüpften Rolle \(IAM-CLI\)](#)
- [Erstellen einer serviceverknüpften Rolle \(IAM-API\)](#)
- [Bearbeiten der Beschreibung einer serviceverknüpften Rolle für Amazon MemoryDB for Redis](#)
 - [Bearbeiten der Beschreibung einer serviceverknüpften Rolle \(IAM-Konsole\)](#)
 - [Bearbeiten der Beschreibung einer serviceverknüpften Rolle \(IAM-CLI\)](#)
 - [Bearbeiten der Beschreibung einer serviceverknüpften Rolle \(IAM-API\)](#)
- [Löschen einer serviceverknüpften Rolle für Amazon MemoryDB for Redis](#)
 - [Bereinigen einer serviceverknüpften Rolle](#)
 - [Löschen einer serviceverknüpften Rolle \(IAM-Konsole\)](#)
 - [Löschen einer serviceverknüpften Rolle \(IAM-CLI\)](#)
 - [Löschen einer serviceverknüpften Rolle \(IAM-API\)](#)

Servicebezogene Rollenberechtigungen für Amazon MemoryDB for Redis

Amazon MemoryDB for Redis verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForMemoryDB`— Diese Richtlinie ermöglicht es MemoryDB, AWS Ressourcen in Ihrem Namen zu verwalten, soweit dies für die Verwaltung Ihrer Cluster erforderlich ist.

Die Richtlinie für `AWSServiceRoleForMemoryDB` servicebezogene Rollenberechtigungen ermöglicht es Amazon MemoryDB for Redis, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

```
{
  "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"
      }
    }
  }
]
}

```

Weitere Informationen finden Sie unter [AWSverwaltete Richtlinie: MemoryDBServiceRolePolicy](#).

Um es einer IAM-Entität zu ermöglichen, serviceverknüpfte Rollen zu erstellen
AWSServiceRoleForMemoryDB

Die folgende Berechtigungsanweisung zu den Berechtigungen für diese IAM-Entität hinzufügen:

```

{
  "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"}}
}

```

Um einer IAM-Entität das Löschen von dienstbezogenen Rollen zu ermöglichen
AWSServiceRoleForMemoryDB

Die folgende Berechtigungsanweisung zu den Berechtigungen für diese IAM-Entität hinzufügen:

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

Alternativ können Sie eine AWS verwaltete Richtlinie verwenden, um vollen Zugriff auf Amazon MemoryDB for Redis zu gewähren.

Erstellen einer serviceverknüpften Rolle (IAM)

Sie können eine serviceverknüpfte Rolle mithilfe der IAM-Konsole, der CLI oder API erstellen.

Erstellen einer serviceverknüpften Rolle (IAM-Konsole)

Sie können die IAM-Konsole für das Erstellen einer serviceverknüpften Rolle verwenden.

So erstellen Sie eine serviceverknüpfte Rolle (Konsole)

1. [Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.](https://console.aws.amazon.com/iam/)
2. Wählen Sie im linken Navigationsbereich der IAM-Konsole Rollen aus. Klicken Sie auf Create New Role (Neue Rolle erstellen).
3. Wählen Sie unter Select type of trusted entity (Typ der vertrauenswürdigen Entität auswählen) die Option AWS -Service aus.
4. Wählen Sie unter Oder wählen Sie einen Dienst aus, um seine Anwendungsfälle anzuzeigen, die Option MemoryDB aus.
5. Wählen Sie Weiter: Berechtigungen aus.
6. Beachten Sie unter Richtliniennamen, dass `MemoryDBServiceRolePolicy` für diese Rolle erforderlich ist. Wählen Sie Weiter: Tags aus.
7. Beachten Sie, dass Tags für serviceverknüpfte Rollen nicht unterstützt werden. Klicken Sie auf Next:Review (Weiter: Prüfen).
8. (Optional:) Bearbeiten Sie in Role description die Beschreibung für die neue serviceverknüpfte Rolle.

9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Erstellen einer serviceverknüpften Rolle (IAM-CLI)

Sie können IAM-Operationen verwenden, um eine AWS Command Line Interface serviceverknüpfte Rolle zu erstellen. Diese Rolle kann die Vertrauensrichtlinie, sowie die enthaltenen Richtlinien enthalten, die der Service für die Zuweisung der Rolle benötigt.

So erstellen Sie eine serviceverknüpfte Rolle (CLI)

Führen Sie die folgenden Operationen aus:

```
$ aws iam create-service-linked-role --aws-service-name memorydb.amazonaws.com
```

Erstellen einer serviceverknüpften Rolle (IAM-API)

Sie können die IAM-API für das Erstellen einer serviceverknüpften Rolle verwenden. Diese Rolle kann die Vertrauensrichtlinie, sowie die enthaltenen Richtlinien enthalten, die der Service für die Zuweisung der Rolle benötigt.

So erstellen Sie eine serviceverknüpfte Rolle (API)

Verwenden Sie den [CreateServiceLinkedRole](#)-API-Aufruf. Geben Sie in der Anforderung einen Servicenamen im Format `memorydb.amazonaws.com` an.

Bearbeiten der Beschreibung einer serviceverknüpften Rolle für Amazon MemoryDB for Redis

Amazon MemoryDB for Redis erlaubt Ihnen nicht, die serviceverknüpfte Rolle zu bearbeiten. `AWSServiceRoleForMemoryDB` Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten.

Bearbeiten der Beschreibung einer serviceverknüpften Rolle (IAM-Konsole)

Sie können die IAM-Konsole für das Bearbeiten der Beschreibung einer serviceverknüpften Rolle verwenden.

So bearbeiten Sie die Beschreibung einer serviceverknüpften Rolle (Konsole)

1. Wählen Sie im linken Navigationsbereich der IAM-Konsole die Option Rollen aus.

2. Wählen Sie den Namen der zu ändernden Rolle.
3. Wählen Sie neben Role description ganz rechts Edit.
4. Geben Sie eine neue Beschreibung im Dialogfeld ein und klicken Sie auf Save (Speichern).

Bearbeiten der Beschreibung einer serviceverknüpften Rolle (IAM-CLI)

Sie können IAM-Operationen von aus verwenden, AWS Command Line Interface um eine mit einem Dienst verknüpfte Rollenbeschreibung zu bearbeiten.

So ändern Sie die Beschreibung einer serviceverknüpften Rolle (CLI)

1. (Optional) Verwenden Sie den Vorgang for IAM, um die aktuelle Beschreibung AWS CLI für eine Rolle anzuzeigen. [get-role](#)

Example

```
$ aws iam get-role --role-name AWSServiceRoleForMemoryDB
```

Verwenden Sie den Rollennamen, nicht den ARN, um sich auf Rollen mit den CLI-Operationen zu beziehen. Wenn eine Rolle zum Beispiel folgenden ARN hat: `arn:aws:iam::123456789012:role/myrole`, verweisen Sie auf die Rolle als **myrole**.

2. Verwenden Sie den Vorgang AWS CLI for IAM, um die Beschreibung einer serviceverknüpften Rolle zu aktualisieren. [update-role-description](#)

Für Linux, macOS oder Unix:

```
$ aws iam update-role-description \  
  --role-name AWSServiceRoleForMemoryDB \  
  --description "new description"
```

Für Windows:

```
$ aws iam update-role-description ^ \  
  --role-name AWSServiceRoleForMemoryDB ^ \  
  --description "new description"
```

Bearbeiten der Beschreibung einer serviceverknüpften Rolle (IAM-API)

Sie können die IAM-API für das Bearbeiten der Beschreibung einer serviceverknüpften Rolle verwenden.

So ändern Sie die Beschreibung einer serviceverknüpften Rolle (API)

1. (Optional) Um die aktuelle Beschreibung einer Rolle anzuzeigen, verwenden Sie die IAM-API-Operation [GetRole](#).

Example

```
https://iam.amazonaws.com/  
?Action=GetRole  
&RoleName=AWSServiceRoleForMemoryDB  
&Version=2010-05-08  
&AUTHPARAMS
```

2. Um die Beschreibung einer Rolle zu aktualisieren, verwenden Sie die IAM-API-Operation [UpdateRoleDescription](#).

Example

```
https://iam.amazonaws.com/  
?Action=UpdateRoleDescription  
&RoleName=AWSServiceRoleForMemoryDB  
&Version=2010-05-08  
&Description="New description"
```

Löschen einer serviceverknüpften Rolle für Amazon MemoryDB for Redis

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie löschen können.

Amazon MemoryDB for Redis löscht die serviceverknüpfte Rolle nicht für Sie.

Bereinigen einer serviceverknüpften Rolle

Bevor Sie IAM verwenden können, um eine serviceverknüpfte Rolle zu löschen, stellen Sie zunächst sicher, dass der Rolle keine Ressourcen (Cluster) zugeordnet sind.

So überprüfen Sie in der IAM-Konsole, ob die serviceverknüpfte Rolle über eine aktive Sitzung verfügt

1. [Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.](https://console.aws.amazon.com/iam/)
2. Wählen Sie im linken Navigationsbereich der IAM-Konsole Rollen aus. Wählen Sie dann den Namen (nicht das Kontrollkästchen) der AWSServiceRoleForMemoryDB Rolle aus.
3. Wählen Sie auf der Seite Summary für die ausgewählte Rolle die Registerkarte Access Advisor.
4. Überprüfen Sie auf der Registerkarte Access Advisor die jüngsten Aktivitäten für die serviceverknüpfte Rolle.

Um Amazon MemoryDB for Redis-Ressourcen zu löschen, die (Konsole) benötigen AWSServiceRoleForMemoryDB

- Informationen zum Löschen eines Clusters finden Sie unter:
 - [Mit dem AWS Management Console](#)
 - [Verwenden Sie den AWS CLI](#)
 - [Verwenden der MemoryDB-API](#)

Löschen einer serviceverknüpften Rolle (IAM-Konsole)

Sie können die IAM-Konsole für das Löschen einer serviceverknüpften Rolle verwenden.

So löschen Sie eine serviceverknüpfte Rolle (Konsole)

1. [Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.](https://console.aws.amazon.com/iam/)
2. Wählen Sie im linken Navigationsbereich der IAM-Konsole Rollen aus. Aktivieren Sie dann das Kontrollkästchen neben dem Rollennamen, den Sie löschen möchten, nicht den Namen oder die Zeile selbst.
3. Wählen Sie für Role actions oben auf der Seite Delete role aus.

- Überprüfen Sie auf der Bestätigungsseite die Daten, auf die der Dienst zuletzt zugegriffen hat. Aus diesen Daten geht hervor, wann jede der ausgewählten Rollen zuletzt auf einen AWS Dienst zugegriffen hat. Auf diese Weise können Sie leichter bestätigen, ob die Rolle derzeit aktiv ist. Wenn Sie fortfahren möchten, wählen Sie Yes, Delete aus, um die serviceverknüpfte Rolle zur Löschung zu übermitteln.
- Sehen Sie sich die Benachrichtigungen in der IAM-Konsole an, um den Fortschritt der Löschung der serviceverknüpften Rolle zu überwachen. Da die Löschung der serviceverknüpften IAM-Rolle asynchron erfolgt, kann die Löschung nach dem Übermitteln der Rolle für die Löschung erfolgreich sein oder fehlschlagen. Wenn der Vorgang fehlschlägt, können Sie in den Benachrichtigungen View details oder View Resources auswählen, um zu erfahren, warum die Löschung fehlgeschlagen ist.

Löschen einer serviceverknüpften Rolle (IAM-CLI)

Sie können IAM-Operationen von verwenden, AWS Command Line Interface um eine dienstverknüpfte Rolle zu löschen.

So löschen Sie eine serviceverknüpfte Rolle (CLI)

- Wenn Sie den Namen der serviceverknüpften Rolle, die Sie löschen möchten, nicht kennen, geben Sie den folgenden Befehl ein. Dieser Befehl listet die Rollen und ihre Amazon-Ressourcennamen (ARN) in Ihrem Konto auf.

```
$ aws iam get-role --role-name role-name
```

Verwenden Sie den Rollennamen, nicht den ARN, um sich auf Rollen mit den CLI-Operationen zu beziehen. Wenn eine Rolle zum Beispiel den ARN `arn:aws:iam::123456789012:role/myrole` hat, verweisen Sie auf die Rolle als **myrole**.

- Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die `deletion-task-id` aus der Antwort, um den Status der Löschaufgabe zu überprüfen. Geben Sie Folgendes ein, um eine Anforderung zum Löschen einer serviceverknüpften Rolle abzusenden.

```
$ aws iam delete-service-linked-role --role-name role-name
```

3. Geben Sie Folgendes ein, um den Status der Löschaufgabe zu überprüfen.

```
$ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
```

Der Status der Löschaufgabe kann NOT_STARTED, IN_PROGRESS, SUCCEEDED oder FAILED lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können.

Löschen einer serviceverknüpften Rolle (IAM-API)

Sie können die IAM-API zum Löschen einer serviceverknüpften Rolle verwenden.

So löschen Sie eine serviceverknüpfte Rolle (API)

1. Um eine Löschanfrage für eine serviceverknüpfte Rolle zu übermitteln, rufen Sie [DeleteServiceLinkedRole](#) auf. Geben Sie in der Anforderung einen Rollennamen an.

Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die DeletionTaskId aus der Antwort, um den Status der Löschaufgabe zu überprüfen.

2. Um den Status der Löschung zu überprüfen, rufen Sie [GetServiceLinkedRoleDeletionStatus](#) auf. Geben Sie in der Anforderung die DeletionTaskId an.

Der Status der Löschaufgabe kann NOT_STARTED, IN_PROGRESS, SUCCEEDED oder FAILED lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können.

AWSverwaltete Richtlinien für MemoryDB for Redis

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, von AWS verwaltete Richtlinien zu verwenden, als selbst Richtlinien zu schreiben. Es erfordert Zeit und Fachwissen, um [von Kunden verwaltete IAM-Richtlinien zu erstellen](#), die Ihrem Team nur die benötigten Berechtigungen bieten. Um schnell loszulegen, können Sie unsere von AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken häufige Anwendungsfälle ab und sind in Ihrem AWS-

Konto verfügbar. Weitere Informationen zu verwalteten AWS-Richtlinien finden Sie unter [Verwaltete AWS-Richtlinien](#) im IAM-Leitfaden.

AWS-Services pflegen und Aktualisieren von verwalteten AWS-Richtlinien. Die Berechtigungen in von AWS verwalteten Richtlinien können nicht geändert werden. Services fügen einer von AWS verwalteten Richtlinien gelegentlich zusätzliche Berechtigungen hinzu, um neue Funktionen zu unterstützen. Diese Art von Update betrifft alle Identitäten (Benutzer, Gruppen und Rollen), an welche die Richtlinie angehängt ist. Services aktualisieren eine von AWS verwaltete Richtlinie am ehesten, wenn eine neue Funktion gestartet wird oder neue Vorgänge verfügbar werden. Services entfernen keine Berechtigungen aus einer von AWS verwalteten Richtlinie, so dass Richtlinien-Aktualisierungen Ihre vorhandenen Berechtigungen nicht beeinträchtigen.

Darüber hinaus unterstützt AWS verwaltete Richtlinien für Auftragsfunktionen, die mehrere Services umfassen. Die von ReadOnlyAccessAWS verwaltete Richtlinie bietet beispielsweise schreibgeschützten Zugriff auf alle AWS -Services und -Ressourcen. Wenn ein Service eine neue Funktion startet, fügt AWS schreibgeschützte Berechtigungen für neue Vorgänge und Ressourcen hinzu. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in [Verwaltete AWS-Richtlinien für Auftragsfunktionen](#) im IAM-Leitfaden.

AWSverwaltete Richtlinie: MemoryDBServiceRolePolicy

Sie können die ServiceRolePolicy AWS verwaltete MemoryDB-Richtlinie nicht an Identitäten in Ihrem Konto anhängen. Diese Richtlinie ist Teil der dienstverknüpften Rolle AWS MemoryDB. Diese Rolle ermöglicht es dem Dienst, Netzwerkschnittstellen und Sicherheitsgruppen in Ihrem Konto zu verwalten.

MemoryDB verwendet die Berechtigungen in dieser Richtlinie, um EC2-Sicherheitsgruppen und Netzwerkschnittstellen zu verwalten. Dies ist erforderlich, um MemoryDB-Cluster zu verwalten.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

```

{
  "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"
        }
      }
    }
  ]
}

```

AWS-verwaltete (vordefinierte) Richtlinien für MemoryDB for Redis

AWS Durch die Bereitstellung von eigenständigen IAM-Richtlinien, die von erstellt und verwaltet werden, deckt viele häufige Anwendungsfälle ab AWS. Die verwalteten Richtlinien erteilen die erforderlichen Berechtigungen für viele häufige Anwendungsfälle, sodass Sie nicht mühsam ermitteln müssen, welche Berechtigungen erforderlich sind. Weitere Informationen finden Sie unter [AWS-verwaltete Richtlinien](#) im IAM-Benutzerhandbuch.

Die folgenden von AWS verwalteten Richtlinien, die Sie an Benutzer in Ihrem Konto anhängen können, sind spezifisch für MemoryDB:

AmazonMemoryDBReadOnlyAccess

Sie können die AmazonMemoryDBReadOnlyAccess-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt Administratorberechtigungen, die einen schreibgeschützten Zugriff auf alle MemoryDB-Ressourcen erlauben.

AmazonMemoryDBReadOnlyAccess: Gewährt Lesezugriff auf MemoryDB für Redis-Ressourcen.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "memorydb:Describe*",
      "memorydb:List*"
    ],
    "Resource": "*"
  }]
}
```

AmazonMemoryDBFullAccess

Sie können die AmazonMemoryDBFullAccess-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt Administratorberechtigungen, die vollen Zugriff auf alle MemoryDB-Ressourcen ermöglichen.

AmazonMemoryDBFullAccess — Gewährt vollen Zugriff auf MemoryDB for Redis-Ressourcen.

```
{
  "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"
  }
}
]
}

```

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für MemoryDB for Redis API-Aktionen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen.

MemoryDB-Updates für AWS verwaltete Richtlinien

Anzeigen von Details zu Aktualisierungen für AWS -verwaltete Richtlinien für MemoryDB, seit dieser Dienst mit der Verfolgung dieser Änderungen begonnen hat. Um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der MemoryDB-Dokumentverlauf-Seite.

Änderung	Beschreibung	Datum
AmazonMemoryDBFullAccess — Richtlinie hinzufügen	MemoryDB hat neue Berechtigungen hinzugefügt, um unterstützte Ressourcen zu beschreiben und aufzulisten. Diese Berechtigungen sind erforderlich, damit MemoryDB alle unterstützten Ressourcen in einem Konto abfragen kann.	10.07.2021
AmazonMemoryDBReadOnlyAccess — Richtlinie hinzufügen	MemoryDB hat neue Berechtigungen hinzugefügt, um unterstützte Ressourcen zu beschreiben und aufzulisten. Diese Berechtigungen sind	10.07.2021

Änderung	Beschreibung	Datum
	erforderlich, damit MemoryDB kontobasierte Anwendungen erstellen kann, indem alle unterstützten Ressourcen in einem Konto abgefragt werden.	
MemoryDB hat mit Verfolgung von Änderungen begonnen	Servicestart	19.08.2021

MemoryDB-API-Berechtigungen: Referenz zu Aktionen, Ressourcen und Bedingungen

Wenn Sie [Zugriffskontroll](#) - und Schreibberechtigungsrichtlinien einrichten, um sie an eine IAM-Richtlinie anzuhängen (entweder identitäts- oder ressourcenbasiert), verwenden Sie die folgende Tabelle als Referenz. In der Tabelle sind alle API-Operationen von MemoryDB for Redis und die entsprechenden Aktionen aufgeführt, für die Sie Berechtigungen zur Ausführung der Aktion erteilen können. Sie geben die Aktionen im Feld `Action` der Richtlinie und einen Ressourcenwert im Feld `Resource` der Richtlinie an. Sofern nicht anders angegeben, ist die Ressource erforderlich. Einige Felder enthalten sowohl eine erforderliche Ressource als auch optionale Ressourcen. Wenn kein Ressourcen-ARN vorhanden ist, ist die Ressource in der Richtlinie als Platzhalter (*) dargestellt.

Note

Um eine Aktion anzugeben, verwenden Sie das Präfix `memorydb:` gefolgt vom Namen der API-Operation (z. B. `memorydb:DescribeClusters`).

Protokollierung und Überwachung

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von MemoryDB for Redis und Ihren anderen Lösungen. AWS bietet die folgenden Überwachungstools, um MemoryDB zu überwachen, zu melden, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen zu ergreifen:

- Amazon CloudWatch überwacht Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen, AWS in Echtzeit. Sie können Kennzahlen erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die CPU-Auslastung oder andere Kennzahlen Ihrer Amazon EC2 EC2-Instances CloudWatch verfolgen und bei Bedarf automatisch neue Instances starten. Weitere Informationen finden Sie im [CloudWatch Amazon-Benutzerhandbuch](#).
- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien von Amazon EC2 EC2-Instances und anderen Quellen überwachen CloudTrail, speichern und darauf zugreifen. CloudWatch Logs können Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im [Amazon CloudWatch Logs-Benutzerhandbuch](#).

- AWS CloudTrail fasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können feststellen, welche Benutzer und Konten angerufen wurden AWS, von welcher Quell-IP-Adresse aus die Anrufe getätigt wurden und wann die Aufrufe erfolgten. Weitere Informationen finden Sie im [AWS CloudTrail -Benutzerhandbuch](#).

Überwachung von MemoryDB für Redis mit Amazon CloudWatch

Sie können MemoryDB for Redis verwenden CloudWatch, das Rohdaten sammelt und zu lesbaren Metriken verarbeitet, die nahezu in Echtzeit verfügbar sind. Diese Statistiken werden 15 Monate gespeichert, damit Sie auf Verlaufsdaten zugreifen können und einen besseren Überblick darüber erhalten, wie Ihre Webanwendung oder der Service ausgeführt werden. Sie können auch Alarme einrichten, die auf bestimmte Grenzwerte achten und Benachrichtigungen senden oder Aktivitäten auslösen, wenn diese Grenzwerte erreicht werden. Weitere Informationen finden Sie im [CloudWatch Amazon-Benutzerhandbuch](#).

In den folgenden Abschnitten sind die Metriken und Dimensionen für MemoryDB aufgeführt.

Themen

- [Metriken auf Host-Ebene](#)
- [Metriken für MemoryDB](#)
- [Welche Metriken sollte ich überwachen?](#)
- [Auswählen von Metrikstatistiken und -zeiträumen](#)
- [CloudWatch Metriken überwachen](#)

Metriken auf Host-Ebene

Der AWS/MemoryDB Namespace umfasst die folgenden Metriken auf Host-Ebene für einzelne Knoten.

Weitere Informationen finden Sie auch unter:

- [Metriken für MemoryDB](#)

Metrik	Beschreibung	Einheit
CPUUtilization	Der Prozentsatz der CPU-Nutzung für den gesamten Host. Weil Redis Single-Thread-fähig ist und wir empfehlen, die EngineCPU Utilization Metrik für Knoten mit 4 oder mehr vCPUs zu überwachen.	Prozent
FreeableMemory	Größe des freien Arbeitsspeichers auf dem Host. Das ergibt sich aus dem RAM, den Puffern und den Angaben, die das Betriebssystem als frei verfügbar meldet.	Bytes
NetworkBytesIn	Anzahl der Byte, die der Host aus dem Netzwerk gelesen hat.	Bytes
NetworkBytesOut	Anzahl der von der Instance auf allen Netzwerkschnittstellen gesendeten Byte.	Bytes
NetworkPacketsIn	Anzahl der von der Instance auf allen Netzwerkschnittstellen empfangenen Pakete. Diese Metrik gibt das an eine einzelne Instance eingehende Netzwerkdatenvolumen an, ausgedrückt in Anzahl an Paketen.	Anzahl
NetworkPacketsOut	Anzahl der von der Instance auf allen Netzwerkschnittstellen gesendeten Pakete. Diese Metrik gibt das von einer einzelnen Instance ausgehende Netzwerkdatenvolumen an, ausgedrückt in Anzahl an Paketen.	Anzahl
NetworkBandwidthIn AllowanceExceeded	Die Anzahl der Pakete wurde geformt, weil die eingehende aggregierte Bandbreite das Maximum für die Instance überschritten hat.	Anzahl
NetworkConntrackAllowanceExceeded	Die Anzahl der Pakete wurde geformt, weil die Verbindungsverfolgung das Maximum für die Instance überschritten hat und keine neuen	Anzahl

Metrik	Beschreibung	Einheit
	Verbindungen hergestellt werden konnten. Dies kann zu einem Paketverlust für den Datenverkehr zur oder von der Instance führen.	
NetworkBandwidthOutAllowanceExceeded	Die Anzahl der Pakete wurde geformt, weil die ausgehende aggregierte Bandbreite das Maximum für die Instance überschritten hat.	Anzahl
NetworkPacketsPerSecondAllowanceExceeded	Die Anzahl der Pakete, die geformt wurden, weil Anzahl der bidirektionalen Pakete pro Sekunde das Maximum für die Instance überschritten hat.	Anzahl
NetworkMaxBytesIn	Die maximale Steigerung der empfangenen Bytes innerhalb jeder Minute.	Bytes
NetworkMaxBytesOut	Die maximale Steigerung der übertragenen Bytes innerhalb jeder Minute.	Bytes
NetworkMaxPacketsIn	Die maximale Steigerung der empfangenen Pakete innerhalb jeder Minute.	Anzahl
NetworkMaxPacketsOut	Die maximale Steigerung der übertragenen Pakete innerhalb jeder Minute.	Anzahl
SwapUsage	Größe des belegten Auslagerungsspeichers auf dem Host.	Bytes

Metriken für MemoryDB

Der AWS/MemoryDB-Namespace enthält die folgenden Redis-Metriken.

Mit Ausnahme von `ReplicationLag` und `EngineCPUUtilization` werden diese Metriken vom Redis-Befehl `info` abgeleitet. Jede Metrik wird auf Knotenebene berechnet.

Die vollständige Dokumentation für den Redis-Befehl `info` finden Sie unter <http://redis.io/commands/info>.

Weitere Informationen finden Sie auch unter:

- [Metriken auf Host-Ebene](#)

Metrik	Beschreibung	Einheit
ActiveDefragHits	Die Anzahl der Werteneuzuweisungen pro Minute, die der aktive Defragmentierungsprozess durchführt. Dies ist von der Statistik <code>active_defrag_hits</code> unter Redis INFO abgeleitet.	Zahl
AuthenticationFailures	Die Gesamtzahl der fehlgeschlagenen Versuche, sich mit dem AUTH-Befehl bei Redis zu authentifizieren. Weitere Informationen zu einzelnen Authentifizierungsfehlern finden Sie mit dem Befehl ACL LOG . Wir empfehlen, hierauf einen Alarm zu setzen, um unberechtigte Zugriffsversuche zu erkennen.	Anzahl
BytesUsedForMemoryDB	Die Gesamtzahl der Byte, die MemoryDB für alle Zwecke zugewiesen hat, einschließlich des Datensatzes, der Puffer usw.	Bytes
	Dimension: Tier=SSD für Cluster mit Daten-Tiering : Die Gesamtzahl der von SSD verwendeten Byte.	Bytes
	Dimension: Tier=Memory für Cluster mit Daten-Tiering : Die Gesamtzahl der vom Speicher verwendeten Byte. Das ist der Wert der <code>used_memory</code> -Statistik bei Redis INFO .	Bytes
BytesReadFromDisk	Gesamtzahl der von der Festplatte pro Minute gelesenen Bytes. Wird nur für Cluster mit Daten-Tiering unterstützt.	Bytes

Metrik	Beschreibung	Einheit
BytesWrittenToDisk	Gesamtzahl der Bytes, die pro Minute auf den Datenträger geschrieben werden. Wird nur für Cluster mit Daten-Tiering unterstützt.	Bytes
CommandAuthorizationFailures	Die Gesamtzahl der fehlgeschlagenen Versuche von Benutzern, Befehle auszuführen, für deren Aufruf sie keine Berechtigung haben. Weitere Informationen zu einzelnen Authentifizierungsfehlern finden Sie mit dem Befehl ACL LOG . Wir empfehlen, hierauf einen Alarm zu setzen, um unberechtigte Zugriffsversuche zu erkennen.	Anzahl
CurrConnections	Die Anzahl der Clientverbindungen, ausgenommen Verbindungen von Leserepliken. MemoryDB verwendet jeweils zwei bis vier der Verbindungen, um den Cluster zu überwachen. Dies ist von <code>connected_clients</code> unter Redis INFO abgeleitet.	Anzahl
CurrItems	Anzahl der Elemente im Cache. Dies wird aus der Redis-Statistik <code>keyspace</code> abgeleitet, indem alle Schlüssel im gesamten Schlüsselbereich summiert werden.	Anzahl
	Dimension: <code>Tier=Memory</code> für Cluster mit Daten-Tiering . Anzahl der Elemente im Speicher.	Anzahl
	Dimension: <code>Tier=SSD</code> (Solid-State-Laufwerke) für Cluster mit Daten-Tiering . Anzahl der Elemente im SSD.	Anzahl

Metrik	Beschreibung	Einheit
DatabaseMemoryUsagePercentage	Prozentsatz des Speichers, der für den verwendeten Cluster verfügbar ist. Eine Berechnung erfolgt mit <code>used_memory/maxmemory</code> aus Redis INFO .	Prozent
DatabaseCapacityUsagePercentage	<p>Prozentsatz der gesamten Datenkapazität für den Cluster, die genutzt wird.</p> <p><u>Bei Data-Tiered-Instances wird die Metrik wie folgt $(\text{used_memory} - \text{mem_not_counted_for_evict} + \text{SSD used}) / (\text{maxmemory} + \text{SSD total capacity})$ berechnet <code>used_memory</code> und stammt aus <code>maxmemory</code> Redis INFO.</u></p> <p>In allen anderen Fällen wird die Metrik mit <code>used_memory/maxmemory</code> berechnet.</p>	Prozent
DB0AverageTTL	Stellt <code>avg_ttl</code> von DBO aus der keyspace-Statistik des Redis INFO -Befehls dar.	Millisekunden

Metrik	Beschreibung	Einheit
EngineCPUUtilization	<p>Gibt die CPU-Auslastung des Redis-Engine-Threads an. Weil Redis nur über einen Thread verfügt, können Sie diese Metrik zum Analysieren der Auslastung des Redis-Prozesses selbst verwenden. Die EngineCPUUtilization-Metrik bietet eine genauere Sichtbarkeit des Redis-Prozesses. Sie können dies in Verbindung mit der CPUUtilization-Metrik verwenden. CPUUtilization legt die CPU-Auslastung für die Server-Instance als Ganzes offen, einschließlich anderer Betriebssystem- und Verwaltungsprozesse. Verwenden Sie für größere Knotentypen mit vier vCPUs oder mehr die EngineCPUUtilization-Metrik zum Überwachen und Einstellen der Schwellenwerte für die Skalierung.</p> <div data-bbox="592 1024 1269 1862"><p> Note</p><p>Auf einem MemoryDB-Host überwachen Hintergrundprozesse den Host, um eine verwaltete Datenbankumgebung zu gewährleisten. Diese Hintergrundprozesse können einen erheblichen Teil der CPU-Workload beanspruchen. Dies ist auf größeren Hosts mit mehr als zwei vCPUs nicht signifikant. Es kann sich jedoch auf kleinere Hosts mit 2vCPUs oder weniger auswirken. Wenn Sie nur die EngineCPUUtilization-Metrik überwachen, werden Sie keine Situationen erkennen, in denen der Host sowohl mit hoher CPU-Auslastung von Redis als auch mit hoher CPU-Auslastung aus den</p></div>	Prozent

Metrik	Beschreibung	Einheit
	<div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p>Hintergrundüberwachungsprozessen überlastet ist. Daher empfehlen wir die Überwachung der CPUUtilization - Metrik für Hosts mit zwei vCPUs oder weniger.</p> </div>	
Evictions	Anzahl der Schlüssel, die infolge des maxmemory -Grenzwertes bereinigt worden sind. Dies ist von evicted_keys unter Redis INFO abgeleitet.	Anzahl
IsPrimary	Gibt an, ob es sich bei dem Knoten um den Primärknoten des aktuellen Shards handelt. Die Metrik kann entweder 0 (nicht primär) oder 1 (primär) sein.	Anzahl
KeyAuthorizationFailures	Die Gesamtzahl der fehlgeschlagenen Versuche von Benutzern, auf Schlüssel zuzugreifen, für die sie keine Zugriffsberechtigung haben. Weitere Informationen zu einzelnen Authentifizierungsfehlern finden Sie mit dem Befehl ACL LOG . Wir empfehlen, hierauf einen Alarm zu setzen, um unberechtigte Zugriffsversuche zu erkennen.	Anzahl
KeyspaceHits	Die Anzahl der erfolgreichen schreibgeschützten Schlüsselsuchereignisse im Hauptverzeichnis. Dies ist von der Statistik keyspace_hits unter Redis INFO abgeleitet.	Anzahl

Metrik	Beschreibung	Einheit
KeyspaceMisses	Die Anzahl der nicht erfolgreichen schreibgeschützten Schlüsselsuchereignisse im Hauptverzeichnis. Dies ist von der Statistik <code>keyspace_misses</code> unter Redis INFO abgeleitet.	Anzahl
KeysTracked	Die Anzahl der Schlüssel, die von der Redis-Schlüsselverfolgung verfolgt werden, in Prozent von <code>tracking-table-max-keys</code> . Die Schlüsselverfolgung wird verwendet, um das clientseitige Caching zu unterstützen und Clients zu benachrichtigen, wenn Schlüssel geändert werden.	Anzahl
MaxReplicationThroughput	Der maximale beobachtete Replikationsdurchsatz während des letzten Messzyklus.	Bytes pro Sekunde
MemoryFragmentationRatio	Gibt die Effizienz bei der Speicherzuweisung der Redis Engine an. Bestimmte Schwellenwerte weisen auf unterschiedliche Verhaltensweisen hin. Der empfohlene Wert ist eine Fragmentierung über 1,0. Diese wird aus der <code>mem_fragmentation_ratio</code> statistic von Redis INFO berechnet.	Zahl
NewConnections	Gesamtanzahl der Verbindungen, die in diesem Zeitraum vom Server akzeptiert worden sind. Dies ist von <code>total_connections_received</code> unter Redis INFO abgeleitet.	Anzahl
NumItemsReadFromDisk	Die Gesamtzahl der pro Minute von der Festplatte abgerufenen Elemente. Wird nur für Cluster mit Daten-Tiering unterstützt.	Anzahl

Metrik	Beschreibung	Einheit
NumItemsWrittenToDisk	Die Gesamtzahl der pro Minute auf die Festplatte geschriebenen Elemente. Wird nur für Cluster mit Daten-Tiering unterstützt.	Anzahl
PrimaryLinkHealthStatus	Dieser Status kann zwei Werte annehmen: 0 oder 1. Der Wert 0 gibt an, dass die Daten im primären MemoryDB-Knoten nicht mit Redis auf EC2 synchronisiert sind. Der Wert 1 bedeutet, dass die Daten synchronisiert sind.	Boolesch
Reclaimed	Gesamtanzahl der Schlüsselablaufereignisse Dies ist von <code>expired_keys</code> unter Redis INFO abgeleitet.	Anzahl
ReplicationBytes	Für Knoten in einer replizierten Konfiguration gibt <code>ReplicationBytes</code> die Anzahl der Bytes an, die der Primärknoten an alle seine Replikationen sendet. Diese Metrik ist repräsentativ für die Schreiblast auf dem Cluster. Dies ist von <code>master_repl_offset</code> unter Redis INFO abgeleitet.	Bytes
ReplicationDelayedWriteCommands	Anzahl der Schreibbefehle, die aufgrund der synchronen Replikation verzögert wurden. Die Replikation kann sich aufgrund verschiedener Faktoren verzögern, z. B. aufgrund von Netzwerküberlastung oder Überschreitung des maximalen Replikationsdurchsatzes .	Anzahl
ReplicationLag	Diese Metrik ist nur für einen als Read Replica laufenden Knoten verfügbar. Sie stellt die Zeitverzögerung in Sekunden dar, mit der die Replica die vom primären Knoten kommenden Änderungen anwendet.	Sekunden

Im Folgenden finden Sie Zusammenfassungen bestimmter Befehle, die von `info commandstats` abgeleitet sind. Der Abschnitt `commandstats` enthält Statistiken, die auf dem Befehlstyp basieren, einschließlich der Anzahl der Aufrufe.

Eine vollständige Liste der verfügbaren Befehle finden Sie unter [Redis-Befehle](#) in der Redis-Dokumentation.

Metrik	Beschreibung	Einheit
<code>EvalBasedCmds</code>	Die Gesamtzahl der Befehle für EVAL-basierte Befehle. Diese wird aus der Statistik der Redis- <code>commandstats</code> -Statistik abgeleitet. Diese wird aus der Redis- <code>commandstats</code> -Statistik durch Summieren von <code>eval</code> , <code>evalsha</code> abgeleitet.	Anzahl
<code>GeoSpatialBasedCmds</code>	Die Gesamtzahl der Befehle für raumbezogene Befehle. Diese wird aus der Statistik der Redis- <code>commandstats</code> -Statistik abgeleitet. Es wird abgeleitet, indem alle Befehle des Geo-Typs summiert werden: <code>geoadd</code> , <code>geodist</code> , <code>geohash</code> , <code>geopos</code> , <code>georadius</code> und <code>georadiusbymember</code> .	Anzahl
<code>GetTypeCmds</code>	Gesamtanzahl der auf read-only basierenden Befehle. Dies wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, in der alle read-only-Befehle (<code>get</code> , <code>hget</code> , <code>scard</code> , <code>lrange</code> usw.) summiert werden	Anzahl
<code>HashBasedCmds</code>	Gesamtanzahl der Hash-basierten Befehle. Diese wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, indem alle Befehle, die auf eine oder mehrere Hashes wirken, summiert werden (<code>hget</code> , <code>hkeys</code> , <code>hvals</code> , <code>hdel</code> usw.).	Anzahl
<code>HyperLogLogBasedCmds</code>	Gesamtanzahl der auf HyperLogLog basierenden Befehle. Dies wird aus der Redis-	Anzahl

Metrik	Beschreibung	Einheit
	Statistik <code>commandstats</code> abgeleitet, indem alle <code>pf</code> -Befehle (<code>pfadd</code> , <code>pfcount</code> , <code>pfmerge</code> usw.) summiert werden.	
<code>JsonBasedCmds</code>	Die Gesamtanzahl der JSON-basierten Befehle. Dies wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, indem alle Befehle summiert werden, die auf ein oder mehrere JSON-Dokumentobjekte einwirken.	Anzahl
<code>KeyBasedCmds</code>	Gesamtanzahl der schlüsselbasierten Befehle. Dies wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, indem alle Befehle summiert werden, die über mehrere Datenstrukturen hinweg auf mindestens einen Schlüssel einwirken (<code>del</code> , <code>expire</code> , <code>rename</code> usw.).	Anzahl
<code>ListBasedCmds</code>	Gesamtanzahl der listenbasierten Befehle. Dieser Wert wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, indem alle Befehle summiert werden, die einzelne oder mehrere Listen bearbeiten (<code>lindex</code> , <code>lrange</code> , <code>lpush</code> , <code>ltrim</code> usw.).	Anzahl
<code>PubSubBasedCmds</code>	Die Gesamtzahl der Befehle für die Pub/Sub-Funktionalität. Dies wird aus den <code>commandstats</code> Redis-Statistiken abgeleitet, indem alle Befehle zusammengefasst werden, die für die Pub/Sub-Funktionalität verwendet werden: <code>punsubscribe</code> , <code>publish</code> , <code>pubsub</code> und <code>punsubscribe</code> <code>subscribe</code> <code>unsubscribe</code>	Anzahl

Metrik	Beschreibung	Einheit
SearchBasedCmds	Die Gesamtzahl der sekundären Index- und Suchbefehle, einschließlich Lese- und Schreibbefehlen. Dies wird aus der <code>commandstats</code> Redis-Statistik abgeleitet, indem alle Suchbefehle summiert werden, die sich auf sekundäre Indizes auswirken.	Anzahl
SearchBasedGetCmds	Gesamtzahl der Befehle für den sekundären Index und die schreibgeschützte Suche. Dies wird aus der <code>commandstats</code> Redis-Statistik abgeleitet, indem alle sekundären Index- und Suchabfragebefehle summiert werden.	Anzahl
SearchBasedSetCmds	Gesamtzahl der sekundären Index- und Suchbefehle zum Schreiben. Dies wird aus der <code>commandstats</code> Redis-Statistik abgeleitet, indem alle Befehle für den sekundären Index und die Suchgruppe summiert werden.	Anzahl
SearchNumberOfIndexes	Gesamtzahl der Indizes.	Anzahl
SearchNumberOfIndexedKeys	Gesamtzahl der indizierten Redis-Schlüssel	Anzahl
SearchTotalIndexSize	Speicher (Byte), der von allen Indizes verwendet wird.	Bytes
SetBasedCmds	Gesamtanzahl der Set-basierten Befehle. Diese wird aus der Redis-Statistik <code>commandstats</code> abgeleitet, indem alle Befehle, die auf eine oder mehrere Sets wirken, summiert werden (<code>scard</code> , <code>sdiff</code> , <code>sadd</code> , <code>sunion</code> usw.).	Anzahl

Metrik	Beschreibung	Einheit
SetTypeCmds	Gesamtanzahl der auf write basierenden Befehle. Dieser Wert wird von der Redis-commandstats -Statistik abgeleitet, indem alle Arten von mutative-Befehlen, die auf Daten wirken, addiert werden (set, hset, sadd, lpop usw.).	Anzahl
SortedSetBasedCmds	Gesamtanzahl der Sorted Set-basierten Befehle. Sie wird von der Redis-commandstats -Statistik abgeleitet, indem alle Befehle, die auf eine oder mehrere sortierte Gruppen wirken, addiert werden (zcount, zrange, zrank, zadd usw.).	Anzahl
StringBasedCmds	Gesamtanzahl der Zeichenfolge-basierten Befehle. Diese wird aus der Redis-commandstats -Statistik abgeleitet, indem alle Befehle, die auf eine oder mehrere Zeichenketten wirken, summiert werden (strlen, setex, setrange usw.).	Anzahl
StreamBasedCmds	Die Gesamtanzahl Stream-basierter Befehle. Dieser Wert wird aus der Redis-commandstats -Statistik abgeleitet, indem alle Befehle summiert werden, die einzelne oder mehrere Stream-Datentypen bearbeiten (xrange, xlen, xadd, xdel usw.).	Anzahl

Welche Metriken sollte ich überwachen?

Die folgenden CloudWatch Metriken bieten einen guten Einblick in die Leistung von MemoryDB. In den meisten Fällen empfehlen wir, CloudWatch Alarmer für diese Metriken einzurichten, damit Sie Korrekturmaßnahmen ergreifen können, bevor Leistungsprobleme auftreten.

Zu überwachende Metriken

- [CPUUtilization](#)
- [EngineCPUUtilization](#)
- [SwapUsage](#)
- [Evictions](#)
- [CurrConnections](#)
- [Arbeitsspeicher](#)
- [Network \(Netzwerk\)](#)
- [Replikation](#)

CPUUtilization

Diese Metrik auf Hostebene wird in Prozent angegeben. Weitere Informationen finden Sie unter [Metriken auf Host-Ebene](#).

Verwenden Sie für kleinere Knotentypen mit 2vCPUs oder weniger die Metrik `CPUUtilization`, um Ihre Workload zu überwachen.

Generell empfehlen wir, den Schwellenwert auf 90 % der verfügbaren CPU-Kapazität festzulegen. Da Redis mit nur einem Thread arbeitet, sollte der tatsächliche Schwellenwert als ein Bruchteil der Gesamtkapazität des Knotens berechnet werden. Angenommen, Sie verwenden einen Knotentyp mit zwei Kernen. In diesem Fall wäre der Grenzwert für `CPUUtilization` $90/2$ bzw. 45 %. Informationen zur Anzahl der Kerne (vCPUs) Ihres Knotentyps finden Sie unter [MemoryDB-Preise](#).

Sie müssen Ihren eigenen Schwellenwert festlegen, der auf der Anzahl der Kerne in dem Knoten basiert, den Sie verwenden. Wenn Sie diesen Schwellenwert überschreiten und Ihre Hauptlast aus Leseanfragen besteht, skalieren Sie Ihren Cluster, indem Sie Read Replicas hinzufügen. Wenn die Hauptlast aus Schreibanforderungen besteht, empfehlen wir Ihnen, mehr Shards hinzuzufügen, um die Schreiblast auf mehr Primärknoten zu verteilen.

Tip

Anstatt die Metrik auf Host-Ebene zu verwenden `CPUUtilization`, können Sie möglicherweise die Redis-Metrik verwenden `EngineCPUUtilization`, die den Prozentsatz der Nutzung auf dem Redis-Engine-Kern meldet. [Um zu sehen, ob diese Metrik auf Ihren Knoten verfügbar ist, und weitere Informationen finden Sie unter Metriken für MemoryDB.](#)

Verwenden Sie für größere Knotentypen mit 4vCPUs oder mehr die Metrik `EngineCPUUtilization`, die den Prozentsatz der Auslastung des Redis-Engine-Kerns angibt. Um zu sehen, ob diese Metrik auf Ihren Knoten verfügbar ist, und weitere Informationen finden Sie unter [Metriken für MemoryDB](#).

EngineCPUUtilization

Verwenden Sie für größere Knotentypen mit 4vCPUs oder mehr die Metrik `EngineCPUUtilization`, die den Prozentsatz der Auslastung des Redis-Engine-Kerns angibt. Um zu sehen, ob diese Metrik auf Ihren Knoten verfügbar ist, und weitere Informationen finden Sie unter [Metriken für MemoryDB](#).

SwapUsage

Diese Metrik auf Hostebene wird in Bytes angegeben. Weitere Informationen finden Sie unter [Metriken auf Host-Ebene](#).

Diese Metrik sollte 50 MB nicht überschreiten.

Evictions

Dies ist eine Engine-Metrik. Wir empfehlen Ihnen, einen eigenen Grenzwert für diese Metrik basierend auf den Anforderungen Ihrer Anwendung zu bestimmen.

CurrConnections

Dies ist eine Motormetrik. Wir empfehlen Ihnen, einen eigenen Grenzwert für diese Metrik basierend auf den Anforderungen Ihrer Anwendung zu bestimmen.

Eine zunehmende Anzahl von `CurrConnections` kann auf ein Problem mit Ihrer Anwendung hinweisen. Um dieses Problem zu beheben, müssen Sie das Verhalten der Anwendung untersuchen.

Arbeitsspeicher

Speicher ist ein zentraler Aspekt von Redis. Es ist notwendig, die Speicherauslastung Ihres Clusters zu verstehen, um Datenverluste zu vermeiden und das zukünftige Wachstum Ihres Datasets berücksichtigen zu können. Statistiken zur Speicherauslastung eines Knotens finden Sie im Speicherbereich des Redis [INFO](#)-Befehls.

Network (Netzwerk)

Einer der entscheidenden Faktoren für die Kapazität der Netzwerkbandbreite Ihres Clusters ist der von Ihnen ausgewählte Knotentyp. Weitere Informationen zur Netzwerkkapazität Ihres Nodes finden Sie unter [Amazon MemoryDB-Preise](#).

Replikation

Das Datenvolumen, das repliziert wird, ist über die `ReplicationBytes`-Metrik ersichtbar. Sie können den Durchsatz der Replikationskapazität `MaxReplicationThroughput` anhand der Replikationskapazität überwachen. Es wird empfohlen, weitere Shards hinzuzufügen, wenn der maximale Durchsatz für die Replikationskapazität erreicht ist.

`ReplicationDelayedWriteCommand` kann auch angeben, ob die Arbeitslast den maximalen Durchsatz der Replikationskapazität überschreitet. Weitere Informationen zur Replikation in MemoryDB finden Sie unter [Grundlegendes](#) zur MemoryDB-Replikation

Auswählen von Metrikstatistiken und -zeiträumen

Es ermöglicht CloudWatch Ihnen zwar, für jede Metrik eine beliebige Statistik und einen beliebigen Zeitraum auszuwählen, aber nicht alle Kombinationen sind sinnvoll. So sind zum Beispiel die Statistiken Durchschnitt, Minimum und Maximum für CPUUtilization nützlich, die Statistik Summe jedoch nicht.

Alle MemoryDB-Samples werden für einen Zeitraum von 60 Sekunden für jeden einzelnen Knoten veröffentlicht. Für jeden Zeitraum von 60 Sekunden enthält eine Knotenmetrik nur eine einzige Stichprobe.

CloudWatch Metriken überwachen

MemoryDB und CloudWatch sind integriert, sodass Sie eine Vielzahl von Metriken sammeln können. Sie können diese Metriken überwachen mit CloudWatch

Note

Für die folgenden Beispiele sind die CloudWatch Befehlszeilentools erforderlich. Weitere Informationen zu den Entwicklertools CloudWatch und zum Herunterladen finden Sie auf der [CloudWatch Produktseite](#).

Die folgenden Verfahren zeigen Ihnen, wie Sie Speicherplatzstatistiken für einen Cluster für die letzte Stunde sammeln können. CloudWatch

Note

Die in den folgenden Beispielen angegebenen EndTime Werte StartTime und dienen der Veranschaulichung. Stellen Sie sicher, dass Sie Ihre Knoten durch geeignete Start- und Endzeitwerte ersetzen.

Informationen zu MemoryDB-Grenzwerten finden Sie unter [AWS Service Limits](#) for MemoryDB.

Metriken überwachen CloudWatch (Konsole)

Um Statistiken zur CPU-Auslastung für einen Cluster zu sammeln

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie die Knoten aus, für die Sie Metriken anzeigen möchten.

 Note

Bei der Auswahl von mehr als 20 Knoten wird die Ansicht von Metriken auf der Konsole deaktiviert.

- a. Klicken Sie auf der Seite Cluster der AWS Management Console auf den Namen eines oder mehrerer Cluster.

Die Detailseite für den Cluster wird angezeigt.

- b. Klicken Sie oben im Fenster auf die Registerkarte Nodes.
- c. Wählen Sie auf der Registerkarte Knoten des Detailfensters die Knoten aus, für die Sie Metriken anzeigen möchten.

Eine Liste der verfügbaren CloudWatch Metriken wird unten im Konsolenfenster angezeigt.

- d. Klicken Sie auf die Metrik CPU Utilization.

Die CloudWatch Konsole wird geöffnet und zeigt Ihre ausgewählten Metriken an. Sie können die Dropdown-Listenfelder Statistic und Period und die Registerkarte Time Range verwenden, um die angezeigten Metriken zu ändern.

Überwachen von CloudWatch Metriken mit der CloudWatch CLI

Um Statistiken zur CPU-Auslastung für einen Cluster zu sammeln

- Verwenden Sie den CloudWatch Befehl `aws cloudwatch get-metric-statistics` mit den folgenden Parametern (beachten Sie, dass die Start- und Endzeiten nur als Beispiele angezeigt werden; Sie müssen Ihre eigenen entsprechenden Start- und Endzeiten ersetzen):

Für Linux, macOS oder Unix:

```
aws cloudwatch get-metric-statistics CPUUtilization \  
  --dimensions=ClusterName=mycluster,NodeId=0002 \  
  --statistics=Average \  
  --start-time=2017-01-01T00:00:00Z \  
  --end-time=2017-01-01T00:00:00Z
```

```
--namespace="AWS/MemoryDB" \  
--start-time 2013-07-05T00:00:00 \  
--end-time 2013-07-06T00:00:00 \  
--period=60
```

Für 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
```

Überwachung von CloudWatch Metriken mithilfe der CloudWatch API

Um Statistiken zur CPU-Auslastung für einen Cluster zu sammeln

- Rufen Sie die CloudWatch API `GetMetricStatistics` mit den folgenden Parametern auf (beachten Sie, dass die Start- und Endzeiten nur als Beispiele angezeigt werden; Sie müssen Ihre eigenen entsprechenden Start- und Endzeiten ersetzen):
 - `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>
```

Überwachung von MemoryDB auf Redis-Ereignisse

Wenn für einen Cluster wichtige Ereignisse eintreten, sendet MemoryDB eine Benachrichtigung an ein bestimmtes Amazon SNS SNS-Thema. Zu den wichtigen Ereignissen zählen beispielsweise das fehlgeschlagene Hinzufügen eines Knotens, das erfolgreiche Hinzufügen eines Knotens und die Änderung einer Sicherheitsgruppe. Durch die Überwachung wichtiger Schlüsselereignisse können Sie den aktuellen Status Ihrer Cluster erfahren und, je nach Ereignis, Korrekturen vornehmen.

Themen

- [MemoryDB Amazon SNS SNS-Benachrichtigungen verwalten](#)
- [MemoryDB-Ereignisse anzeigen](#)
- [Ereignisbenachrichtigungen und Amazon SNS](#)

MemoryDB Amazon SNS SNS-Benachrichtigungen verwalten

Sie können MemoryDB so konfigurieren, dass Benachrichtigungen für wichtige Cluster-Ereignisse mithilfe von Amazon Simple Notification Service (Amazon SNS) gesendet werden. In diesen Beispielen konfigurieren Sie einen Cluster mit dem Amazon-Ressourcenname (ARN) eines Amazon-SNS-Themas, um Benachrichtigungen zu erhalten.

Note

In diesem Thema wird davon ausgegangen, dass Sie sich bei Amazon SNS angemeldet und ein Amazon-SNS-Thema eingerichtet und abonniert haben. Informationen dazu finden Sie im [Entwicklerhandbuch zu Amazon Simple Notification Service](#).

Hinzufügen eines Amazon-SNS-Themas

In den folgenden Abschnitten erfahren Sie, wie Sie mithilfe der AWS Konsole, der oder der MemoryDB-API ein Amazon SNS SNS-Thema hinzufügen. AWS CLI

Hinzufügen eines Amazon-SNS-Themas (Konsole)

Das folgende Verfahren zeigt Ihnen, wie Sie ein Amazon-SNS-Thema für einen Cluster hinzufügen.

Note

Diese Vorgehensweise kann auch zum Ändern des Amazon-SNS-Themas verwendet werden.

So fügen Sie ein Amazon-SNS-Thema für einen Cluster hinzu oder ändern es (Konsole)

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie unter Cluster den Cluster aus, für den Sie einen Amazon-SNS-Thema-ARN hinzufügen oder ändern möchten.
3. Wählen Sie Ändern aus.
4. Wählen Sie im Feld Modify Cluster (Cluster ändern) unter Topic for SNS Notification (Thema für SNS-Benachrichtigung) das SNS-Thema aus, das Sie hinzufügen möchten, oder wählen Sie Manual ARN input (Manuelle ARN-Eingabe) aus und geben Sie den ARN des Amazon-SNS-Themas ein.
5. Wählen Sie Ändern aus.

Hinzufügen eines Amazon SNS SNS-Themas (AWS CLI)

Verwenden Sie den AWS CLI Befehl `update-cluster`, um ein Amazon SNS SNS-Thema für einen Cluster hinzuzufügen oder zu ändern.

Das folgende Codebeispiel fügt einen Amazon-SNS-Themen-ARN zu `my-cluster` hinzu.

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

Für Windows:

```
aws memorydb update-cluster ^  
  --cluster-name my-cluster ^  
  --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

Weitere Informationen finden Sie unter [UpdateCluster](#).

Hinzufügen eines Amazon SNS SNS-Themas (MemoryDB-API)

Um ein Amazon SNS SNS-Thema für einen Cluster hinzuzufügen oder zu aktualisieren, rufen Sie die `UpdateCluster` Aktion mit den folgenden Parametern auf:

- `ClusterName=my-cluster`
- `SnsTopicArn=arn%3Aaws%3Asns%3Aus-east-1%3A565419523791%3AmemorydbNotifications`

Um ein Amazon SNS SNS-Thema für einen Cluster hinzuzufügen oder zu aktualisieren, rufen Sie die `UpdateCluster` Aktion auf.

Weitere Informationen finden Sie unter [UpdateCluster](#).

Aktivieren und Deaktivieren von Amazon-SNS-Benachrichtigungen

Sie können Benachrichtigungen für einen Cluster aktivieren oder deaktivieren. Das folgende Verfahren zeigt, wie Sie Amazon-SNS-Benachrichtigungen deaktivieren.

Aktivieren und Deaktivieren von Amazon-SNS-Benachrichtigungen (Konsole)

Um Amazon SNS SNS-Benachrichtigungen zu deaktivieren, verwenden Sie den AWS Management Console

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole unter https://console.aws.amazon.com/memorydb/ an AWS Management Console und öffnen Sie sie.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie das Optionsfeld links neben dem Cluster, für den Sie die Benachrichtigung ändern möchten.
3. Wählen Sie Ändern aus.
4. Wählen Sie im Feld Modify Cluster unter Topic for SNS Notification die Option Disable Notifications aus.
5. Wählen Sie Ändern aus.

Amazon SNS SNS-Benachrichtigungen (AWS CLI) aktivieren und deaktivieren

Verwenden Sie zum Deaktivieren von Amazon-SNS-Benachrichtigungen den Befehl `update-cluster` mit folgenden Parametern:

Für Linux, macOS oder Unix:

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --sns-topic-status inactive
```

Für Windows:

```
aws memorydb update-cluster ^  
  --cluster-name my-cluster ^  
  --sns-topic-status inactive
```

Amazon SNS SNS-Benachrichtigungen aktivieren und deaktivieren (MemoryDB-API)

Rufen Sie zum Deaktivieren von Amazon-SNS-Benachrichtigungen die `UpdateCluster`-Aktion mit folgenden Parametern auf:

- `ClusterName=my-cluster`
- `SnsTopicStatus=inactive`

Diese Aktion führt zu folgender oder einer ähnlichen Ausgabe:

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>
```

MemoryDB-Ereignisse anzeigen

MemoryDB protokolliert Ereignisse, die sich auf Ihre Cluster, Sicherheitsgruppen und Parametergruppen beziehen. Diese Informationen beinhalten Datum und Zeit eines Ereignisses, den Quellnamen und Quelltyp sowie eine Beschreibung des Ereignisses. Sie können Ereignisse einfach mit der MemoryDB-Konsole, dem AWS CLI `describe-events` Befehl oder der MemoryDB-API-Aktion aus dem Protokoll abrufen. `DescribeEvents`

Die folgenden Verfahren zeigen Ihnen, wie Sie alle MemoryDB-Ereignisse der letzten 24 Stunden (1440 Minuten) anzeigen können.

MemoryDB-Ereignisse anzeigen (Konsole)

Das folgende Verfahren zeigt Ereignisse mithilfe der MemoryDB-Konsole an.

Um Ereignisse mit der MemoryDB-Konsole anzuzeigen

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Ereignisse aus.

Der Bildschirm „Ereignisse“ wird angezeigt und listet alle verfügbaren Ereignisse auf. Jede Zeile der Liste steht für ein Ereignis und zeigt die Ereignisquelle, den Ereignistyp (wie Cluster, Parametergruppe, ACL, Sicherheitsgruppe oder Subnetzgruppe), die GMT-Zeit des Ereignisses und die Beschreibung des Ereignisses an.

Mit der Option `Filter` können Sie angeben, ob alle Ereignisse oder nur Ereignisse eines bestimmten Typs in der Ereignisliste enthalten sein sollen.

MemoryDB-Ereignisse anzeigen (CLI)AWS

Verwenden Sie den Befehl, um mit dem eine Liste von MemoryDB-Ereignissen zu generieren. `AWS CLI describe-events` Mit optionalen Parametern können Sie u. a. den Typ und den Zeitrahmen der aufgelisteten Ereignisse sowie die maximale Anzahl der aufzulistenden Ereignisse steuern.

Der folgende Code listet bis zu 40 Cluster-Ereignisse auf.

```
aws memorydb describe-events --source-type cluster --max-results 40
```

Mit dem folgenden Code werden alle Ereignisse der letzten 24 Stunden (1 440 Minuten) aufgelistet.

```
aws memorydb describe-events --duration 1440
```

Die Ausgabe des Befehls `describe-events` sieht in etwa wie folgt aus:

```
{
  "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"
    }
  ]
}
```

Weitere Informationen z. B. zu den verfügbaren Parametern und den zulässigen Parameterwerten finden Sie unter [describe-events](#).

MemoryDB-Ereignisse anzeigen (MemoryDB-API)

Verwenden Sie die Aktion, um mithilfe der MemoryDB-API eine Liste von MemoryDB-Ereignissen zu generieren. `DescribeEvents` Mit optionalen Parametern können Sie u. a. den Typ und den Zeitrahmen der aufgelisteten Ereignisse sowie die maximale Anzahl der aufzulistenden Ereignisse steuern.

Der folgende Code listet die 40 neuesten -Cluster-Ereignisse auf.

```
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>
```

Der folgende Code listet die Cluster-Ereignisse der letzten 24 Stunden (1440 Minuten) auf.

```
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>
```

Die Ausgabe der oben angegebenen Aktionen sollte in etwa wie folgt aussehen:

```
<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>
```

Weitere Informationen z. B. zu den verfügbaren Parametern und den zulässigen Parameterwerten finden Sie unter [DescribeEvents](#).

Ereignisbenachrichtigungen und Amazon SNS

MemoryDB kann Nachrichten über den Amazon Simple Notification Service (SNS) veröffentlichen, wenn wichtige Ereignisse auf einem Cluster auftreten. Mit dieser Funktion können die Serverlisten auf Client-Computern aktualisiert werden, die mit einzelnen Knotenendpunkten eines Clusters verbunden sind.

Note

Weitere Informationen zum Amazon Simple Notification Service (SNS) sowie Informationen zu Preisen und Links zur Amazon-SNS-Dokumentation finden Sie auf der [Produktseite zu Amazon SNS](#).

Benachrichtigungen werden in einem bestimmten Amazon-SNS-Thema veröffentlicht. Für Benachrichtigungen müssen folgende Voraussetzungen erfüllt sein:

- Es kann nur ein Thema für MemoryDB-Benachrichtigungen konfiguriert werden.
- Die AWS Das Amazon SNS-Thema muss im selben Konto enthalten sein wie das Cluster, für das die Benachrichtigungen aktiviert sind.

MemoryDB Ereignisse

Die folgenden MemoryDB-Ereignisse lösen Amazon SNS SNS-Benachrichtigungen aus:

Ereignisname	Fehlermeldung	Description
memoryDB:addNodeComplete	"Modified number of nodes from %d to %d"	Ein Knoten wurde zum Cluster hinzugefügt und steht zur Verwendung bereit.
memoryDB:AddnodeFailed to insufficient free IP-Adressen	"Failed to modify number of nodes from %d to %d due to insufficient free IP addresses"	Ein Knoten konnte nicht hinzugefügt werden, da nicht genug IP-Adressen verfügbar sind.

Ereignisname	Fehlermeldung	Description
memoryDB:ClusterParameterSchanged	<p>"Updated parameter group for the cluster"</p> <p>Beim Erstellen wird auch "Updated to use a ParameterGroup %s" gesendet.</p>	Ein oder mehrere Clusterparameter wurden geändert.
memoryDB:ClusterProvisioningComplete	"Cluster created."	Die Bereitstellung eines Clusters ist abgeschlossen und die Knoten im Cluster stehen zur Verwendung bereit.
memoryDB:ClusterProvisioningFailed to incompatible network state	"Failed to create cluster due to incompatible network state. %s"	Es wurde versucht, einen neuen Cluster in einer nicht vorhandenen Virtual Private Cloud (VPC) zu starten.
memoryDB:ClusterRestoreFailed	"Restore from %s failed for node %s. %s"	<p>MemoryDB konnte den Cluster nicht mit Redis-Snapshot-Daten füllen. Dies könnte auf eine nicht vorhandene Snapshot-Datei in Amazon S3 oder auf falsche Berechtigungen für diese Datei zurückzuführen sein. Wenn Sie den Cluster beschreiben (siehe), ist dessen Status <code>restore-failed</code> aus. Sie müssen den Cluster löschen und von Neuem beginnen.</p> <p>Weitere Informationen finden Sie unter Einen neuen Cluster mit einem extern erstellten Snapshot erstellen.</p>

Ereignisname	Fehlermeldung	Description
memoryDB:ClusterScalingComplete	"Succeeded applying modification to node type to %s."	Skalierung des Clusters wurde erfolgreich abgeschlossen.
memoryDB:ClusterScalingfehlgeschlagen	"Failed applying modification to node type to %s."	Die Skalierung des Clusters ist fehlgeschlagen.
memoryDB:ClusterSecurityGroupModified	"Modified security group for cluster."	<p>Eines der folgenden Ereignisse ist aufgetreten:</p> <ul style="list-style-type: none"> • Die Liste der für den Cluster autorisierten Sicherheitsgruppen wurde geändert. • Eine oder mehrere neue EC2-Sicherheitsgruppe wurde für eine der Sicherheitsgruppen autorisiert, die dem Cluster zugeordnet sind. • Eine oder mehrere EC2-Sicherheitsgruppen wurden aus den Sicherheitsgruppen entfernt, die dem Cluster zugeordnet sind.

Ereignisname	Fehlermeldung	Description
memoryDB:NoderePlaceStarted	"Recovering node %s"	<p>MemoryDB hat erkannt, dass der Host, auf dem ein Knoten ausgeführt wird, schlecht oder nicht erreichbar ist, und hat damit begonnen, den Knoten zu ersetzen.</p> <div data-bbox="1068 541 1507 808"><p> Note</p><p>Der DNS-Eintrag für den ersetzten Knoten bleibt unverändert.</p></div> <p>In den meisten Fällen müssen Sie die Serverliste für die Clients nicht aktualisieren, wenn dieses Ereignis auftritt. Einige Client-Bibliotheken verwenden den Knoten möglicherweise aber nicht mehr, auch wenn MemoryDB den Knoten ersetzt hat. In diesem Fall muss die Serverliste durch die Anwendung aktualisiert werden, wenn dieses Ereignis auftritt.</p>

Ereignisname	Fehlermeldung	Description
memoryDB:NoderePlaceComplete	"Finished recovery for node %s"	<p>MemoryDB hat erkannt, dass der Host, auf dem ein Knoten ausgeführt wird, schlecht oder nicht erreichbar ist, und hat den Knoten ersetzt.</p> <div data-bbox="1068 495 1510 760" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"><p> Note</p><p>Der DNS-Eintrag für den ersetzten Knoten bleibt unverändert.</p></div> <p>In den meisten Fällen müssen Sie die Serverliste für die Clients nicht aktualisieren, wenn dieses Ereignis auftritt. Einige Client-Bibliotheken verwenden den Knoten möglicherweise aber nicht mehr, auch wenn MemoryDB den Knoten ersetzt hat. In diesem Fall muss die Serverliste durch die Anwendung aktualisiert werden, wenn dieses Ereignis auftritt.</p>
memoryDB:createClusterComplete	"Cluster created"	Der Cluster wurde erfolgreich erstellt.

Ereignisname	Fehlermeldung	Description
memoryDB:createClusterFailed	"Failed to create cluster due to unsuccessful creation of its node(s)." und "Deleting all nodes belonging to this cluster."	Der Cluster wurde nicht erstellt.
memoryDB:deleteClusterComplete	"Cluster deleted."	Das Löschen eines Clusters und aller zugeordneten Knoten wurde abgeschlossen.
memoryDB:FailOverComplete	"Failover to replica node %s completed"	Failover zu einem Replikationsknoten wurde erfolgreich abgeschlossen.
memoryDB:NoderePlacementCancelementCanceledR	"The replacement of node %s which was scheduled during the maintenance window from start time: %s, end time: %s has been canceled"	Ein Knoten in Ihrem Cluster, der ersetzt werden sollte, soll nicht länger ersetzt werden.
memoryDB:NoderePlacementMentreGeplant	"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"	Für einen zu ersetzenden Knoten wurde eine spätere Ersetzung im neuen Fenster geplant, das in der Benachrichtigung angegeben wurde. Weitere Informationen zu den möglichen Aktionen erhalten Sie unter Ersetzen von Knoten .

Ereignisname	Fehlermeldung	Description
memoryDB:NodereplacementGeplante	"The node %s is scheduled for replacement during the maintenance window from start time: %s to end time: %s"	Ein Knoten in Ihrem Cluster soll während des in der Benachrichtigung beschriebenen Fensters ersetzt werden. Weitere Informationen zu den möglichen Aktionen erhalten Sie unter Ersetzen von Knoten .
memoryDB:removenodeComplete	"Removed node %s"	Ein Knoten wurde aus dem Cluster entfernt.
memoryDB:snapshotComplete	"Snapshot %s succeeded for node %s"	Ein Snapshot wurde erfolgreich abgeschlossen.
memoryDB:SnapshotFailed	"Snapshot %s failed for node %s"	Ein Snapshot ist fehlgeschlagen. Weitere Informationen zur Ursache finden Sie in den Ereignissen des Clusters. Wenn Sie den Snapshot beschreiben (siehe) DescribeSnapshots , der Status wird sein <code>failed</code> aus.

Protokollieren von MemoryDB für Redis-API-Aufrufe mit AWS CloudTrail

MemoryDB for Redis ist in einen Service integriert AWS CloudTrail, der die Aktionen eines Benutzers, einer Rolle oder eines AWS -Service in MemoryDB for Redis aufzeichnet. CloudTrail erfasst alle API-Aufrufe für MemoryDB for Redis als Ereignisse, einschließlich Aufrufen von der MemoryDB for Redis-Konsole und von Code-Aufrufen an die MemoryDB for Redis API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket, einschließlich Ereignissen für MemoryDB for Redis aktivieren. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail -Konsole trotzdem in Event

history (Ereignisverlauf) anzeigen. Anhand der von CloudTrail erfassten Informationen können Sie die an MemoryDB for Redis gestellte Anfrage, die IP-Adresse, von der die Anfrage, den Initiator der Anforderung und weitere Details bestimmen.

Weitere Informationen CloudTrail dazu finden Sie im [AWS CloudTrailBenutzerhandbuch](#).

MemoryDB for Redis-Informationen in CloudTrail

CloudTrail wird beim Erstellen IhresAWS -Kontos für Sie aktiviert. Die in MemoryDB for Redis auftretenden Aktivitäten werden als CloudTrail Ereignis zusammen mit anderenAWS - Serviceereignissen im Ereignisverlauf aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter [Anzeigen von Ereignissen mit dem CloudTrail -API-Ereignisverlauf](#).

Erstellen Sie für eine fortlaufende Aufzeichnung der Ereignisse in IhremAWS -Konto, darunter Ereignisse für MemoryDB for Redis, einen Trail. Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon-S3-Bucket bereit. Darüber hinaus können Sie andereAWS -Services konfigurieren, um die in den CloudTrail -Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie unter:

- [Übersicht zum Erstellen eines Trails](#)
- [CloudTrail Unterstützte Dienste und Integrationen](#)
- [Konfigurieren von Amazon SNS SNS-Benachrichtigungen für CloudTrail](#)
- [Empfangen von CloudTrail Protokolldateien aus mehreren Regionen](#) und [Empfangen von CloudTrail Protokolldateien von mehreren Konten](#)

Alle MemoryDB for Redis-Aktionen werden protokolliert von CloudTrail. Beispielsweise generieren Aufrufe vonDescribeClusters undUpdateCluster Aktionen Einträge in den CloudTrail Protokolldateien.CreateCluster

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anfrage mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.

- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen verbundenen Benutzer gesendet wurde.
- Gibt an, ob die Anforderung aus einem anderen AWS-Service gesendet wurde

Weitere Informationen finden Sie unter [CloudTrail userIdentity-Element](#).

Grundlegendes zu MemoryDB for Redis-Logdateieinträgen

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen Amazon-S3-Bucket übermittelt werden. CloudTrail Protokolldateien können einen oder mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anfrageparameter. CloudTrail Protokolleinträge sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die `CreateCluster` Aktion demonstriert.

```
{
  "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"
}

```

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die `DescribeClusters` Aktion demonstriert. Beachten Sie, dass für alle MemoryDB for Redis Describe- und List-Aufrufe (`Describe*` und `List*`) der `responseElements` Abschnitt entfernt wurde und als `angezeigt wird 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"
}

```

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der eine `updateCluster` Aktion aufzeichnet.

```

{
  "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"
```

```
}

```

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die `CreateUser` Aktion demonstriert. Beachten Sie, dass bei MemoryDB for Redis-Aufrufen, die vertrauliche Daten enthalten, diese Daten im entsprechenden CloudTrail Ereignis redigiert werden, wie im folgenden `requestParameters` Abschnitt gezeigt.

```
{
  "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"
}
```

Konformitätsprüfung für MemoryDB for Redis

Externe Prüfer bewerten die Sicherheit und Konformität von MemoryDB for Redis im Rahmen mehrerer Compliance-Programme. AWS Dies umfasst:

- Payment Card Industry Data Security Standard (PCI DSS). Weitere Informationen finden Sie unter [PCI DSS](#).
- Health Insurance Portability and Accountability Act Business Associate Agreement (HIPAA BAA). Weitere Informationen finden Sie unter [HIPAA-Compliance](#).
- System and Organization Controls (SOC) 1, 2 und 3. Weitere Informationen finden Sie unter [SOC](#).
- Moderat des Federal Risk and Authorization Management Program (FedRAMP). Weitere Informationen finden Sie unter [FedRAMP](#).
- ISO/IEC 27001:2013, 27017:2015, 27018:2019 und ISO/IEC 9001:2015. Weitere Informationen finden Sie unter ISO- und CSA STAR-Zertifizierungen und [AWS-Services](#).

Eine Liste der AWS-Services, die in den Geltungsbereich bestimmter Compliance-Programme fallen, finden Sie auf der Seite [AWS-Services in Scope nach Compliance-Programm](#).

Sie können Auditberichte von Drittanbietern unter AWS Artifact herunterladen. Weitere Informationen finden Sie unter [Berichte herunterladen in AWS Artifact](#).

Ihre Compliance-Verantwortung bei der Verwendung von MemoryDB hängt von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften

ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung von Vorschriften unterstützen:

- [Quick-Start-Anleitungen für Sicherheit und Compliance](#) – In diesen Bereitstellungsleitfäden finden Sie wichtige Überlegungen zur Architektur sowie die einzelnen Schritte zur Bereitstellung von Sicherheits- und Compliance-orientierten Basisumgebungen in AWS.
- [AWS-Compliance-Ressourcen](#) – Diese Arbeitsbücher und Leitfäden könnten für Ihre Branche und Ihren Standort relevant sein.
- Die [Bewertung von Ressourcen anhand von Regeln](#) im AWS Config-Entwicklerhandbuch — AWS Config bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- [AWS Security Hub](#) – Dieser AWS-Service liefert einen umfassenden Überblick über den Sicherheitsstatus in AWS. So können Sie die Compliance mit den Sicherheitsstandards in der Branche und den bewährten Methoden abgleichen.
- [AWS Audit Manager](#) – Dieser AWS-Service hilft Ihnen, Ihre AWS-Nutzung kontinuierlich zu überprüfen, um zu vereinfachen, wie Sie mit Risiken umgehen und Compliance sowie Branchenstandards einhalten.

Sicherheit der Infrastruktur in Amazon MemoryDB in Amazon MemoryDB in Amazon MemoryDB für Redis

Als verwalteter Service ist MemoryDB durch die AWS globalen Verfahren zur Gewährleistung der Netzwerksicherheit geschützt, die im Whitepaper [Amazon Web Services: Übersicht über die Sicherheitsprozesse](#) beschrieben sind.

Sie verwenden von AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf MemoryDB, Clients müssen Transport Layer Security (TLS) 1.2 oder höher unterstützen. Wir empfehlen TTTTTTTTTTTTTTTTTT Clients müssen außerdem Verschlüsselungssammlungen mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit [AWS Security Token Service](#) (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Richtlinie für den Datenverkehr zwischen Netzwerken

MemoryDB for Redis verwendet die folgenden Techniken, um Ihre Daten zu sichern und vor unbefugtem Zugriff zu schützen:

- [MemoryDB und Amazon VPC](#) erklärt den Typ der Sicherheitsgruppe, die Sie für Ihre Installation benötigen.
- [VPC-Endpunkte MemoryDB für Redis-API und Schnittstellen-VPC-Endpunkte \(AWS PrivateLink\)](#) ermöglicht es Ihnen, eine private Verbindung zwischen Ihrer VPC und MemoryDB for Redis API-Endpunkten herzustellen.
- [Identitäts- und Zugriffsmanagement in MemoryDB für Redis](#) Verwenden Sie zum Erteilen und Beschränken von Aktionen von Benutzern, Gruppen und Rollen.

MemoryDB und Amazon VPC

Der Amazon Virtual Private Cloud (Amazon VPC)-Service definiert ein virtuelles Netzwerk, das einem herkömmlichen Rechenzentrum sehr ähnlich ist. Wenn Sie eine Virtual Private Cloud (VPC) mit Amazon VPC konfigurieren, können Sie den IP-Adressbereich auswählen, Subnetze erstellen sowie Routing-Tabellen, Netzwerk-Gateways und Sicherheitseinstellungen konfigurieren. Sie können dem virtuellen Netzwerk außerdem einen -Cluster hinzufügen und den Zugriff auf den Cluster über Amazon-VPC-Sicherheitsgruppen steuern.

In diesem Abschnitt wird erklärt, wie ein MemoryDB-Cluster manuell in einer VPC konfiguriert wird. Diese Informationen richten sich an Benutzer, die mehr darüber wissen möchten, wie MemoryDB und Amazon VPC miteinander funktionieren.

Themen

- [MemoryDB und VPCs verstehen](#)
- [Zugriffsmuster zum Zugreifen auf einen MemoryDB-Cluster in einer Amazon VPC](#)
- [Erstellen einer Virtual Private Cloud \(VPC\)](#)

MemoryDB und VPCs verstehen

MemoryDB ist vollständig in Amazon VPC integriert. Für MemoryDB-Benutzer bedeutet dies Folgendes:

- MemoryDB startet Ihren Cluster immer in einer VPC.
- Wenn für Sie neu ist AWS wird automatisch eine Standard-VPC für Sie erstellt.
- Wenn Sie eine Standard-VPC haben und beim Starten eines Clusters kein Subnetz angeben, wird der Cluster in Ihrer Standard-Amazon-VPC gestartet.

Weitere Informationen finden Sie unter [Detecting Your Supported Platforms and Whether You Have a Default VPC](#).

Mit Amazon VPC können Sie ein virtuelles Netzwerk in der AWS-Cloud, die einem herkömmlichen Rechenzentrum sehr ähnlich ist. Sie können Ihre VPC konfigurieren und den IP-Adressbereich auswählen, Subnetze erstellen sowie Routing-Tabellen, Netzwerk-Gateways und Sicherheitseinstellungen konfigurieren.

MemoryDB verwaltet Software-Upgrades, Patch-Vorgänge, Fehlererkennung und Wiederherstellung.

Überblick über MemoryDB in einer VPC

1

Eine VPC ist ein isolierter Teil des AWS-Cloud, der ein eigener Block von IP-Adressen zugeordnet ist.

2

Ein Internet-Gateway verbindet Ihre VPC direkt mit dem Internet und bietet Zugriff auf andere AWS-Ressourcen wie Amazon Simple Storage Service (Amazon S3), die außerhalb Ihrer VPC ausgeführt werden.

3

Ein Amazon VPC-Subnetz ist ein Segment des IP-Adressbereichs einer VPC, in dem Sie isolieren können AWS-Ressourcen entsprechend Ihren Sicherheits- und Betriebsanforderungen.

4

Eine Routing-Tabelle in Ihrer VPC steuert den Netzwerkdatenverkehr zwischen dem Subnetz und dem Internet. Die Amazon VPC hat einen impliziten Router.

- 5 Eine Amazon VPC-Sicherheitsgruppe steuert den eingehenden und ausgehenden Datenverkehr für Ihre MemoryDB-Cluster und Amazon EC2 Instances.
- 6 Sie können einen MemoryDB-Cluster im Subnetz starten. Die Knoten haben private IP-Adressen aus dem Adressbereich des Subnetzes.
- 7 Sie können Amazon-EC2-Instances auch im Subnetz starten. Jede Amazon-EC2-Instance hat eine private IP-Adresse aus dem Adressbereich des Subnetzes. Die Amazon EC2 EC2-Instance kann eine Verbindung zu jedem Knoten in demselben Subnetz herstellen.
- 8 Damit eine Amazon EC2 EC2-Instance in Ihrer VPC vom Internet aus erreichbar ist, müssen Sie der Instance eine statische, öffentliche Adresse namens Elastic IP-Adresse zuordnen.

Voraussetzungen

Zum Erstellen eines MemoryDB-Clusters innerhalb einer VPC muss Ihre VPC die folgenden Anforderungen erfüllen:

- Ihre VPC muss nicht dedizierte Amazon EC2 Instances zulassen. Sie können MemoryDB nicht in einer VPC verwenden, die für eine Dedicated Instance-Tenancy konfiguriert ist.
- Für Ihre VPC muss eine Subnetzgruppe definiert werden. MemoryDB verwendet diese Subnetzgruppe, um ein Subnetz und IP-Adressen innerhalb dieses Subnetzes auszuwählen und diese mit Ihrem Knoten zu verknüpfen.
- Eine Sicherheitsgruppe muss für Ihre VPC definiert sein oder Sie können die bereitgestellte standardmäßige verwenden.
- CIDR-Blöcke für jedes Subnetz müssen groß genug sein, um freie IP-Adressen für MemoryDB bereitzustellen, die während Wartungsarbeiten genutzt werden können.

Weiterleitung und Sicherheit

Sie können Weiterleitung in Ihrer VPC so konfigurieren, dass gesteuert wird, wo der Datenverkehr fließt (z. B. zum Internet-Gateway oder Virtual Private Gateway). Mit einem Internet-Gateway hat Ihre VPC direkten Zugriff auf andere AWS-Ressourcen, die nicht in Ihrer VPC ausgeführt werden. Wenn Sie sich nur für ein virtuelles privates Gateway mit einer Verbindung zum lokalen Netzwerk Ihrer

Organisation entscheiden, können Sie Ihren Internet-Datenverkehr über das VPN leiten und lokale Sicherheitsrichtlinien und eine Firewall verwenden, um ausgehenden Datenverkehr zu steuern. In diesem Fall können zusätzliche Bandbreitengebühren anfallen, wenn Sie auf AWS-Ressourcen über das Internet.

Sie können Amazon VPC-Sicherheitsgruppen verwenden, um die MemoryDB-Cluster und Amazon EC2 Instances in Ihrer Amazon VPC zu schützen. Sicherheitsgruppen wirken wie eine Firewall auf der Instance-Ebene, nicht auf der Subnetzebene.

Note

Es wird ausdrücklich empfohlen, DNS-Namen für die Verbindung mit Ihren Knoten zu verwenden, da sich die zugrunde liegende IP-Adresse im Laufe der Zeit ändern kann.

Amazon VPC-Dokumentation

Amazon VPC hat eine eigene Dokumentation, in der das Erstellen und Nutzen Ihrer erklärt wird. In der folgenden Tabelle ist aufgeführt, wo Informationen in den Amazon-VPC-Handbüchern zu finden sind.

Beschreibung	Dokumentation
Erste Schritte bei der Verwendung von Amazon VPC	Erste Schritte mit Amazon VPC
Verwenden Sie Amazon VPC über die AWS Management Console	Amazon VPC User Guide
Vollständige Beschreibungen aller Amazon-VP C-Befehle	Amazon EC2 Command Line Reference (Die Amazon VPC Befehle befinden sich in der Amazon EC2 Referenz)
Vollständige Beschreibungen der Amazon-VP C-API-Aktionen, -Datentypen und -Fehler	Amazon-EC2-API-Referenz (Die Amazon VPC API-Vorgänge befinden sich in der Amazon EC2 Referenz)
Informationen für den Netzwerkadministrator, der das Gateway an Ihrem Ende einer	Was ist AWS-Site-to-Site-VPN?

Beschreibung	Dokumentation
optionalen IPsec-VPN-Verbindung konfigurieren muss	

Weitere Informationen zur Amazon Virtual Private Cloud finden Sie unter [Amazon Virtual Private Cloud](#).

Zugriffsmuster zum Zugreifen auf einen MemoryDB-Cluster in einer Amazon VPC

MemoryDB for Redis unterstützt die folgenden Szenarien für den Zugriff auf einen -Cluster in einer Amazon VPC:

Inhalt

- [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in derselben Amazon VPC sind.](#)
- [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs sind.](#)
 - [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus derselben Region sind.](#)
 - [Verwenden von Transit Gateway](#)
 - [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus verschiedenen Regions sind.](#)
 - [Verwenden von Transit VPC](#)
- [Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung](#)
 - [Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels VPN-Konnektivität](#)
 - [Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels Direkt Connect](#)

Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in derselben Amazon VPC sind.

Der gängigste Anwendungsfall ist, wenn eine auf einer EC2-Instance bereitstellte Anwendung eine Verbindung mit einem Cluster in der gleichen VPC herstellen muss.

Nachfolgend finden Sie den leichtesten Weg für die Verwaltung des Zugriffs zwischen EC2-Instances und DB-Instances in derselben VPC:

1. Erstellen Sie eine VPC-Sicherheitsgruppe für Ihren Cluster. Diese Sicherheitsgruppe kann verwendet werden, um den Zugriff auf die Cluster zu beschränken. Sie können für diese Sicherheitsgruppe beispielsweise eine benutzerdefinierte Regel erstellen, die TCP-Zugriff über

den Port, den Sie dem Cluster bei seiner Erstellung zugeordnet haben, und eine IP-Adresse gewährt, mit der Sie auf den Cluster zugreifen.

Der Standard-Port für MemoryDB-Cluster ist 6379 aus.

2. Erstellen Sie eine VPC-Sicherheitsgruppe für Ihre EC2-Instances (Web- und Anwendungsserver). Mithilfe dieser Sicherheitsgruppe können Sie bei Bedarf den Internetzugriff auf die EC2-Instance über die Routing-Tabelle der VPC zulassen. Sie können beispielsweise Regeln für diese Sicherheitsgruppe festlegen, damit der TCP-Zugriff auf die EC2-Instance über Port 22 möglich ist.
3. Erstellen Sie in der Sicherheitsgruppe benutzerdefinierte Regeln für den -Cluster, um Verbindungen von der (für die EC2-Instances) erstellten Sicherheitsgruppe zuzulassen. Damit wird jedem Mitglied der Sicherheitsgruppe der Zugriff auf die DB-Instances gestattet.

So erstellen Sie eine Regel in einer VPC-Sicherheitsgruppe, die Verbindungen über eine andere Sicherheitsgruppe zulässt

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon-VPC-Konsole unter <https://console.aws.amazon.com/vpc>.
2. Klicken Sie im linken Navigationsbereich auf Security Groups.
3. Wählen Sie eine aus oder erstellen Sie eine, die Sie für Ihre Cluster verwenden werden. Wählen Sie unter Inbound Rules (Eingangsregeln) die Option Edit Inbound Rules (Eingangsregeln bearbeiten) und dann Add Rule (Regeln hinzufügen). Diese Sicherheitsgruppe gewährt Mitgliedern einer anderen Sicherheitsgruppe Zugriff.
4. Wählen Sie für Type die Option Custom TCP Rule aus.
 - a. Geben Sie für Port Range den Port an, den Sie beim Erstellen des Clusters verwendet haben.

Der Standard-Port für MemoryDB-Cluster ist 6379 aus.

- b. Geben Sie in das Feld Source die ersten Zeichen der ID der Sicherheitsgruppe ein. Wählen Sie aus der Liste die Sicherheitsgruppe aus, die Sie für Ihre Amazon-EC2-Instances verwenden werden.
5. Wählen Sie Save, wenn Sie fertig sind.

Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs sind.

Wenn sich Ihr Cluster in einer anderen VPC als die EC2-Instance befindet, mit der Sie auf ihn zugreifen, kann der Zugriff auf die DB-Instance auf verschiedene Arten erfolgen. Wenn sich der Cluster und die EC2-Instance in verschiedenen VPCs, aber in derselben Region befinden, können Sie VPC-Peering verwenden. Wenn sich der Cluster und die EC2-Instance in verschiedenen Regionen befinden, können Sie VPN-Konnektivität zwischen den Regionen aufbauen.

Themen

- [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus derselben Region sind.](#)
- [Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus verschiedenen Regions sind.](#)

Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus derselben Region sind.

Clusterzugriff über eine Amazon-EC2-Instance in einer anderen Amazon VPC innerhalb derselben Region – VPC-Peering-Verbindung

Peering: Eine VPC-Peering-Verbindung ist eine Netzwerkverbindung zwischen zwei VPCs. Diese ermöglicht die Weiterleitung des Datenverkehrs zwischen den VPCs mithilfe von privaten IP-Adressen. Instances in jeder der VPCs können so miteinander kommunizieren, als befänden sie sich im selben Netzwerk. Sie können eine VPC-Peering-Verbindung zwischen Ihren eigenen Amazon VPCs oder zu einer Amazon VPC eines anderen AWS-Kontos innerhalb einer Region herstellen. Weitere Informationen zum Amazon-VPC-Peering finden Sie in der [VPC-Dokumentation](#)

So greifen Sie auf einen Cluster in einer anderen Amazon VPC über Peering zu

1. Vergewissern Sie sich, dass sich der IP-Bereich der beiden VPCs nicht überlappt. Andernfalls ist kein Peering zwischen ihnen möglich.
2. Stellen Sie eine Peering-Verbindung zwischen den beiden VPCs her. Weitere Informationen finden Sie unter [Erstellen und Akzeptieren einer Amazon-VPC-Peering-Verbindung](#).
3. Aktualisieren Sie Ihre Routing-Tabelle. Weitere Informationen finden Sie unter [Aktualisieren der Routing-Tabellen für eine VPC-Peering-Verbindung](#).

4. Bearbeiten Sie die Sicherheitsgruppe des MemoryDB-Clusters, um eingehende Verbindungen von der Sicherheitsgruppe der Anwendung in der per Peering verbundenen VPC zuzulassen. Weitere Informationen finden Sie unter [Verweisen auf Peer-VPC-Sicherheitsgruppen](#).

Beim Zugriff auf einen Cluster über eine Peering-Verbindung fallen zusätzliche Datenübertragungskosten an.

Verwenden von Transit Gateway

Ein Transit Gateway ermöglicht es Ihnen, VPCs und VPN-Verbindungen in derselben AWS-Region anzufügen und Datenverkehr zwischen ihnen weiterzuleiten. Ein Transit Gateway funktioniert über AWS-Konten hinweg und Sie können den Resource Access Manager AWS verwenden, um Ihr Transit Gateway mit anderen Konten gemeinsam zu nutzen. Wenn Sie ein Transit Gateway mit einem anderen AWS-Konto gemeinsam nutzen, kann der Besitzer des Kontos Ihrem Transit Gateway seine VPCs anhängen. Benutzer in einem der Konten können die Anhang jederzeit löschen.

Sie können Multicast auf einem Transit Gateway aktivieren und dann eine Transit Gateway-Multicast-Domäne erstellen, mit der Multicast-Datenverkehr von der Multicast-Quelle über VPC-Anhängen, die Sie der Domäne zuordnen, an Multicast-Gruppenmitglieder gesendet werden kann.

Sie können auch einen Peering-Verbindungs-Anhang zwischen Transit Gateways in verschiedenen AWS-Regionen erstellen. Auf diese Weise können Sie den Datenverkehr zwischen den Anhängen der Transit Gateways über verschiedene Regionen hinweg leiten.

Weitere Informationen finden Sie unter [Transit Gateways](#).

Zugreifen auf einen MemoryDB-Cluster, wenn er und der Amazon-EC2-Instance in verschiedenen Amazon VPCs aus verschiedenen Regions sind.

Verwenden von Transit VPC

Eine gängige Strategie zum Herstellen einer Verbindung zwischen vielen, geografisch verstreuten VPCs und Remote-Netzwerken besteht darin, eine Transit-VPC zu erstellen, die als globales Netzwerk-Transit-Zentrum fungiert. Eine Transit-VPC vereinfacht die Netzwerkverwaltung und minimiert die Anzahl von Verbindungen, die für die Vernetzung mehrerer VPCs mit Remote-Netzwerken erforderlich sind. Dieses Design kann Zeit und Aufwand verringern und auch Kosten reduzieren, da es virtuell ohne die herkömmlichen Ausgaben implementiert wird, die beim Einrichten

einer physischen Präsenz in einem Co-Location-Transit-Hub oder beim Bereitstellen physischer Netzwerkausstattung anfallen.

Herstellen einer Verbindung zwischen verschiedenen VPCs in verschiedenen Regionen

Nachdem die Transit-Amazon-VPC eingerichtet wurde, kann eine Anwendung, die in einer Spoke-VPC in einer Region bereitgestellt wird, eine Verbindung mit einem MemoryDB-Cluster in einer Spoke-VPC innerhalb einer anderen Region aufbauen.

So greifen Sie auf einen Cluster in einer anderen VPC innerhalb einer anderen AWS-Region zu

1. Stellen Sie eine Transit-VPC-Lösung bereit. Weitere Informationen finden Sie unter [AWS-Transit-Gateway](#).
2. Aktualisieren Sie die VPC-Routing-Tabellen in der App und in den VPCs, um den Datenverkehr durch das VGW (Virtual Private Gateway) und die VPN Appliance weiterzuleiten. Im Falle des dynamischen Routing mit Border Gateway Protocol (BGP) werden Ihre Routen möglicherweise automatisch gefüllt.
3. Bearbeiten Sie die Sicherheitsgruppe des MemoryDB-Clusters, um eingehende Verbindungen von dem IP-Bereich der Anwendungs-Instances zuzulassen. Beachten Sie, dass Sie in diesem Szenario nicht auf die Sicherheitsgruppe des Anwendungsservers verwiesen können.

Beim regionsübergreifenden Zugriff auf einen Cluster entstehen Netzwerklatenzen und fallen zusätzliche, regionsübergreifende Datenübertragungskosten an.

Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung

Ein weiteres mögliches Szenario ist eine hybride Architektur, in der Clients oder Anwendungen im Rechenzentrum eines Kunden möglicherweise auf einen MemoryDB-Cluster in der VPC zugreifen müssen. Dieses Szenario wird auch unterstützt, vorausgesetzt, dass entweder über VPN oder Direct Connect Konnektivität zwischen der VPC des Kunden und dem Rechenzentrum besteht.

Themen

- [Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels VPN-Konnektivität](#)
- [Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels Direkt Connect](#)

Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels VPN-Konnektivität

Herstellen einer Verbindung zu MemoryDB von Ihrem Rechenzentrum über ein VPN

So greifen Sie auf einen Cluster in einer VPC von einer Anwendung vor Ort über eine VPN-Verbindung zu

1. Richten Sie VPN-Konnektivität ein, indem Sie ein Hardware Virtual Private Gateway zu Ihrer VPC hinzufügen. Weitere Informationen finden Sie unter [Hinzufügen eines Hardware Virtual Private Gateway zu Ihrer VPC](#).
2. Aktualisieren Sie die VPC-Routing-Tabelle für das Subnetz, in dem Ihr MemoryDB-Cluster bereitgestellt wird, um Datenverkehr von Ihrem Anwendungsserver vor Ort zuzulassen. Im Falle des dynamischen Routing mit BGP werden Ihre Routen möglicherweise automatisch gefüllt.
3. Bearbeiten Sie die Sicherheitsgruppe des MemoryDB-Clusters, um eingehende Verbindungen von lokalen Anwendungsservern zuzulassen.

Beim Zugriff auf einen Cluster über eine VPN-Verbindung entstehen Netzwerklatenzen und fallen zusätzliche Datenübertragungskosten an.

Zugreifen auf einen MemoryDB-Cluster über eine in einem Kunden-Rechenzentrum ausgeführte Anwendung mittels Direkt Connect

Herstellen einer Verbindung zu MemoryDB von Ihrem Rechenzentrum über ein Direct Connect

So greifen Sie auf einen MemoryDB-Cluster über eine in Ihrem Netzwerk ausgeführte Anwendung mittels Direct Connect zu

1. Richten Sie Direct Connect-Konnektivität ein. Weitere Informationen finden Sie unter [Erste Schritte mit AWS-Direct-Connect](#).
2. Bearbeiten Sie die Sicherheitsgruppe des MemoryDB-Clusters, um eingehende Verbindungen von lokalen Anwendungsservern zuzulassen.

Beim Zugriff auf einen Cluster über eine DX-Verbindung können Netzwerklatenzen entstehen und zusätzliche Datenübertragungskosten anfallen.

Erstellen einer Virtual Private Cloud (VPC)

In diesem Beispiel erstellen Sie eine virtuelle private Cloud (VPC) auf der Grundlage des Amazon VPC-Service mit einem privaten Subnetz für jede Availability Zone.

Erstellen einer VPC (Konsole)

So erstellen Sie einen MemoryDB-Cluster innerhalb einer Amazon Virtual Private Cloud

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon VPC-Konsole unter <https://console.aws.amazon.com/vpc/>.
2. Wählen Sie im VPC-Dashboard Erstellen einer VPC aus.
3. Wählen Sie unter Resources to create (Zu erstellende Ressourcen) die Option VPC and more (VPC und mehr) aus.
4. Wählen Sie die Anzahl der Availability Zones (AZs) aus, in denen Sie Ihre Subnetze starten möchten.
5. Wählen Sie die Anzahl der öffentlichen Subnetze aus, die Sie zu Ihrer VPC hinzufügen möchten.
6. Wählen Sie die Anzahl der privaten Subnetze aus, die Sie zu Ihrer VPC hinzufügen möchten.

Tip

Notieren Sie sich Ihre Subnetz-IDs und welches öffentlich und welches privat ist. Sie benötigen diese Informationen später, wenn Sie Ihre Cluster starten und Ihrer Amazon VPC eine Amazon-EC2-Instance hinzufügen.

7. Erstellen Sie eine Amazon-VPC-Sicherheitsgruppe. Sie werden diese Gruppe für Ihren Cluster und Ihre Amazon EC2 Instance verwenden.
 - a. In der linken Navigationsleiste der AWS Management Console, wählen Sie Sicherheitsgruppen aus.
 - b. Wählen Sie Sicherheitsgruppen erstellen aus.
 - c. Geben Sie in den entsprechenden Feldern einen Namen und eine Beschreibung für Ihre Sicherheitsgruppe ein. Für VPC wählen Sie die ID für Ihre VPC aus.
 - d. Wenn Sie die gewünschten Einstellungen vorgenommen haben, wählen Sie Ja, erstellen aus.

8. Definieren Sie eine Netzwerkeingangsregel für Ihre Sicherheitsgruppe. Diese Regel wird Ihnen ermöglichen, mithilfe von Secure Shell (SSH) eine Verbindung zu Ihrer Amazon-EC2-Instance herzustellen.
 - a. Klicken Sie im linken Navigationsbereich auf Security Groups.
 - b. Suchen Sie Ihre Sicherheitsgruppe in der Liste und wählen Sie sie aus.
 - c. Wählen Sie unter Security Group die Registerkarte Inbound aus. Wählen Sie im Feld Create a new rule die Option SSH und anschließend Add Rule aus.

Stellen Sie die folgenden Werte für Ihre neue eingehende Regel ein, um HTTP-Zugriff zuzulassen:

- Type: HTTP
 - Quelle: 0.0.0.0/0
- d. Stellen Sie die folgenden Werte für Ihre neue eingehende Regel ein, um HTTP-Zugriff zuzulassen:
 - Type: HTTP
 - Quelle: 0.0.0.0/0

Wählen Sie Apply Rule Changes aus.

Jetzt können Sie eine erstellen [-Subnetzgruppe](#) und [Erstellen eines Clusters](#) in Ihrer VPC.

Subnetze und Subnetzgruppen

Eine Subnetzgruppe ist eine Sammlung von Subnetzen (in der Regel private Subnetze), die Sie für Ihre, in einer Amazon Virtual Private Cloud (VPC)-Umgebung ausgeführten, Cluster festlegen können.

Wenn Sie einen Cluster in einer Amazon VPC erstellen, können Sie eine Subnetzgruppe angeben oder die bereitgestellte Standardgruppe verwenden. MemoryDB verwendet diese Subnetzgruppe, um ein Subnetz und IP-Adressen innerhalb dieses Subnetzes auszuwählen, die Ihren Knoten zugeordnet werden sollen.

In diesem Abschnitt wird beschrieben, wie Sie Subnetze und Subnetzgruppen erstellen und nutzen, um den Zugriff auf Ihre MemoryDB-Ressourcen zu verwalten.

Weitere Informationen zur Verwendung von Subnetzgruppen in einer Amazon-VPC-Umgebung finden Sie unter [Schritt 2: Autorisieren Sie den Zugriff auf den Cluster](#).

Unterstützte MemoryDB AZ-IDs

Regionsname/Region	Unterstützte AZ-IDs		
Region USA Ost (Ohio) us-east-2	use2-az1, use2-az2, use2-az3		
Region USA Ost (Nord-Virginia) us-east-1	use1-az2, use1-az4, use1-az6		
Region US West (N. California) us-west-1	usw1-az1, usw1-az2, usw1-az3		
Region USA West (Oregon) us-west-2	usw2-az1, usw2-az2, usw2-az3		

Regionsname/Region	Unterstützte AZ-IDs		
Region Kanada (Zentral) ca-central-1	cac1-az1, cac1-az2, cac1-az4		
Region Asien-Pazifik (Hongkong) ap-east-1	ape1-az1, ape1-az2, ape1-az3		
Region Asien-Pazifik (Mumbai) ap-south-1	aps1-az1, aps1-az2, aps1-az3		
Region Asien-Pazifik (Tokio) ap-northeast-1	apne1-az1, apne1-az2, apne1-az4		
Asia Pacific (Seoul) Region ap-northeast-2	apne2-az1, apne2-az2, apne2-az3		
Region Asien-Pazifik (Singapur) ap-southeast-1	apse1-az1, apse1-az2, apse1-az3		
Region Asien-Pazifik (Sydney) ap-southeast-2	apse2-az1, apse2-az2, apse2-az3		
Region Europa (Frankfurt) eu-central-1	euc1-az1, euc1-az2, euc1-az3		

Regionsname/Region	Unterstützte AZ-IDs		
Region Europa (Irland) eu-west-1	euw1-az1, euw1-az2, euw1-az3		
Region Europa (London) eu-west-2	euw2-az1, euw2-az2, euw2-az3		
Region Europa (Paris) eu-west-3	euw3-az1, euw3-az2, euw3-az3		
Region Europa (Stockholm) eu-north-1	eun1-az1, eun1-az2, eun1-az3		
Region Europa (Mailand) eu-south-1	eus1-az1, eus1-az2, eus1-az3		
Region Südamerika (São Paulo) sa-east-1	sae1-az1, sae1-az2, sae1-az3		
Region China (Peking) cn-north-1	cnn1-az1, cnn1-az2		
Region China (Ningxia) cn-northwest-1	cnw1-az1, cnw1-az2, cnw1-az3		

Themen

- [Erstellen einer Subnetzgruppe](#)
- [Aktualisierung einer Subnetzgruppe](#)
- [Details zur Subnetzgruppe anzeigen](#)
- [Löschen einer Subnetzgruppe](#)

Erstellen einer Subnetzgruppe

Wenn Sie eine neue Subnetzgruppe erstellen, notieren Sie sich die Anzahl der verfügbaren IP-Adressen. Wenn das Subnetz nur über wenige freie IP-Adressen verfügt, beschränkt dies auch die Anzahl der neuen Knoten, die Sie zu dem Cluster hinzufügen können. Um dieses Problem zu lösen, können Sie einer Subnetzgruppe weitere Subnetze zuweisen, um ausreichend IP-Adressen in der Availability Zone Ihres Clusters bereitzustellen. Danach können Sie dem Cluster weitere Knoten hinzufügen.

Die folgenden Verfahren zeigen Ihnen, wie Sie eine Subnetzgruppe mit den Namen `mysubnetgroup` (Konsole) AWS CLI, die und die MemoryDB-API erstellen.

Erstellen einer Subnetzgruppe (Konsole)

Im folgenden Verfahren wird das Erstellen einer Subnetzgruppe (Konsole) erläutert.

Erstellen einer DB-Sicherheitsgruppe (Konsole)

1. [Melden Sie sich bei der AWS Management Console an und öffnen Sie die MemoryDB-Konsole unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Subnet Groups aus.
3. Klicken Sie auf Create Subnet Group (Subnetzgruppe ändern).
4. Gehen Sie auf der Seite „Subnetzgruppe erstellen“ wie folgt vor:
 - a. Geben Sie im Feld Name einen Namen für Ihre Subnetzgruppe ein.

Für die Benennung von Clustern gelten die folgenden Einschränkungen:

- Er muss 1-40 alphanumerische Zeichen oder Bindestriche enthalten.
 - Er muss mit einem Buchstaben beginnen.
 - Er darf keine zwei aufeinanderfolgenden Bindestriche enthalten.
 - Er darf nicht mit einem Bindestrich enden.
- b. Geben Sie im Feld Description eine Beschreibung für Ihre Subnetzgruppe ein.
 - c. Wählen Sie im Feld VPC ID die erstellte Amazon VPC aus. Wenn Sie noch keine erstellt haben, klicken Sie auf die Schaltfläche VPC erstellen und folgen Sie den Schritten, um eine zu erstellen.
 - d. Wählen Sie unter Ausgewählte Subnetze die Availability Zone und die ID Ihres privaten Subnetzes aus und klicken Sie dann auf Auswählen.

5. Für Tags können Sie optional Tags anwenden, um Ihre Subnetze zu durchsuchen und zu filtern oder Ihre Kosten zu verfolgen. AWS
6. Wenn Sie die gewünschten Einstellungen vorgenommen haben, wählen Sie Erstellen aus.
7. Klicken Sie in der angezeigten Bestätigungsmeldung auf Close.

Ihre neue Subnetzgruppe wird in der Liste der Subnetzgruppen der MemoryDB-Konsole angezeigt. Unten im Fenster können Sie die Subnetzgruppe auswählen, um Details wie die der Gruppe zugeordneten Subnetze anzuzeigen.

Erstellen einer Subnetzgruppe (AWS CLI)

Geben Sie in einem Befehlszeilenfenster den Befehl `create-subnet-group` ein, um eine Subnetzgruppe zu erstellen.

Für Linux, macOS oder Unix:

```
aws memorydb create-subnet-group \  
  --subnet-group-name mysubnetgroup \  
  --description "Testing" \  
  --subnet-ids subnet-53df9c3a
```

Für Windows:

```
aws memorydb create-subnet-group ^  
  --subnet-group-name mysubnetgroup ^  
  --description "Testing" ^  
  --subnet-ids subnet-53df9c3a
```

Die Ausgabe dieses Befehls sieht ähnlich wie folgt aus:

```
{  
  "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"
  }
}
```

Weitere Informationen finden Sie im AWS CLI Thema [create-subnet-group](#).

Eine Subnetzgruppe erstellen (MemoryDB-API)

Rufen `CreateSubnetGroup` Sie mithilfe der MemoryDB-API mit den folgenden Parametern auf:

- `SubnetGroupName`=*mysubnetgroup*
- `Description`=*Testing*
- `SubnetIds.member.1`=*subnet-53df9c3a*

Aktualisierung einer Subnetzgruppe

Sie können die Beschreibung einer Subnetzgruppe aktualisieren oder die Liste der Subnetz-IDs ändern, die der Subnetzgruppe zugeordnet sind. Es ist nicht möglich, Subnetz-IDs aus einer Subnetzgruppe zu löschen, wenn das Subnetz derzeit von einem Cluster verwendet wird.

Die folgenden Verfahren zeigen Ihnen, wie Sie eine Subnetzgruppe aktualisieren.

Subnetzgruppen aktualisieren (Konsole)

Um eine Subnetzgruppe zu aktualisieren

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Subnet Groups aus.
3. Wählen Sie in der Liste der Subnetzgruppen die gewünschte Subnetzgruppe aus.
4. Die Felder Name, VpcID und Beschreibung können nicht geändert werden.
5. Klicken Sie im Abschnitt Ausgewählte Subnetze auf Verwalten, um alle Änderungen an den Availability Zones vorzunehmen, die Sie für die Subnetze benötigen. Klicken Sie auf Save (Speichern), um die Änderungen zu speichern.

Aktualisierung von Subnetzgruppen (AWS CLI)

Verwenden Sie in einer Befehlszeile den Befehl, `update-subnet-group` um eine Subnetzgruppe zu aktualisieren.

Für Linux, macOS oder Unix:

```
aws memorydb update-subnet-group \  
  --subnet-group-name mysubnetgroup \  
  --description "New description" \  
  --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Für Windows:

```
aws memorydb update-subnet-group ^  
  --subnet-group-name mysubnetgroup ^  
  --description "New description" ^  
  --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Die Ausgabe dieses Befehls sieht ähnlich wie folgt aus:

```
{
  "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",
  }
}
```

Weitere Informationen finden Sie im AWS CLI Thema [update-subnet-group](#).

Aktualisierung von Subnetzgruppen (MemoryDB-API)

Rufen UpdateSubnetGroup Sie mithilfe der MemoryDB-API mit den folgenden Parametern auf:

- SubnetGroupName=*mysubnetgroup*
- Alle anderen Parameter, deren Werte Sie ändern möchten. In diesem Beispiel wird Description=*New%20description* verwendet, um die Beschreibung der Subnetzgruppe zu ändern.

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

Wenn Sie eine neue Subnetzgruppe erstellen, notieren Sie sich die Anzahl der verfügbaren IP-Adressen. Wenn das Subnetz nur über wenige freie IP-Adressen verfügt, beschränkt dies auch die Anzahl der neuen Knoten, die Sie zu dem Cluster hinzufügen können. Um dieses Problem zu lösen, können Sie einer Subnetzgruppe weitere Subnetze zuweisen, um ausreichend IP-Adressen in der Availability Zone Ihres Clusters bereitzustellen. Danach können Sie dem Cluster weitere Knoten hinzufügen.

Details zur Subnetzgruppe anzeigen

Die folgenden Verfahren zeigen Ihnen, wie Sie Details zu einer Subnetzgruppe anzeigen.

Details von Subnetzgruppen anzeigen (Konsole)

Um Details einer Subnetzgruppe anzuzeigen (Konsole)

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/memorydb/.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Subnet Groups aus.
3. Wählen Sie auf der Seite Subnetzgruppen die Subnetzgruppe unter Name aus, oder geben Sie den Namen der Subnetzgruppe in die Suchleiste ein.
4. Wählen Sie auf der Seite Subnetzgruppen die Subnetzgruppe unter Name aus, oder geben Sie den Namen der Subnetzgruppe in die Suchleiste ein.

5. Unter Subnetzgruppeneinstellungen können Sie den Namen, die Beschreibung, die VPC-ID und den Amazon-Ressourcennamen (ARN) der Subnetzgruppe einsehen.
6. Unter Subnetze können Sie die Availability Zones, Subnetz-IDs und CIDR-Blöcke der Subnetzgruppe einsehen
7. Unter Tags können Sie alle Tags anzeigen, die der Subnetzgruppe zugeordnet sind.

Details zu Subnetzgruppen anzeigen (AWS CLI)

Verwenden Sie in einer Befehlszeile den Befehl, `describe-subnet-groups` um die Details einer bestimmten Subnetzgruppe anzuzeigen.

Für Linux, macOS oder Unix:

```
aws memorydb describe-subnet-groups \  
  --subnet-group-name mysubnetgroup
```

Für Windows:

```
aws memorydb describe-subnet-groups ^  
  --subnet-group-name mysubnetgroup
```

Die Ausgabe dieses Befehls sieht ähnlich wie folgt aus:

```
{  
  "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"
}
```

Um Details zu allen Subnetzgruppen anzuzeigen, verwenden Sie denselben Befehl, jedoch ohne Angabe eines Subnetzgruppennamens.

```
aws memorydb describe-subnet-groups
```

Weitere Informationen finden Sie im AWS CLI Thema. [describe-subnet-groups](#)

Subnetzgruppen anzeigen (MemoryDB-API)

Rufen `DescribeSubnetGroups` Sie mithilfe der MemoryDB-API mit den folgenden Parametern auf:

`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
```

Löschen einer Subnetzgruppe

Wenn Sie eine Subnetzgruppe nicht mehr benötigen, können Sie sie löschen. Sie können eine Subnetzgruppe, die derzeit von einem Cluster verwendet wird, nicht löschen. Sie können auch eine Subnetzgruppe in einem Cluster mit aktiviertem Multi-AZ nicht löschen, wenn dieser Cluster mit weniger als zwei Subnetzen belässt. Sie müssen zuerst Multi-AZ deaktivieren und dann das Subnetz löschen.

Das folgende Verfahren zeigt, wie Sie eine Subnetzgruppe löschen.

Löschen einer Subnetzgruppe (Konsole)

So löschen Sie eine Subnetzgruppe

1. [Melden Sie sich bei der MemoryDB for Redis-Konsole an AWS Management Console und öffnen Sie sie unter `https://console.aws.amazon.com/memorydb/`.](https://console.aws.amazon.com/memorydb/)
2. Wählen Sie im linken Navigationsbereich Subnet Groups aus.
3. Wählen Sie in der Liste der Subnetzgruppen die aus, die Sie löschen möchten, klicken Sie auf Aktionen und dann auf Löschen.

Note

Sie können keine Standard-Subnetzgruppe oder eine, die mit Clustern verknüpft ist, löschen.

4. Der Bestätigungsbildschirm „Subnetzgruppen löschen“ wird angezeigt.
5. Um die Subnetzgruppe zu löschen, geben Sie `delete` in das Bestätigungstextfeld ein. Wählen Sie Cancel (Abbrechen), um die Subnetzgruppe zu erhalten.

Löschen einer Subnetzgruppe (AWS CLI)

Rufen Sie mit dem AWS CLI den Befehl `delete-subnet-group` mit dem folgenden Parameter auf:

- `--subnet-group-name mysubnetgroup`

Für Linux, macOS oder Unix:

```
aws memorydb delete-subnet-group \
```

```
--subnet-group-name mysubnetgroup
```

Für Windows:

```
aws memorydb delete-subnet-group ^  
  --subnet-group-name mysubnetgroup
```

Weitere Informationen finden Sie im AWS CLI Thema [delete-subnet-group](#).

Löschen einer Subnetzgruppe (MemoryDB-API)

Rufen DeleteSubnetGroup Sie mithilfe der MemoryDB-API mit dem folgenden Parameter auf:

- 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
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie im Thema MemoryDB-API. [DeleteSubnetGroup](#)

VPC-Endpunkte MemoryDB für Redis-API und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)

Sie können eine private Verbindung zwischen Ihrer VPC und herstellen Amazon MemoryDB für Redis-API-Endpunkte durch Erstellen eines Schnittstellen-VPC-Endpunktaus.

Schnittstellenendpunkte werden unterstützt von [AWS PrivateLink](#). AWS PrivateLink erlaubt Sie können privat auf MemoryDB für Redis-API-Operationen ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder zugreifen AWS Direct Connect-Verbindung.

Die Instances in Ihrer VPC benötigen für die Kommunikation mit MemoryDB für Redis-API-Endpunkte keine öffentlichen IP-Adressen. Ihre Instances benötigen auch keine öffentlichen IP-Adressen, um einen der verfügbaren MemoryDB -API-Operationen zu verwenden. Der Datenverkehr zwischen Ihrer VPC und MemoryDB für Redis verlässt das Amazon-Netzwerk nicht. Jeder Schnittstellenendpunkt wird durch eine oder mehrere Elastic Network-Schnittstellen in Ihren Subnetzen dargestellt. Weitere Informationen zu Elastic Network-Schnittstellen finden Sie unter [Elastic Network-Schnittstellen](#) im Amazon EC2 Benutzerhandbuch.

- Weitere Informationen über VPC-Endpunkte finden Sie unter [Schnittstellen-VPC-Endpunkte \(AWS PrivateLink\)](#) im Amazon VPC User Guide aus.
- Weitere Informationen über MemoryDB -API-Operationen finden Sie unter [MemoryDB API-Operationen](#) aus.

Wenn Sie einen Schnittstellen-VPC-Endpunkt erstellt haben, wenn Sie [Private DNS](#) Hostnamen für den Endpunkt, der Standard-MemoryDB-Endpunkt (<https://memorydb.Region.amazonaws.com>) wird auf Ihren VPC-Endpunkt aufgelöst. Wenn Sie keine privaten DNS-Hostnamen aktiviert haben, stellt Amazon VPC einen DNS-Endpunktnamen bereit, den Sie im folgenden Format verwenden können:

```
VPC_Endpoint_ID.memorydb.Region.vpce.amazonaws.com
```

Weitere Informationen finden Sie unter [Schnittstellen-VPC-Endpunkte \(AWS PrivateLink\)](#) im Amazon VPC User Guide aus. MemoryDB unterstützt Aufrufe all seiner [API-Aktionen](#) in deiner VPC.

Note

Private DNS-Hostnamen können nur für einen VPC-Endpunkt in der VPC aktiviert werden. Wenn Sie einen zusätzlichen VPC-Endpunkt erstellen möchten, sollte der private DNS-Hostname dafür deaktiviert sein.

Überlegungen zu VPC-Endpunkten

Bevor Sie einen Schnittstellen-VPC-Endpunkt für MemoryDB für Redis-API-Endpunkte einrichten, stellen Sie sicher, dass Sie die Überprüfung [Eigenschaften und Beschränkungen von](#)

[Schnittstellenendpunkten](#) im Amazon VPC User Guide aus. Alle MemoryDB API-Operationen, die für die Verwaltung von MemoryDB für Redis-Ressourcen relevant sind, sind über Ihre VPC verfügbar. AWS PrivateLink aus. VPC-Endpunkttrichtlinien werden für MemoryDB -API-Endpunkte unterstützt. Standardmäßig ist der vollständige Zugriff auf MemoryDB -API-Operationen über den Endpunkt zulässig. Weitere Informationen finden Sie unter [Steuerung des Zugriffs auf Services mit VPC-Endpunkten](#) im Amazon-VPC-Benutzerhandbuch.

Erstellen eines Schnittstellen-VPC-Endpunkts für die MemoryDB API

Sie können einen VPC-Endpunkt für die MemoryDB für Redis-API mithilfe der Amazon VPC-Konsole oder der AWS CLI aus. Weitere Informationen finden Sie unter [Erstellung eines Schnittstellenendpunkts](#) im Benutzerhandbuch für Amazon VPC.

Nachdem Sie einen Schnittstellen-VPC-Endpunkt erstellt haben, können Sie private DNS-Hostnamen für den Endpunkt aktivieren. Wenn Sie dies tun, ist der Standard MemoryDB für Redis-Endpunkt (<https://memorydb.Region.amazonaws.com>) wird auf Ihren VPC-Endpunkt aufgelöst. Weitere Informationen finden Sie unter [Zugriff auf einen Service über einen Schnittstellenendpunkt](#) im Benutzerhandbuch für Amazon VPC.

Erstellen einer VPC-Endpunkttrichtlinie für die Amazon MemoryDB API

Sie können eine Endpunkttrichtlinie an Ihren VPC-Endpunkt anhängen, der den Zugriff auf die MemoryDB -API steuert. Die Richtlinie legt Folgendes fest:

- Prinzipal, der die Aktionen ausführen kann
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter [Steuerung des Zugriffs auf Services mit VPC-Endpunkten](#) im Amazon-VPC-Benutzerhandbuch.

Example VPC-Endpunkttrichtlinie für MemoryDB -API-Aktionen

Im Folgenden finden Sie ein Beispiel für eine Endpunkttrichtlinie für die MemoryDB -API. Wenn diese Richtlinie an einen Endpunkt angefügt wird, gewährt sie Zugriff auf die aufgelisteten MemoryDB -API-Aktionen für alle Prinzipale auf allen Ressourcen.

```
{
  "Statement": [{
    "Principal": "*",
```

```

"Effect": "Allow",
"Action": [
  "memorydb:CreateCluster",
  "memorydb:UpdateCluster",
  "memorydb:CreateSnapshot"
],
"Resource": "*"
}]
}

```

Example VPC-Endpunktrichtlinie, die den gesamten Zugriff über eine angegebene verweigertAWSKonto

Die folgende VPC-Endpunktrichtlinie verweigert dem AWS-Konto **123456789012** jeglichen Zugriff auf Ressourcen, die den Endpunkt verwenden. Die Richtlinie erlaubt alle Aktionen von anderen Konten.

```

{
  "Statement": [{
    "Action": "*",
    "Effect": "Allow",
    "Resource": "*",
    "Principal": "*"
  },
  {
    "Action": "*",
    "Effect": "Deny",
    "Resource": "*",
    "Principal": {
      "AWS": [
        "123456789012"
      ]
    }
  }
]
}

```

Service-Updates in MemoryDB für Redis

MemoryDB for Redis überwacht automatisch die Flotte an Clustern und Knoten, um Service-Updates anzuwenden, sobald sie verfügbar werden. In der Regel richten Sie ein vordefiniertes

Wartungsfenster ein, damit MemoryDB diese Updates anwenden kann. In einigen Fällen könnte dieser Ansatz jedoch zu starr sein und möglicherweise Ihre Geschäftsabläufe einschränken.

Mit Service-Updates steuern Sie, wann und welche Updates angewendet werden. Sie können außerdem den Fortschritt dieser Updates des ausgewählten MemoryDB-Clusters in Echtzeit überwachen.

Verwalten der Service-Updates

Die MemoryDB-Service-Updates werden regelmäßig veröffentlicht. Wenn Sie mindestens einen qualifizierten Cluster für diese Service-Updates haben, erhalten Sie Benachrichtigungen per E-Mail, SNS, Personal Health Dashboard (PHD) und Amazon CloudWatch CloudWatch-Ereignisse, sobald die Updates veröffentlicht werden. Die Updates werden auch auf der Service-Updates-Seite auf der MemoryDB-Konsole. Mit diesem Dashboard können Sie alle Service-Updates und deren Status für Ihre MemoryDB-Flotte anzeigen.

Sie steuern, wann ein Update angewendet wird, bevor eine automatische Aktualisierung gestartet wird. Wir empfehlen dringend, alle Updates des Typs anzuwenden Security-Aktualisierung so schnell wie möglich, um sicherzustellen, dass Ihre MemoryDB immer mit den aktuellen Sicherheits-Patches aktualisiert wird.

In den folgenden Abschnitten werden diese Optionen ausführlich erörtert.

Themen

- [Anwenden der Dienstaktualisierungen](#)

Anwenden der Dienstaktualisierungen

Sie können Service-Updates auf die Flotte anwenden, sobald diese Updates verfügbar sind. Service-Updates sind kumulativ. Mit anderen Worten, alle Updates, die Sie noch nicht angewendet haben, sind in Ihrem neuesten Update enthalten.

Wenn für ein Dienstupdate die automatische Aktualisierung aktiviert ist, können Sie keine Maßnahmen ergreifen, wenn es verfügbar wird. MemoryDB plant, das Update während des Wartungsfensters Ihrer Cluster nach dem Startdatum automatisch auszuführen. Sie erhalten zugehörige Benachrichtigungen für jede Phase des Updates.

 Note

Sie können nur die Serviceupdates anwenden, die über einen `availableOrPlannedStatus`.

Für weitere Informationen zum Überprüfen und Anwenden Service-spezifischer Updates auf relevante MemoryDB-Cluster siehe [Anwenden der Service-Updates mithilfe der Konsole](#) aus.

Wenn für mindestens einen MemoryDB-Cluster ein neues Service-Update verfügbar ist, können Sie die MemoryDB-Konsole, -API oder AWS CLI um das Update anzuwenden. Die folgenden Abschnitte beschreiben die Optionen, die zum Anwenden von Updates genutzt werden können.

Anwenden der Service-Updates mithilfe der Konsole

Um die Liste der verfügbaren Service-Updates sowie weitere Informationen anzuzeigen, gehen Sie zur `Service-Updates` Seite in der -Konsole.

1. Melden Sie sich bei der AWS Management Console und öffnen Sie die MemoryDB for Redis-Konsole unter <https://console.aws.amazon.com/memorydb/> aus.
2. Klicken Sie im Navigationsbereich auf `Service-Updates` aus.

UNTER Details zum Service-Update Folgendes kann angezeigt werden:

- Name des Service-Updates: Der eindeutige Name des Service-Updates
- Beschreibung aktualisieren: Detaillierte Informationen zum Service-Update
- Startdatum automatisch aktualisieren: Wenn dieses Attribut festgelegt ist, beginnt MemoryDB mit der Planung, dass Ihre Cluster nach diesem Datum in den entsprechenden Wartungsfenstern automatisch aktualisiert werden. Sie erhalten im Voraus Benachrichtigungen über das genaue geplante Wartungsfenster, das möglicherweise nicht unmittelbar nach dem Startdatum automatisch aktualisieren aus. Sie können das Update weiterhin jederzeit auf Ihre Cluster anwenden. Wenn das Attribut nicht festgelegt ist, ist das Dienstupdate nicht automatisch aktiviert und MemoryDB aktualisiert Ihre Cluster nicht automatisch.

In der `Clusteraktualisierungsstatus` können Sie eine Liste von Clustern anzeigen, bei denen das Service-Update nicht angewendet wurde oder gerade erst angewendet wurde. Für jeden Cluster können Sie Folgendes anzeigen:

- Clustername: Der Name des Clusters

- **Aktualisierte Knoten:** Das Verhältnis der einzelnen Knoten in einem bestimmten Cluster, die aktualisiert wurden oder weiterhin für das betreffende Service-Update verfügbar sind.
- **Typ aktualisieren:** Der Typ des Service-Updates, also einer von Security-Aktualisierung oder Motor-Aktualisierung
- **Status:** Der Status des Service-Updates im Cluster, also einer der folgenden Werte:
 - **verfügbar:** Das Update ist für den relevanten Cluster verfügbar.
 - **In-Progress:** Das Update wird auf diesen Cluster angewendet.
 - **geplant:** Das Aktualisierungsdatum wurde geplant.
 - **Komplett:** Das Update wurde erfolgreich angewendet. Cluster mit einem vollständigen Status wird nach seiner Fertigstellung 7 Tage lang angezeigt.

Wenn Sie einen oder alle Cluster mit dem verfügbar oder geplant status, und dann wählen Sie jetzt bewerben wird das Update auf diese Cluster angewendet.

Anwenden der Service-Updates mit dem AWS CLI

Nachdem Sie benachrichtigt wurden, dass Service-Updates verfügbar sind, können Sie sie mit der AWS CLI inspizieren und anwenden:

- Führen Sie den folgenden Befehl aus, um eine Beschreibung der verfügbaren Serviceaktualisierungen abzurufen:

```
aws memorydb describe-service-updates --status available
```

Weitere Informationen finden Sie unter [beschreiben-Service-Updates](#) aus.

- Führen Sie den folgenden Befehl aus, um ein Service-Update auf eine Clusterliste anzuwenden:

```
aws memorydb batch-update-cluster --service-update  
ServiceUpdateNameToApply=sample-service-update --cluster-names cluster-1  
cluster2
```

Weitere Informationen finden Sie unter [Batch-Update-Cluster](#) aus.

Referenz

Die Themen in diesem Abschnitt beschreiben die Arbeit mit der MemoryDB -API und dem -Abschnitt derAWS CLIaus. In diesem Abschnitt sind auch allgemeine Fehlermeldungen und Servicemeldungen enthalten.

- [Verwenden der MemoryDB-API](#)
- [MemoryDB - API-Referenz](#)
- [MemoryDB -Abschnitt imAWS CLI--Referenz](#)

Verwenden der MemoryDB-API

Dieser Abschnitt enthält aufgabenorientierte Beschreibungen der Verwendung und Implementierung von MemoryDB-Operationen. Eine vollständige Beschreibung dieser Operationen finden Sie unter [MemoryDB API-Referenz](#) aus.

Themen

- [Verwenden der Abfrage-API](#)
- [Verfügbare Bibliotheken](#)
- [Fehlerbehebung bei Anwendungen](#)

Verwenden der Abfrage-API

Abfrageparameter

HTTP-Query-basierte Anfragen sind HTTP-Anfragen, die das HTTP-Verb GET oder POST und einen Query-Parameter namens `action` verwenden.

Jede Query-Anfrage muss einige allgemeine Parameter enthalten, um die Authentifizierung und Auswahl einer Aktion zu bearbeiten.

Einige Operationen verwenden Parameterlisten. Diese Listen werden mit der Notation `param.n` definiert. Werte von `n` sind Ganzzahlen ab 1.

Authentifizierung von Abfrageanforderungen

Sie können Abfrageanforderungen nur über HTTPS senden und müssen in jede Abfrageanforderung eine Signatur einschließen. In diesem Abschnitt wird beschrieben, wie Sie die Signatur erstellen. Die in den folgenden Schritten beschriebene Methode wird als Signaturversion 4 bezeichnet.

Die folgenden grundlegenden Schritte dienen der Authentifizierung von Anfragen an AWS. Dieser Vorgang setzt voraus, dass Sie bei AWS registriert sind und eine Zugriffsschlüssel-ID sowie einen geheimen Zugriffsschlüssel besitzen.

Abfrageauthentifizierungsprozess

1. Der Sender erstellt eine Anforderung an AWS.

2. Der Sender berechnet die Anforderungssignatur, ein Keyed-Hashing for Hash-based Message Authentication Code (HMAC) mit einer SHA-1-Hash-Funktion, wie im nächsten Abschnitt dieses Themas beschrieben.
3. Der Sender der Anforderung sendet die Anforderungsdaten, die Signatur und die Zugriffsschlüssel-ID (den Schlüsselbezeichner des verwendeten geheimen Zugriffsschlüssels) an AWS.
4. AWS verwendet die Zugriffsschlüssel-ID, um den geheimen Zugriffsschlüssel zu ermitteln.
5. AWS erstellt anhand der Anforderungsdaten und des geheimen Zugriffsschlüssels eine Signatur und verwendet dabei denselben Algorithmus, den Sie zum Berechnen der Signatur in der Anforderung verwendet haben.
6. Wenn die Signaturen übereinstimmen, wird die Anforderung als authentisch betrachtet. Falls der Vergleich fehlschlägt, wird die Anforderung verworfen, und AWS gibt eine Fehlerantwort zurück.

 Note

Wenn eine Anforderung einen `Timestamp`-Parameter enthält, läuft die für die Anforderung berechnete Signatur 15 Minuten nach dessen Wert ab.

Wenn eine Anforderung einen `Expires`-Parameter enthält, läuft die Signatur an dem durch den `Expires`-Parameter angegebenen Zeitpunkt ab.

So wird die Anfragesignatur berechnet

1. Erstellen Sie eine vereinheitlichte Abfragezeichenfolge, die Sie später bei dieser Prozedur benötigen:
 - a. Sortieren Sie die UTF-8-Abfragezeichenkomponenten nach Parameternamen in natürlicher Bytereihenfolge. Die Parameter können von einer GET-URI oder von einem POST-Test stammen (wenn `Content-Type = application/x-www-form-urlencoded`).
 - b. Führen Sie eine URL-Codierung des Parameternamens und -werts nach folgenden Regeln durch:
 - i. Führen Sie keine URL-Codierung der nicht reservierten, von RFC 3986 definierten Zeichen durch. Folgende Zeichen sind nicht reserviert: A – Z, a – z, 0 – 9, Bindestrich (-), Unterstrich (_), Punkt (.)

- ii. Versehen Sie alle anderen Zeichen mit Prozentcode (%XY), wobei X und Y für Hexadezimalzeichen, d. h. 0-9 und die Großbuchstaben A-F, steht.
 - iii. Versehen Sie alle erweiterten UTF-8-Zeichen mit Prozentcode im Format %XY%ZA....
 - iv. Versehen Sie das Leerzeichen mit dem Prozentcode %20 (und nicht + wie in herkömmlichen Codierungsschemata).
- c. Trennen Sie die codierten Parameternamen mit dem Gleichzeichen (=) (ASCII-Zeichen 61) von den zugehörigen codierten Werten, auch wenn der Parameterwert leer ist.
 - d. Trennen Sie diese Namen-Wert-Paare durch ein kaufmännisches Und (&) (ASCII-Code 38).
2. Erstellen Sie die Zeichenfolge für die Signatur anhand der folgenden Pseudogrammatik (das "\n" steht in ASCII für eine neue Zeile).

```
StringToSign = HTTPVerb + "\n" +  
ValueOfHostHeaderInLowercase + "\n" +  
HTTPRequestURI + "\n" +  
CanonicalizedQueryString <from the preceding step>
```

Die HTTPRequestURI-Komponente ist die absolute HTTP-Pfadkomponente des URI bis zur, jedoch nicht einschließlich der Zeichenfolge. Falls HTTPRequestURI leer ist, verwenden Sie den Schrägstrich (/).

3. Berechnen Sie eine RFC 2104-kompatible HMAC-Signatur, wobei Sie die soeben erstellte Zeichenfolge, den geheimen Zugriffsschlüssel als Schlüssel und SHA256 oder SHA1 als Hashalgorithmus verwenden.

Weitere Informationen finden Sie unter <https://www.ietf.org/rfc/rfc2104.txt>.

4. Konvertieren Sie den resultierenden Wert zu base64.
5. Schließen Sie den Wert als Wert des Signature-Parameters in die Anforderung ein.

Das folgende Beispiel zeigt eine Anforderung (die Zeilenumbrüche wurden aus Gründen der Übersichtlichkeit hinzugefügt).

```
https://memory-db.us-east-1.amazonaws.com/  
?Action=DescribeClusters  
&ClusterName=myCluster  
&SignatureMethod=HmacSHA256  
&SignatureVersion=4
```

```
&Version=2021-01-01
```

Für die vorangehende Abfragezeichenfolge berechnen Sie die HMAC-Signatur über die folgende Zeichenfolge.

```
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:
```

Das Ergebnis ist die folgende signierte Anforderung.

```
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
```

Ausführliche Informationen zum Signieren und Berechnen der Anforderungssignatur finden Sie im Thema [Signierung mit Signatur Version 4](#) und seine Unterthemen.

Verfügbare Bibliotheken

AWS stellt Software Development Kits (SDKs) für Softwareentwickler bereit, die Anwendungen bevorzugt unter Verwendung sprachspezifischer APIs statt der Abfrage-API entwickeln.

Diese SDKs stellen grundlegende Funktionen bereit, die nicht in den APIs enthalten sind, wie Anforderungsauthentifizierung, Anforderungswiederholungen und Fehlerbehandlung, damit die ersten Schritte einfacher sind. SDKs und zusätzliche Ressourcen sind für die folgenden Programmiersprachen verfügbar:

- [Java](#)
- [Windows und .NET](#)
- [PHP](#)
- [Python](#)
- [Ruby](#)

Weitere Informationen zu anderen Sprachen finden Sie unter [Beispiel-Code und Bibliotheken](#) aus.

Fehlerbehebung bei Anwendungen

MemoryDB stellt spezifische und beschreibende Fehlermeldungen bereit, um Sie bei der Behebung von Problemen während der Interaktion mit der MemoryDB-API zu unterstützen.

Fehler bei Abrufen

In der Regel sollte Ihre Anwendung überprüfen, ob eine Anforderung einen Fehler verursacht hat, bevor Sie Zeit für die Verarbeitung von Ergebnissen aufwenden. Die einfachste Möglichkeit, herauszufinden, ob ein Fehler aufgetreten ist, besteht darin, nach einem `ERROR`-Knoten in der Antwort aus der MemoryDB-API.

Die XPath-Syntax bietet eine einfache Möglichkeit, nach einem `ERROR`-Knoten zu suchen. Darüber hinaus vereinfacht sie den Abruf von Fehlercode und Fehlermeldung. Der folgende Codeausschnitt verwendet Perl und das `XML::XPath`-Modul, um zu ermitteln, ob während einer Anfrage ein Fehler aufgetreten ist. Wenn ein Fehler aufgetreten ist, gibt der Code den ersten Fehlercode und die erste Fehlermeldung in der Antwort an.

```
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: ",
$xml->findvalue("//Error[1]/Code"), "\n", " ",
$xml->findvalue("//Error[1]/Message"), "\n\n"; }
```

Tipps zur Problembehebung

Die folgenden Prozesse werden empfohlen, um Probleme mit der MemoryDB-API zu diagnostizieren und zu beheben.

- Überprüfen Sie, dass MemoryDB ordnungsgemäß ausgeführt wird.

Öffnen Sie dazu ein Browserfenster und senden Sie eine Abfrageanforderung an den MemoryDB-Dienst (z. B. <https://memory-db.us-east-1.amazonaws.com>). Eine `MissingAuthenticationTokenException` oder `Un-KnowNoPerationException` bestätigt, dass der Dienst verfügbar ist und auf Anforderungen reagiert.

- Überprüfen Sie die Struktur Ihrer Anforderung.

Jede MemoryDB-Operation verfügt über eine Referenzseite im MemoryDB API-Referenzaus. Prüfen Sie nochmals, dass Sie die Parameter korrekt verwenden. Die Beispielanforderungen oder Benutzerszenarien zeigen Ihnen, ob ähnliche Operationen ausgeführt werden, und vermitteln Ihnen eine Vorstellung von möglichen Fehlern.

- Sehen Sie im Forum nach.

MemoryDB verfügt über ein Diskussionsforum, indem Sie nach Lösungen für Probleme suchen können, die andere Teilnehmer bereits hatten. Weitere Informationen zur Anzeige des Forums finden Sie unter

<https://forums.aws.amazon.com/aus>.

Kontingente für Amazon MemoryDB für Redis

Ihr AWS Konto verfügt über Standardkontingente, die früher als Limits bezeichnet wurden, für jeden AWS Dienst. Wenn nicht anders angegeben, gilt jedes Kontingent spezifisch für eine Region. Sie können Erhöhungen für einige Kontingente beantragen und andere Kontingente können nicht erhöht werden.

Informationen zur Erhöhung eines Kontingents finden Sie unter [Anfordern einer Kontingenterhöhung](#) im Benutzerhandbuch zu Service Quotas. Wenn das Kontingent unter Service Quotas noch nicht in verfügbar ist, verwenden Sie das [Formular zur Erhöhung des Service-Limits](#).

Ihr AWS Konto hat die folgenden Kontingente für MemoryDB.

Ressource	Standard
Knoten pro Region	300
Knoten pro Cluster pro Instance-Typ	90
Knoten pro Shard	6
Parametergruppen pro Region	150
Subnetzgruppen pro Region	150
Subnetze pro Subnetzgruppe	20
Benutzer pro Benutzergruppe	100
Gesamtzahl der Benutzer	1000
Anzahl der Benutzergruppen	100

Dokumentverlauf für das MemoryDB-Benutzerhandbuch

In der folgenden Tabelle werden die Dokumentation für MemoryDB beschrieben.

Änderung	Beschreibung	Datum
MemoryDB unterstützt jetzt IAM	Mit der IAM-Authentifizierung können Sie eine Verbindung mit MemoryDB für Redis mithilfe von Identitäten authentifizieren. AWS Identity and Access Management Auf diese Weise können Sie Ihr Sicherheitsmodell stärken und viele administrative Sicherheitssaufgaben vereinfachen. Weitere Informationen finden Sie unter Authentifizierung mit IAM .	10. Mai 2023
MemoryDB unterstützt jetzt Redis 7	Diese Version erweitert MemoryDB für Redis um mehrere neue Funktionen: Redis-Funktionen, ACL-Verbesserungen, Sharded Pub/Sub und verbessertes I/O-Multiplexing. Weitere Informationen finden Sie unter Redis-Engine-Versionen .	09. Mai 2023
MemoryDB bietet jetzt reservierte Knoten	Reservierte Knoten bieten Ihnen einen deutlichen Discount im Vergleich zu den bedarfsorientierten Preisen für Knoten. Bei reservierten Knoten handelt es sich nicht um physische Knoten, sondern	27. Dezember 2022

um einen Fakturierungsrabat
t für die Nutzung gewisser
On-Demand-Nodes in Ihrem
Konto. Weitere Informationen
finden Sie unter [Reservierte
MemoryDB-Nodes](#).

[MemoryDB unterstützt jetzt Data Tiering](#)

Daten-Tiering von MemoryDB
für Redis. Sie können Daten-
Tiering als kostengünstigere
Methode verwenden, um Ihre
Cluster auf bis zu Hunderte
von Terabyte Kapazität zu
skalieren. Weitere Informati
onen finden Sie unter [Daten-
Tiering](#).

03. November 2022

[MemoryDB unterstützt jetzt das native JavaScript JSON- Format \(JSON Notation\).](#)

Das native JavaScript JSON-
Format (JSON Notation) ist
eine einfache schemalos
e Methode zum Codieren
komplexer Datensätze
innerhalb von Redis-Clu
stern. Sie können Daten nativ
mithilfe des JSON-Formats
(JavaScriptObjektNotation)
in Redis-Clustern speichern
, und darauf zugreifen. Sie
können die in diesen Clustern
gespeicherten JSON-Daten
aktualisieren, ohne benutzerd
efinierten Code verwalten zu
müssen, um ihn zu serialisi
eren und zu deserialisieren.
Weitere Informationen finden
Sie unter [Erste Schritte mit
JSON](#).

25. Mai 2022

[MemoryDB unterstützt jetzt AWS PrivateLink](#)

AWS PrivateLink ermöglicht den privaten Zugriff auf MemoryDB-API-Operationen ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder AWS Direct Connect-Verbindung. Weitere Informationen finden Sie unter [MemoryDB-API und Schnittstellen-VPC-Endpunkte](#) (). AWS PrivateLink

24. Januar 2022

[Erstversion](#)

Erstveröffentlichung des MemoryDB-Benutzerhandbuchs. Weitere Informationen finden Sie unter [Was ist MemoryDB?](#)

19. August 2021

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.