aws

開發人員指南

Amazon MemoryDB for Redis

Amazon MemoryDB for Redis: 開發人員指南

Table of Contents

什麼是記憶數據庫的 Redis？ 1
內存數據庫的功能 1
記憶體 DB 核心組件 2
叢集 2
節點 4
碎片 4
參數群組 4
子網路群組 4
存取控制清單 5
使用者 5
相關服務 5
選擇區域與可用區域 5
安置您的節點 7
支援的地區和端點 8
訪問內存數據庫 11
內存數據庫安全 11
開始使用記憶體資料庫 13
設定 13
建立 AWS 帳戶 13
授予程式設計存取權 15
設置您的權限（僅限新的 MemoryDB 用戶） 16
下載和設定 AWS CLI 17
步驟 1：建立叢集 18
建立記憶體資料庫叢集 18
設定驗證 27
步驟 2：授權對叢集的存取 28
步驟 3：Connect 到叢集 30
尋找您的叢集端點 30
Connect 至記憶體資料庫叢集 (Linux) 30
步驟 4：刪除叢集 32
接下來做些什麼？ 34
管理節點 35
內存數據庫節點和碎片 35
支援的節點類型 36

預留節點 38
預留節點概觀 38
替換節點 48
管理叢集 50
資料分層 51
最佳實務 51
限制 52
資料層分定價 52
監控 52
使用資料分層 53
在資料分層啟用的情況下，將資料從快照還原到叢集 54
準備叢集 56
判斷要求 56
建立叢集 59
檢視叢集的詳細資訊 60
修改叢集 64
從叢集新增/移除節點 67
存取叢集 69
授予對您叢集的存取 69
從外部訪問內存數據庫AWS 71
尋找連線端點 77
碎片 80
尋找碎片的名字 80
管理您的記憶體資料庫實作 85
引擎版本 85
Redis 7.0 (增強版) 85
Redis 7.0 (增強版) 86
Redis 6.2 (增強版) 86
升級引擎版本 87
JSON 入門 89
JSON 數據類型概述 90
支援的命令 101
標記您的內存 DB 資源 142
使用標籤監控成本 146
使用 AWS CLI 管理標籤 147
使用 MemoryDB API 管理標籤 150

管理維護作業 153
最佳實務 154
受限制的 Redis 命令 155
恢復能力 156
提供發佈/訂閱閱讀和增實務 158
最佳實務：線上叢集大小調整 158
了解記憶體資料庫複寫 159
一致性 159
叢集中的複寫 159
使用異地同步備份將停機時間降至最低 161
變更複本的數量 168
快照和還原 178
限制 179
成本 179
排程自動快照 180
製作手動快照 181
建立最終快照 184
說明快照 186
複製快照 189
匯出快照 192
從快照還原 201
使用快照植入叢集 206
標記快照 212
刪除快照 213
擴展 214
擴展 MemoryDB 叢集 215
使用參數群組設定引擎參數 235
參數管理 236
參數群組層 237
建立參數群組 237
依名稱列出參數群組 242
列出參數群組的值 247
修改參數群組 248
刪除參數群組 250
雷迪斯特定參數 252
教學課程：設定 Lambda 函數以存取 Amazon VPC 中的記憶體資料庫 266

步驟 1：建立叢集 267
步驟 2：建立 Lambda 函數 270
步驟 3：測試 Lambda 函數 274
步驟 4：清理（可選） 274
向量搜尋 276
向量搜尋概觀 276
索引和密鑰空間 277
索引欄位類型 278
向量索引演算法 278
向量搜尋查詢運算式 279
資訊指令 281
向量搜尋安全 283
向量搜尋功能和限制 283
向量搜尋可用性 283
參數限制 283
縮放限制 284
操作限制 284
快照匯入/匯出和實時移轉 285
記憶體消耗 285
回填期間記憶體不足 285
交易 285
使用案例 285
檢索增強生成 (RAG) 286
基礎型號 (FM) 緩衝記憶體 286
詐騙偵測 287
其他使用案例 287
使用 AWS Management Console 288
使用 AWS Command Line Interface 288
向量搜尋指令 289
FT. 創建 289
英特搜索 293
英尺彙總 296
英尺下降指數 297
英尺資訊 297
英尺 _ 列表 299
英尺別名 299

英尺 300
英尺更新 300
英尺 _ 別名列表 300
FT.CONFIG 獲取 301
FT.CONFIG 說明 301
英特配置集 301
英尺. 輪廓 302
英尺解釋 302
英尺解釋 302
安全性 304
資料保護 304
適用於 Redis 的記憶體資料庫中的資料安全性 305
靜態加密 307
傳輸中加密 (TLS) 309
使用 ACL 驗證使用者 310
以 IAM 進行身分驗證 324
身分與存取管理 330
物件 331
使用身分驗證 332
使用政策管理存取權 334
適用於 Redis 的記憶體資料庫如何與 IAM 搭配使用 336
身分型政策範例 342
故障診斷 344
存取控制 346
管理存取概觀 347
日誌記錄和監控 372
使用監控 CloudWatch 372
監控事件 389
記錄內存數據庫的 Redis API 調用AWS CloudTrail 401
合規驗證 407
基礎設施安全性 408
網際網路流量隱私權 408
記憶數據庫和 Amazon VPC 408
子網路和子網路群組 420
Redis API 和界面 VPC 端點 (AWS PrivateLink) 432
服務更新 435

管理服務更新 435
參考 438
使用內存 DB API 439
使用查詢 API 439
可用程式庫 442
對應用程式進行疑難排解 442
配額 444
文件歷史紀錄 445
..... cdxlvii

什麼是記憶數據庫的 Redis？

適用於 Redis 的 MemoryDB 是一種耐用的記憶體內資料庫服務，可提供超快速的效能。它是專為具有微服務架構的現代應用程式所打造。

MemoryDB 與熱門的開放原始碼資料存放區 Redis 相容，可讓您使用目前使用的相同靈活且易於使用的 Redis 資料結構、API 和命令，快速建置應用程式。使用 MemoryDB，您的所有數據都存儲在內存中，從而使您能夠實現微秒讀取和 10 毫秒寫入延遲和高輸送量。MemoryDB 還使用異地同步備份交易記錄檔在多個可用區域 (AZ) 之間持久儲存資料，以實現快速容錯移轉、資料庫復原和節點重新啟動。

MemoryDB 同時提供記憶體內效能和異地同步備份耐久性，可作為微服務應用程式的高效能主要資料庫使用，無需分別管理快取記憶體和耐用資料庫。

主題

- 內存數據庫的功能
- 記憶體 DB 核心組件
- 相關服務
- 選擇區域與可用區域
- 訪問內存數據庫
- 內存數據庫安全

內存數據庫的功能

適用於 Redis 的 MemoryDB 是一種耐用的記憶體內資料庫服務，可提供超快速的效能。內存數據庫的功能包括：

- 主節點具有強大的一致性，並保證複本節點的最終一致性。如需詳細資訊，請參閱一致性。
- 微秒讀取和 10 毫秒寫入延遲，每個叢集最高可達 1.6 億 TPS。
- 靈活且易於使用的 Redis 資料結構和 API。輕鬆建置新的應用程式或遷移現有 Redis 應用程式，幾乎不需要修改。
- 使用異地同步備份交易記錄的資料耐久性，可提供快速的資料庫復原和
- 異地同步備份可用性，具備自動容錯移轉功能，以及從節點故障偵測和復原
- 通過添加和刪除節點輕鬆水平縮放，或通過移動到更大或更小的節點類型進行垂直縮放。您可以透過新增碎片來擴展寫入輸送量，並透過新增複本來擴展讀取輸送量。

- 主節點的 R ead-after-write 一致性，並保證複本節點的最終一致性。
- MemoryDB 支持傳輸中的加密，靜態加密和用戶通過身份驗證。使用存取控制清單 (ACL) 驗證使用者
- Amazon S3 中的自動快照可保留長達 35 天。
- 每個叢集最多 Support 500 個節點和 100 TB 以上的儲存空間 (每個碎片有 1 個複本)。
- 使用 TLS 加密傳輸中，並使用金鑰靜態加密。AWS KMS
- 使用 Redis 使用存取控制清單 (ACL) 驗證使用者 的使用者驗證和授權。
- Support AWS 重力 on2 執行個體類型。
- 與其他AWS服務 (例如 CloudWatch Amazon VPC 和 Amazon SNS) 整合 CloudTrail，以進行監控、安全性和通知。
- 完全受管理的軟體修補與升級。
- AWS適用於管理 API 的身分與存取管理 (IAM) 整合和標籤式存取控制。

記憶體 DB 核心組件

您可以在下面找到 MemeryDB 部署的主要元件概觀。

主題

- 叢集
- 節點
- 碎片
- 參數群組
- 子網路群組
- 存取控制清單
- 使用者

叢集

叢集是為單個資料集提供服務的一或多個節點的集合。MemeryDB 資料集會被劃分為碎片，每個碎片都具有一個主要節點，以及最多 5 個可選複本節點。主節點為讀取和寫入請求提供服務，而副本只提供讀取請求。主節點可以故障切換到副本節點，從而將該副本提升到該分片的新主節點。MemeryDB

將 Redis 作為其數據庫引擎運行，當您創建集羣時，您可以為集羣指定 Redis 版本。您可以使用AWS CLI、內存數據庫 API 或AWS Management Console。

每個記憶體 DB 叢集都會執行 Redis 引擎版本。每個 Redis 引擎版本都有自己的支援功能。此外，每個 Redis 引擎版本的參數群組都具有參數集，可控制所管理叢集的行為。

叢集的運算和記憶體容量取決於叢集的節點類型。您可以選擇最符合您需求的節點類型。若您的需求隨時間而有所改變，可變更節點類型。如需相關資訊，請參閱 支援的節點類型。

> **Note**
>
> 如需「記憶體 DB」節點類型的定價資訊，請參記憶體 DB 定價。

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 服務，在虛擬私有雲端 (VPC) 上執行叢集。使用 VPC 時，您可以掌控您的虛擬聯網環境。您可以選擇自己的 IP 地址範圍、建立子網路，以及設定路由和存取控制清單。MemoryDB 會管理快照、軟體修補、自動故障偵測與復原作業。在 VPC 中執行叢集無需額外成本。如需將 Amazon VPC 與記憶體 DB 結合使用的詳細資訊，請參記憶數據庫和 Amazon VPC。

許多記憶體 DB 操作都在叢集中做為目標：

- 建立叢集
- 修改叢集
- 拍攝叢集快照
- 刪除叢集
- 在叢集中檢視元素
- 從叢集中新增或移除成本配置標籤

如需詳細資訊，請參閱下列相關主題：

- 管理叢集 與 管理節點

  叢集、節點和相關操作的詳細資訊。
- Redis 內存數據庫中的恢復能力

  提升叢集容錯能力的相關資訊。

節點

一個節點是記憶體 DB 部署的最小建置區塊，並且會使用 Amazon EC2 執行個體。每個節點都會執行您建立叢集時所選擇的 Redis 版本。節點屬於屬於羣集的分片。

每個節點都會在您建立叢集時所選擇的版本執行個體。如果需要，您可以向上或縮減叢集中的節點，以使叢集中的節點向上或縮減為其他類型。如需詳細資訊，請參閱 擴展 。

叢集中每個節點的節點類型都相同。支援多種類型的節點，每個節點都有各種數量的記憶體。如需支援的節點類型清單，請參閱「支援的節點類型」。

如需節點的詳細資訊，請參閱管理節點。

碎片

分片是由 1 到 6 個節點組成的分組，其中一個節點用作主寫入節點，另外 5 個節點用作只讀副本。內存數據庫羣集始終至少有一個分片。

MemoryDB 羣集最多可以有 500 個分片，您的數據跨分片進行分區。例如，您可以選擇設定具有 500 個節點的叢集，並容許碎片在 83 個（每個碎片一個主要版本和 5 個複本）到 500 個（單一主要版本並且沒有複本）之間變化。請確保有足夠的可用 IP 地址來容納增加的數量。常見的缺陷包括子網路群組中的子網路的 CIDR 範圍太小，或是子網路被共用並被其他叢集大量使用。

多個節點碎片實作複寫的方式，是使用一個讀/寫主節點及 1 至 5 個複本節點。如需詳細資訊，請參閱了解記憶體資料庫複寫。

如需碎片的詳細資訊，請參閱使用碎片。

參數群組

參數組是管理叢集中 Redis 運行時間設定的便利方式。參數用於控制記憶體使用情況、項目大小等。MemeryDB 參數組是引擎特定參數的命名集合，您可以將此參數應用於叢集，並且該叢集中的所有節點都以完全相同的方式進行配置。

如需 MeryDB 參數組的詳細資訊，請參使用參數群組設定引擎參數。

子網路群組

子網路群組是子網路的集合 (一般是私有)，您可以為在 Amazon Virtual Private Cloud (VPC) 環境中執行的叢集指定這些子網路。

在 Amazon VPC 中建立叢集時，您可以指定子網路組或使用提供的默認子網路組。MemeryDB 會使用該子網路組來選擇子網路及該子網路中的 IP 地址，以與您的節點建立關聯。

如需記憶體 DB 子網路組的詳細資訊，請參子網路和子網路群組。

存取控制清單

訪問控制列表是一或多個用户的集合。訪問字符串跟隨 RedisACL 規則授權用户訪問 Redis 命令和數據。

有關內存數據庫訪問控制列表的詳細信息，請參閱使用存取控制清單 (ACL) 驗證使用者。

使用者

用户具有用户名和密碼，用於訪問 MemeryDB 羣集上的數據和發出命令。用户是訪問控制列表 (ACL) 的成員，您可以使用該列表確定該用户在 MemeryDB 羣集上的權限。如需詳細資訊，請參閱「」使用存取控制清單 (ACL) 驗證使用者

相關服務

ElastiCache 對於雷迪斯

在決定是否為 Redis 或 ElastiCache Redis 使用內存數據庫時，請考慮以下比較：

- 適用於 Redis 的 MemoryDB 是耐用的記憶體內資料庫，適用於需要超快速主要資料庫的工作負載。若您的工作負載需要可提供超快效能 (微秒讀取和 1 毫秒寫入延遲) 的耐用資料庫，您應該考慮使用 MemoryDB。若您想使用具有主要耐久資料庫的 Redis 資料結構和 API 來建置應用程式，MemoryDB 也非常適合您。最後，您應考慮使用 MemoryDB 來簡化您的應用程式架構，並以快取取代使用資料庫，以提高耐久性和效能，進而降低成本。
- ElastiCache Redis 是一種通常用於緩存使用 Redis 的其他數據庫和數據存儲中的數據的服務。您應該考慮使 ElastiCache 用 Redis 來快取工作負載，以便透過現有的主要資料庫或資料存放區加速資料存取 (微秒讀取和寫入效能)。如果您想要使 ElastiCache 用 Redis 資料結構和 API 來存取主要資料庫或資料存放區中儲存的資料，您也應該考慮 Redis 使用案例。

選擇區域與可用區域

AWS 雲端運算資源放在高可用性資料中心設施中。為了提供額外的可擴展性和可靠性，這些資料中心設施會位在不同的實體位置。這些地點是依「區域」及「可用區域」分類。

AWS 區域範圍都很大，且廣泛分散於個別地理位置中。可用區域是 AWS 區域內的不同位置，設計成可與其他可用區域中的故障隔離。它們提供同一 AWS 區域中其他可用區域的價廉、低延遲網路連線能力。

> **Important**
>
> 每個區域都是完全獨立的。您啟動的任何 MemoryDB 活動（例如，建立叢集）只會在目前的預設區域中執行。

若要在特定區域中建立或使用叢集，請使用對應的區域服務端點。如需了解服務端點，請參閱支援的地區和端點。

安置您的節點

至少有一個複本的任何叢集都必須分散在 AZ 中。尋找單一 AZ 中所有項目的唯一方法是使用由單一節點碎片組成的叢集。

透過將節點定位在不同的 AZ 中，MemoryDB 可消除在一個 AZ 中發生故障 (例如停電) 的機會造成可用性喪失。

- 建立記憶體資料庫叢集
- 修改記憶體資料庫叢集

支援的地區和端點

Redis 的內存數據庫是在多個區域可用。AWS這表示您可以在符合需求的位置啟動 MemoryDB 叢集。例如，您可以在最接近您客戶的 AWS 區域中啟動，或是在特定 AWS 區域啟動，以符合特定法規要求。此外，隨著 MemoryDB 將可用性擴展到新的AWS區域，MemoryDB 支持當時的兩個最新MAJOR.MINOR版本的新區域。如需 MemoryDB 版本的詳細資訊，請參閱〈〉。雷迪斯引擎版本

根據預設，AWS開發套件AWS CLI、MemoryDB API 和 MemoryDB 主控台會參考美國東部 (維吉尼亞北部) 區域。隨著 MemoryDB 將可用性擴展到新區域，這些區域的新端點也可用於 HTTP 請求AWS CLI、AWS SDK 和主控台。

每個 區域皆設計為與其他 區域完全隔離。各個區域包含多個可用區域 (AZ)。透過在不同的 AZ 中啟動節點，您可以實現最大的容錯能力。如需區域和可用區域的詳細資訊，請參閱選擇區域與可用區域本主題開頭的。

支援記憶體資料庫的區域

區域名稱/區域	端點	通訊協定	
美國東部 (俄亥俄) 區域 us-east-2	memory-db.us-east-2.amazonaws.com	HTTPS	
美國東部 (維吉尼亞北部) 區域 us-east-1	memory-db.us-east-1.amazonaws.com	HTTPS	
美國西部 (加利佛尼亞北部) 區域 us-west-1	memory-db.us-west-1.amazonaws.com	HTTPS	
美國西部 (奧勒岡) 區域 us-west-2	memory-db.us-west-2.amazonaws.com	HTTPS	

區域名稱/區域	端點	通訊協定	
加拿大 (中部) 區域 ca-central-1	memory-db.ca-central-1.amazonaws.com	HTTPS	
亞太區域 (香港) 區域 ap-east-1	memory-db.ap-eastl-1.amazonaws.com	HTTPS	
亞太 (孟買) 區域 ap-south-1	memory-db.ap-south-1.amazonaws.com	HTTPS	
亞太 (東京) 區域 ap-northeast-1	memory-db.ap-northeast-1.amazonaws.com	HTTPS	
亞太 (首爾) 區域 ap-northeast-2	memory-db.ap-northeast-2.amazonaws.com	HTTPS	
亞太 (新加坡) 區域 ap-southeast-1	memory-db.ap-southeast-1.amazonaws.com	HTTPS	
亞太 (雪梨) 區域 ap-southeast-2	memory-db.ap-southeast-2.amazonaws.com	HTTPS	
歐洲 (法蘭克福) 區域 eu-central-1	memory-db.eu-central-1.amazonaws.com	HTTPS	
歐洲 (愛爾蘭) 區域 eu-west-1	memory-db.eu-west-1.amazonaws.com	HTTPS	

區域名稱/區域	端點	通訊協定	
歐洲 (倫敦) 區域 eu-west-2	memory-db.eu-west-2.amazonaws.com	HTTPS	
歐洲 (巴黎) 區域 eu-west-3	memory-db.eu-west-3.amazonaws.com	HTTPS	
歐洲 (斯德哥爾摩) 區域 eu-north-1	memory-db.eu-north-1.amazonaws.com	HTTPS	
Europe (Milan) Region eu-south-1	memory-db.eu-south-1.amazonaws.com	HTTPS	
南美洲 (聖保羅) 區域 sa-east-1	memory-db.sa-east-1.amazonaws.com	HTTPS	
中國 (北京) 區域 cn-north-1	memory-db.cn-north-1.amazonaws.com.cn	HTTPS	
中國 (寧夏) 區域 cn-northwest-1	memory-db.cn-northwest-1.amazonaws.com.cn	HTTPS	

如需按地區分AWS類的產品與服務表格，請參閱按地區分類的產品和服務。

如需區域內支援的可用區域表格，請參閱子網路和子網路群組。

訪問內存數據庫

每個 MemoryDB 叢集端點都包含一個位址和一個連接埠。此叢集端點支援 Redis 叢集通訊協定，可讓用戶端探索叢集中每個節點的特定角色、IP 位址和插槽。當主節點發生故障且複本已在其位置升級時，您可以連接到叢集端點以使用 Redis 叢集通訊協定探索新的主節點。

您需要連接到叢集端點，以使用cluster nodes或cluster slots命令探索節點端點。在發現金鑰的正確節點之後，您可以直接連線到節點以進行讀取/寫入要求。Redis 用戶端可以使用叢集端點自動連線到正確的節點。

若要疑難排解叢集中的特定節點，您也可以使用節點特定的端點，但對於正常使用而言，這些端點並不是必需的。

若要尋找叢集的端點，請參閱下列內容：

- 尋找記憶體 DB 叢集的端點 (AWSCLI)
- 尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

若要連線至節點或叢集，請參閱使用雷迪斯 CLI 連接到內存數據庫節點。

內存數據庫安全

MemoryDB 的安全性分為三個層級進行管理：

- 若要控制誰可以在 MemoryDB 叢集和節點上執行管理動作，請使用 AWS Identity and Access Management (IAM)。當您連線到AWS使用 IAM 登入資料時，您的AWS帳戶必須具有 IAM 政策，以授予執行操作所需的許可。如需更多資訊，請參閱 適用於 Redis 的記憶體資料庫中的身分識別和存取管理
- 若要控制叢集的存取層級，您可以建立具有指定權限的使用者，並將其指派給存取控制清單 (ACL)。接著，ACL 會與一或多個叢集相關聯。如需詳細資訊，請參閱使用存取控制清單 (ACL) 驗證使用者。
- 必須在以 Amazon VPC 服務為基礎的虛擬私有雲端 (VPC) 中建立 MemoryDB 叢集。若要控制哪些裝置和 Amazon EC2 執行個體可以為 VPC 中的 MemoryDB 叢集開啟連線到節點的端點和連接埠，請使用 VPC 安全群組。您可以使用 Transport Layer Security (TLS)/Secure Sockets Layer (SSL) 進行這些端點和連接埠連線。此外，您公司的防火牆規則可以控制在您公司執行的裝置是否可以開啟與 MemoryDB 叢集的連線。如需 VPC 的詳細資訊，請參閱記憶數據庫和 Amazon VPC。

如需設定安全性的相關資訊，請參閱MemoryDB for Redis is is is is is is is is is。

開始使用記憶體資料庫

此練習會引導您完成使用 MemoryDB 管理主控台建立、授與存取權、連線，以及最終刪除 MemoryDB 叢集的步驟。

> **Note**
>
> 對於本練習的目的，我們建議您在建立叢集時使用「簡易建立」選項，並在進一步探索 MemoryDB 的功能之後返回其他兩個選項。

主題

- 設定
- 步驟 1：建立叢集
- 步驟 2：授權對叢集的存取
- 步驟 3：Connect 到叢集
- 步驟 4：刪除叢集
- 接下來做些什麼？

設定

接下來，您可以找到描述開始使用 MemoryDB 必須採取的一次性動作的主題。

主題

- 建立 AWS 帳戶
- 授予程式設計存取權
- 設置您的權限（僅限新的 MemoryDB 用戶）
- 下載和設定 AWS CLI

建立 AWS 帳戶

註冊 AWS 帳戶

如果您還沒有 AWS 帳戶，請完成以下步驟建立新帳戶。

註冊 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電，並在電話鍵盤輸入驗證碼。

   註冊 AWS 帳戶時，會建立 AWS 帳戶根使用者。根使用者有權存取該帳戶中的所有 AWS 服務和資源。作為最佳安全實務，將管理存取權指派給管理使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

註冊程序完成後，AWS 會傳送一封確認電子郵件給您。您可以隨時登錄 https://aws.amazon.com/ 並選擇 我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立管理使用者

註冊後，請保護您的AWS 帳戶AWS 帳戶根使用者AWS IAM Identity Center、啟用和建立系統管理使用者，這樣您就不會將 root 使用者用於日常工作。

保護您的 AWS 帳戶根使用者

1. 選擇 根使用者 並輸入您的 AWS 帳戶電子郵件地址，以帳戶擁有者身分登入 AWS Management Console。在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入使用者指南中的以根使用者身分登入。
2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需指示，請參閱《IAM 使用者指南》中的為 AWS 帳戶根使用者啟用虛擬 MFA 裝置 (主控台)。

建立管理使用者

1. 啟用 IAM 身分識別中心。

   如需指示，請參閱《AWS IAM Identity Center使用指南》AWS IAM Identity Center中的「啟用」。
2. 在 IAM 身分中心中，將管理存取權授與管理使用者。

   若要取得有關使用IAM Identity Center 目錄做為身分識別來源的自學課程，請參閱《使用指南》IAM Identity Center 目錄中的「以預設值設定使用AWS IAM Identity Center者存取」。

以管理員的身分登入

- 若要使用您的 IAM 身分中心使用者登入，請使用建立 IAM 身分中心使用者時傳送至您電子郵件地址的登入 URL。

  如需有關如何使用 IAM Identity Center 使用者登入的說明，請參閱《AWS 登入 使用者指南》中的登入 AWS存取入口網站。

授予程式設計存取權

若使用者想要與 AWS Management Console 之外的 AWS 互動，則需要程式設計存取權。授予程式設計存取權的方式取決於存取 AWS 的使用者類型。

若要授予使用者程式設計存取權，請選擇下列其中一個選項。

哪個使用者需要程式設計存取權？	到	By
人力身分 (IAM Identity Center 中管理的使用者)	使用臨時憑證簽署對 AWS CLI、AWS SDKs 或 AWS APIs 的程式設計請求。	請依照您要使用的介面所提供的指示操作。 • 關於 AWS CLI，請參閱 AWS Command Line Interface使用者指南 中的 設定 AWS CLI 來使用 AWS IAM Identity Center。 • 關於 AWS SDKs、工具和 AWS APIs，請參閱 AWSSDKs 和工具參考指南 中的 IAM Identity Center 驗證。
IAM	使用臨時憑證簽署對 AWS CLI、AWS SDKs 或 AWS APIs 的程式設計請求。	請遵循 IAM 使用者指南 中 使用臨時憑證搭配 AWS 資源 中的指示。
IAM	(不建議使用)	請依照您要使用的介面所提供的指示操作。

哪個使用者需要程式設計存取權？	到	By
	使用長期憑證簽署 AWS CLI、AWS SDKs 或 AWS APIs 的程式設計要求。	• 關於 AWS CLI，請參閱 AWS Command Line Interface使用者指南 中的 使用 IAM 使用者憑證進行驗證。 • 關於 AWS SDKs 和工具，請參閱 AWSSDKs 和工具參考指南 中的 使用長期憑證進行驗證。 • 關於 AWS API，請參閱 IAM 使用者指南 中的 管理 IAM 使用者的存取金鑰。

相關主題:

- IAM 使用者指南中的什麼是 IAM？。
- AWS 一般參考中的 AWS 安全憑證。

設置您的權限（僅限新的 MemoryDB 用戶）

若要提供存取權，請新增權限至您的使用者、群組或角色：

- AWS IAM Identity Center 中的使用者和群組：

  建立權限合集。請遵循 AWS IAM Identity Center 使用者指南 的 建立權限合集 中的指示。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。請遵循 IAM 使用者指南 的 為第三方身分提供者 (聯合) 建立角色 中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請遵循 IAM 使用者指南 的 為 IAM 使用者建立角色 中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南 的 新增權限至使用者 (主控台) 中的指示。

適用於 Redis 的 MemoryDB 會建立並使用服務連結角色來佈建資源，並代表您存取其他 AWS資源和服務。若要讓 MemoryDB 為您建立服務連結角色，請使用名為的管理原則 AWS。`AmazonMemoryDBFullAccess`此角色隨附了預先佈建、服務代表您建立服務連結角色所需的許可。

您可能決定不使用預設的政策，而是改為使用自訂的受管政策。在此情況下，請確定您具有呼叫權限，`iam:createServiceLinkedRole`或已建立 MemoryDB 服務連結角色。

如需詳細資訊，請參閱下列內容：

- 建立新政策 (IAM)
- AWS適用於 Redis 的記憶體資料庫的受管 (預先定義) 政策
- 針對 Redis 的 Amazon 記憶體資料庫使用服務連結角色

下載和設定 AWS CLI

AWS CLI 可於 http://aws.amazon.com/cli 取得。它可在 Windows、MacOS 和 Linux 上執行。在您下載 AWS CLI 之後，請遵循這些步驟安裝及設定它：

1. 請前往 AWS 命令列界面使用者指南。
2. 依照安裝 AWS CLI 及設定 AWS CLI 的指示操作。

步驟 1：建立叢集

在建立供生產使用的叢集之前，您明顯需要考慮如何設定叢集以符合您的業務需求。這些問題在「準備叢集」一節中說明。基於此入門練習的目的，您可以接受套用預設規劃值的位置。

您建立的叢集將會實際上線，而非在沙盒中執行。您必須支付執行個體的標準 MemoryDB 使用費，直到您刪除執行個體為止。如果您一口氣地完成這裡所述的練習，並在完成時刪除您的叢集，則總計費用會很少 (通常不到 1 美元)。如需有關記憶體資料庫使用率的詳細資訊，請參閱 MemoryDB。

您的叢集會在以 Amazon VPC 服務為基礎的 Virtual Private Cloud (VPC) 中啟動。

建立記憶體資料庫叢集

下列範例說明如何使用 AWS CLI 和記憶體資料庫 API 建立叢集。 AWS Management Console

建立叢集 (主控台)

若要使用 MemoryDB 主控台建立叢集

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 選擇 [叢集] 在左側導覽窗格中，然後選擇 [建立]。

Easy create

1. 填妥 Configuration (組態) 部分。這會設定叢集的節點類型和預設組態。從下列選項中選取您需要的適當記憶體大小和網路效能：
   - 生產
   - 開發/測試
   - 示範
2. 完成「叢集資訊」區段。
   a. 在 Name (名稱) 為叢集輸入名稱。

      叢集命名限制條件如下：
      - 必須包含 1-40 個英數字元或連字號。
      - 必須以字母開頭。

- 不能連續包含兩個連字號。
- 結尾不能是連字號。

b. 在 Description (說明) 方塊中，輸入此叢集的說明。

3. 完成「子網路群組」區段：

- 對於子網路群組，請建立新的子網路群組，或從可用清單中選擇要套用至此叢集的現有子網路群組。如果您要創建一個新的：
  - 輸入名稱
  - 輸入說明
  - 如果您啟用多個可用區，子網路群組必須至少包含兩個位於不同可用區域的子網路。如需詳細資訊，請參閱 子網路和子網路群組。
  - 如果您要建立新的子網路群組，但沒有現有的 VPC，系統會要求您建立 VPC。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。

4. 對於 Vector 搜尋，您可以啟用 Vector 搜尋功能來儲存向量嵌入和執行向量搜尋。請注意，這將修復 Redis 版本兼容性，參數組和碎片的值。如需詳細資訊，請參閱 向量搜尋。
5. 檢視預設設定：

   使用「輕鬆建立」時，預設會設定剩餘的叢集設定。請注意，其中一些設定可在建立之後變更，如建立後可編輯的所示。
6. 對於標籤，您可以選擇性地套用標籤來搜尋和篩選叢集或追蹤 AWS 成本。
7. 檢閱所有項目和選項，然後進行任何所需的更正。準備就緒後，選擇 [建立] 以啟動叢集，或選擇 [取消] 取消作業。

一旦叢集的狀態變為可用，您就可以為其授予 EC2 存取權限、連線至叢集並開始使用叢集。如需更多資訊，請參閱步驟 2：授權對叢集的存取

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未主動使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱步驟 4：刪除叢集。

Create new cluster

1. 完成「叢集資訊」區段。

a. 在 Name (名稱) 為叢集輸入名稱。

叢集命名限制條件如下：

- 必須包含 1-40 個英數字元或連字號。
- 必須以字母開頭。
- 不能連續包含兩個連字號。
- 結尾不能是連字號。

b. 在 Description (說明) 方塊中，輸入此叢集的說明。

2. 完成「子網路群組」區段：

- 對於子網路群組，請建立新的子網路群組，或從可用清單中選擇要套用至此叢集的現有子網路群組。如果您要創建一個新的：
  - 輸入名稱
  - 輸入說明
  - 如果您啟用多個可用區，子網路群組必須至少包含兩個位於不同可用區域的子網路。如需詳細資訊，請參閱 子網路和子網路群組。
  - 如果您要建立新的子網路群組，但沒有現有的 VPC，系統會要求您建立 VPC。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。

3. 完成「叢集設定」區段：

a. 若要啟用 Vector 搜尋功能，您可以啟用此功能來儲存向量嵌入和執行向量搜尋。請注意，這將修復 Redis 版本兼容性，參數組和碎片的值。如需詳細資訊，請參閱 向量搜尋。

b. 如需 Redis 版本相容性，請接受預設值`6.2`。

c. 對於連接埠，請接受 6379 的預設 Redis 連接埠，或者，如果您有理由使用不同的連接埠，請輸入連接埠號碼。

d. 對於「參數」群組，如果您已啟用向量搜尋，請使用`default.memorydb-redis7.search.preview`。否則，請接受`default.memorydb-redis7`參數群組。

參數群組可控制叢集的執行時間參數。如需參數群組的詳細資訊，請參閱雷迪斯特定參數。

e. 針對節點類型，請選擇您想要的節點類型 (及其相關記憶體大小) 的值。

如果您從 r6gd 系列中選擇節點類型，則將自動啟用資料分層，此將在記憶體和 SSD 之間分割資料儲存。如需詳細資訊，請參閱 資料分層。

f. 對於碎片數量，請選擇您要用於此叢集的碎片數目。為了提高叢集的可用性，我們建議您至少新增 2 個碎片。

您可以動態變更叢集中的碎片數目。如需詳細資訊，請參閱 擴展 MemoryDB 叢集。

g. 針對 Replicas per shard (每個碎片的複本)，選擇您要讓每個碎片具備的僅供讀取複本節點數目。

存在以下限制：

- 如果您已啟用多個可用區，請確保每個碎片至少有一個複本。
- 使用主控台建立叢集時，每個碎片的複本數都相同。

h. 選擇下一步

i. 完成「進階設定」區段：

i. 在 Security groups (安全群組) 中，選擇要用於此叢集的安全群組。安全群組可做為防火牆來控制叢集的網路存取。您可以為 VPC 使用預設安全群組，或建立新的安全群組。

如需安全群組的詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 的安全群組。

ii. 若要加密資料，您有下列選項：

- Encryption at rest (靜態加密) - 啟用存放在磁碟上的資料加密功能。如需詳細資訊，請參閱靜態加密。

> **Note**
>
> 透過選擇客戶管理 AWS擁有的 KMS 金鑰並選擇金鑰，您可以選擇提供預設以外的加密金鑰。

- Encryption in-transit (傳輸中加密) - 啟用傳輸中資料加密功能。如果您選擇沒有加密，則將以默認用戶創建一個名為「開放訪問」的開放訪問控制列表。如需詳細資訊，請參閱 使用存取控制清單 (ACL) 驗證使用者。

iii. 針對快照，選擇性地指定快照保留期間和快照視窗。依預設，會預先選取 「啟用自動快照」。

iv. 針對 「維護」視窗，選擇性地指定維護時段 維護時段是 MemoryDB 每週為叢集排程系統維護的時間 (通常為 1 小時)。您可以允許 MemoryDB 選擇維護時段的日期和時間（無偏好），也可以自己選擇日期，時間和持續時間（指定維護時段）。如果您從清單中選擇 Specify maintenance window (指定維護時段)，請為您的維護時段選擇 Start day (開始日)、Start time (開始時間) 和 Duration (持續時間)。所有時間均以 UCT 時間表示。

如需詳細資訊，請參閱 管理維護作業。

v. 針對 Notifications (通知)，選擇現有的 Amazon Simple Notification Service (Amazon SNS) 主題，或選擇手動輸入 ARN，並輸入主題的 Amazon 資源名稱 (ARN)。Amazon SNS 可讓您推播通知到已與網際網路連線的智慧裝置。預設是停用通知。如需詳細資訊，請參閱 https://aws.amazon.com/sns/。

vi. 對於標籤，您可以選擇性地套用標籤來搜尋和篩選叢集或追蹤 AWS 成本。

j. 檢閱所有項目和選項，然後進行任何所需的更正。準備就緒後，選擇 [建立] 以啟動叢集，或選擇 [取消] 取消作業。

一旦叢集的狀態變為可用，您就可以為其授予 EC2 存取權限、連線至叢集並開始使用叢集。如需更多資訊，請參閱步驟 2：授權對叢集的存取

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未主動使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱步驟 4：刪除叢集。

Restore from snapshots

在 [快照來源] 下，選擇要從中移轉資料的來源快照。如需詳細資訊，請參閱 快照和還原 。

> **Note**
>
> 如果您希望新叢集啟用向量搜尋，來源快照也必須啟用向量搜尋。

目標叢集預設為來源叢集的設定。或者，您可以在目標叢集上變更下列設定：

1. 叢集資訊

   a. 在 Name (名稱) 為叢集輸入名稱。

      叢集命名限制條件如下：

      - 必須包含 1-40 個英數字元或連字號。
      - 必須以字母開頭。
      - 不能連續包含兩個連字號。
      - 結尾不能是連字號。

   b. 在 Description (說明) 方塊中，輸入此叢集的說明。

2. 子網路群組

   - 對於子網路群組，請建立新的子網路群組，或從可用清單中選擇要套用至此叢集的現有子網路群組。如果您要創建一個新的：

     - 輸入名稱
     - 輸入說明
     - 如果您啟用多個可用區，子網路群組必須至少包含兩個位於不同可用區域的子網路。如需詳細資訊，請參閱 子網路和子網路群組。
     - 如果您要建立新的子網路群組，但沒有現有的 VPC，系統會要求您建立 VPC。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。

3. 叢集設定

   a. 若要啟用 Vector 搜尋功能，您可以啟用此功能來儲存向量嵌入和執行向量搜尋。請注意，這將修復 Redis 版本兼容性，參數組和碎片的值。如需詳細資訊，請參閱 向量搜尋。

   b. 如需 Redis 版本相容性，請接受預設值`6.2`。

   c. 對於連接埠，請接受 6379 的預設 Redis 連接埠，或者，如果您有理由使用不同的連接埠，請輸入連接埠號碼。

   d. 對於「參數」群組，如果您已啟用向量搜尋，請使用`default.memorydb-redis7.search.preview`。否則，請接受`default.memorydb-redis7`參數群組。

      參數群組可控制叢集的執行時間參數。如需參數群組的詳細資訊，請參閱雷迪斯特定參數。

   e. 針對節點類型，請選擇您想要的節點類型 (及其相關記憶體大小) 的值。

如果您從 r6gd 系列中選擇節點類型，則將自動啟用資料分層，此將在記憶體和 SSD 之間分割資料儲存。如需詳細資訊，請參閱 資料分層。

f. 對於碎片數量，請選擇您要用於此叢集的碎片數目。為了提高叢集的可用性，我們建議您至少新增 2 個碎片。

您可以動態變更叢集中的碎片數目。如需詳細資訊，請參閱 擴展 MemoryDB 叢集。

g. 針對 Replicas per shard (每個碎片的複本)，選擇您要讓每個碎片具備的僅供讀取複本節點數目。

存在以下限制：

- 如果您已啟用多個可用區，請確保每個碎片至少有一個複本。
- 使用主控台建立叢集時，每個碎片的複本數都相同。

h. 選擇下一步

i. 進階設定

i. 在 Security groups (安全群組) 中，選擇要用於此叢集的安全群組。安全群組可做為防火牆來控制叢集的網路存取。您可以為 VPC 使用預設安全群組，或建立新的安全群組。

如需安全群組的詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 的安全群組。

ii. 若要加密資料，您有下列選項：

- Encryption at rest (靜態加密) - 啟用存放在磁碟上的資料加密功能。如需詳細資訊，請參閱靜態加密。

> **Note**
>
> 透過選擇客戶管理 AWS擁有的 KMS 金鑰並選擇金鑰，您可以選擇提供預設以外的加密金鑰。

- Encryption in-transit (傳輸中加密) - 啟用傳輸中資料加密功能。如果您選擇沒有加密，則將以默認用戶創建一個名為「開放訪問」的開放訪問控制列表。如需詳細資訊，請參閱 使用存取控制清單 (ACL) 驗證使用者。

iii. 針對快照，選擇性地指定快照保留期間和快照視窗。依預設，會預先選取 「啟用自動快照」。

iv. 針對 「維護」視窗，選擇性地指定維護時段 維護時段是 MemoryDB 每週為叢集排程系統維護的時間 (通常為 1 小時)。您可以允許 MemoryDB 選擇維護時段的日期和時間（無偏好），也可以自己選擇日期，時間和持續時間（指定維護時段）。如果您從清單中選擇 Specify maintenance window (指定維護時段)，請為您的維護時段選擇 Start day (開始日)、Start time (開始時間) 和 Duration (持續時間)。所有時間均以 UCT 時間表示。

如需詳細資訊，請參閱 管理維護作業。

v. 針對 Notifications (通知)，選擇現有的 Amazon Simple Notification Service (Amazon SNS) 主題，或選擇手動輸入 ARN，並輸入主題的 Amazon 資源名稱 (ARN)。Amazon SNS 可讓您推播通知到已與網際網路連線的智慧裝置。預設是停用通知。如需詳細資訊，請參閱 https://aws.amazon.com/sns/。

vi. 對於標籤，您可以選擇性地套用標籤來搜尋和篩選叢集或追蹤 AWS 成本。

j. 檢閱所有項目和選項，然後進行任何所需的更正。準備就緒後，選擇 [建立] 以啟動叢集，或選擇 [取消] 取消作業。

一旦叢集的狀態變為可用，您就可以為其授予 EC2 存取權限、連線至叢集並開始使用叢集。如需更多資訊，請參閱步驟 2：授權對叢集的存取

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未主動使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱步驟 4：刪除叢集。

建立叢集 (AWS CLI)

若要使用建立叢集 AWS CLI，請參閱create-cluster。以下是範例：

針對 Linux、macOS 或 Unix：

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

針對 Windows：

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

您應該得到以下 JSON 響應：

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

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

一旦叢集的狀態變更為，您就可以開始使用叢集`available`。

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未主動使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱步驟 4：刪除叢集。

建立叢集 (記憶體資料庫 API)

若要使用記憶體資料庫 API 建立叢集，請使用下列CreateCluster動作。

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱步驟 4：刪除叢集。

設定驗證

如需有關為叢集設定驗證的資訊，請參閱以 IAM 進行身分驗證和使用存取控制清單 (ACL) 驗證使用者。

步驟 2：授權對叢集的存取

本節假設您已熟悉如何啟動及連線至 Amazon EC2 執行個體。如需詳細資訊，請參閱 Amazon EC2 入門指南。

記憶體資料庫叢集的設計可從 Amazon EC2 執行個體存取。它們也可以由在 Amazon 彈性容器服務中執行的容器化或無伺服器應用程式存取。 AWS Lambda最常見的情況是從同一個 Amazon Virtual Private Cloud（Amazon VPC）中的 Amazon EC2 實例訪問 MemoryDB 集群，這將是本練習的情況。

因此，在您從 EC2 執行個體連接至叢集之前，必須先授權讓 EC2 執行個體存取叢集。

最常見的使用案例是部署於 EC2 執行個體的應用程式時，需要連線至相同 VPC 中的叢集。若要管理相同 VPC 中 EC2 執行個體與叢集之間的存取權限，最簡單的方式如下：

1. 為您的叢集建立 VPC 安全群組。此安全性群組可用來限制叢集的存取。舉例來說，您可以為此安全群建置立自訂規則，允許使用您在建立自訂規則時指派給叢集的連接埠存取 TCP，並可建立您將用於存取叢集的 IP 位址。

   記憶體資料庫叢集的預設連接埠為。6379
2. 為您的 EC2 執行個體建立 VPC 安全群組 (Web 和應用程式伺服器)。若有需要，此安全群組可允許透過 VPC 路由表存取網際網路上的 EC2 執行個體。舉例來說，您可以在此安全群組上設定規則，允許 TCP 透過連接埠 22 存取 EC2 執行個體。
3. 在叢集的安全群組中建立自訂規則，以允許從您為 EC2 執行個體建立的安全群組進行連線。這樣做會允許安全群組的所有成員存取叢集。

在允許來自其他安全群組連線的 VPC 安全群組中建立規則

1. 登入 AWS 管理主控台，然後開啟 Amazon VPC 主控台，網址為 https://console.aws.amazon.com/vpc。
2. 在左導覽窗格中，選擇 Security Groups (安全群組)。
3. 選取或建立要用於叢集的安全性群組。在 Inbound Rules (傳入規則) 下方，選取 Edit Inbound Rules (編輯傳入規則)，然後選取 Add Rule (新增規則)。此安全群組將允許其他安全群組成員存取。
4. 從 Type (類型) 選擇 Custom TCP Rule (自訂 TCP 規則)。

   a. 針對 Port Range (連接埠範圍)，指定您在建立叢集時所使用的連接埠。

      記憶體資料庫叢集的預設連接埠為。6379

b. 在 Source (來源) 方塊中輸入安全群組的 ID。從清單中選取您將用於 Amazon EC2 執行個體的安全群組。

5. 完成後，請選擇 Save (儲存)。

啟用存取之後，您現在就可以連線至叢集，如下一節所述。

如需從不同 Amazon VPC、不同 AWS 區域甚至企業網路存取 MemoryDB 叢集的相關資訊，請參閱以下內容：

- 用於存取 Amazon VPC 中存取記憶資料庫叢集的存取模式
- 從外部訪問內存 DB 資源AWS

步驟 3：Connect 到叢集

在繼續之前，請先完成步驟 2：授權對叢集的存取。

本節假設您已建立 Amazon EC2 執行個體且可連線至該執行個體。如需操作方式說明，請參閱 Amazon EC2 入門指南。

Amazon EC2 執行個體只有在您已授權的情況下，才能連線到叢集。

尋找您的叢集端點

當您的叢集處於可用狀態，且您獲得存取授權後，您就可以登入 Amazon EC2 執行個體並連線至叢集。若要執行此作業，您必須先判斷端點。

若要進一步探索如何尋找端點，請參閱下列內容：

- 尋找記憶體 DB 叢集的端點 (AWS Management Console)
- 尋找記憶體 DB 叢集的端點 (AWSCLI)
- 尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

Connect 至記憶體資料庫叢集 (Linux)

現在您擁有了所需的端點，您可以登入 EC2 執行個體並連線到叢集。在下列範例中，您可以使用 cli 公用程式連線到使用 Ubuntu 22 的叢集。最新版本的 cli 也支援 SSL/TLS，以連接啟用加密/驗證的叢集。

使用雷迪斯 CLI 連接到內存數據庫節點

若要存取 MemoryDB 節點中的資料，您可以使用安全通訊端層 (SSL) 搭配使用的用戶端。您也可以在 Amazon Linux 和 Amazon Linux 2 上使用 redis-cli 搭配 TLS/SSL。

在 Amazon Linux 2 或 Amazon Linux 上使用 Redis-cli 連接到記憶體資料庫叢集

1. 下載並編譯 redis-cli 公用程式。此公用程式隨附於 Redis 軟體發行版本中。
2. 在 EC2 執行個體的命令提示字元中，為您使用的 Linux 版本輸入適當的命令。

   Amazon Linux 2023

如果使用 Amazon 2023，請輸入以下內容：

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

然後鍵入以下命令，以此示例中顯示的內容替換集群的端點和端口。

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

如需尋找端點的詳細資訊，請參閱尋找您的節點端點。

Amazon Linux 2

如果使用 Amazon Linux 2，請輸入以下內容：

```
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

如果使用 Amazon Linux，請輸入以下內容：

```
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/
```

在 Amazon Linux 上，您可能還需要執行下列額外步驟：

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

3. 下載並安裝 redis-cli 公用程式之後，建議您執行選擇性命令。`make-test`

4. 若要連線至已啟用加密和驗證的叢集，請輸入以下指令：

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

> **Note**
>
> 如果您在 Amazon 2023 上安裝了 redis6，您現在可以使用命令`redis6-cli`來代替：`redis-cli`
>
> ```
> redis6-cli -h Primary or Configuration Endpoint --tls -p 6379
> ```

步驟 4：刪除叢集

一旦叢集處於「可用」狀態，就會開始向您收費，不論您是否主動使用亦同。若要停止產生費用，請刪除叢集。

> **Warning**
>
> 當您刪除 MemoryDB 叢集時，會保留您的手動快照。您也可以在刪除叢集之前建立最終快照。不會保留自動快照。如需詳細資訊，請參閱 快照和還原 。

使用 AWS Management Console

以下程序會從您的部署中刪除單一叢集。若要刪除多個叢集，請針對每個要刪除的叢集重複此程序。您不需要等待某個叢集完成刪除，即可開始刪除其他叢集。

刪除叢集

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要選擇要刪除的叢集，請從叢集清單中選擇叢集名稱旁的圓鈕。此案例中為您在「步驟 1：建立叢集」建立的叢集之名稱。
3. 對於 Actions (動作)，請選擇 Delete (刪除)。
4. 首先選擇是否要在刪除叢集之前先建立快照，然後在確認方塊`delete`中輸入「刪除」(Delete) 以刪除叢集，或選擇「取消」以保留叢集。

如果您選擇 Delete (刪除)，叢集的狀態就會變更為 deleting (正在刪除)。

一旦您的叢集不再列於叢集清單，您就不會再因此產生費用。

使用 AWS CLI

下列程式碼會刪除 my-cluster 叢集。此案例中請將 my-cluster 取代為您在「步驟 1：建立叢集」建立的叢集之名稱。

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

delete-clusterCLI 作業只會刪除一個叢集。若要刪除多個叢集，delete-cluster請呼叫您要刪除的每個叢集。您不需要等待某個叢集完成刪除，然後再刪除另一個叢集。

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

如需詳細資訊，請參閱 delete-cluster。

使用記憶體資料庫 API

下列程式碼會刪除 my-cluster 叢集。此案例中請將 my-cluster 取代為您在「步驟 1：建立叢集」建立的叢集之名稱。

```
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>
```

DeleteClusterAPI 作業只會刪除一個叢集。若要刪除多個叢集，DeleteCluster請呼叫您要刪除的每個叢集。您不需要等待某個叢集完成刪除，然後再刪除另一個叢集。

如需詳細資訊，請參閱DeleteCluster。

接下來做些什麼？

現在您已經嘗試了入門練習，您可以瀏覽下列各節，以深入瞭解 MemoryDB 和可用工具：

- 開始使用 AWS
- 適用於 Amazon Web Services 的工具
- AWS 命令列介面
- 內存數據庫的 Redis 的 API 參考。

管理節點

節點是 Redis 部署的內存數據庫的最小構建塊。節點屬於屬於集群的碎片。每個節點都會執行建立或上次修改叢集時所選擇的引擎版本。每個節點都有自己的網域名稱服務 (DNS) 名稱和連接埠。支援多種類型的 MemoryDB 節點，每個節點都有不同數量的相關記憶體和計算能力。

主題

- 內存數據庫節點和碎片
- 支援的節點類型
- 內存數據庫保留節點
- 替換節點

與節點有關的一些重要操作如下：

- 從叢集新增/移除節點
- 擴展
- 尋找連線端點

內存數據庫節點和碎片

碎片是節點的分層排列，每個節點包裹在一個集群中。碎片支援複寫。在一個碎片中，其中一個節點會做為讀取/寫入主要節點。碎片中的所有其他節點都會做為主要節點的唯讀複本。MemoryDB 支持集群中的多個碎片。此支援可讓您在 MemoryDB 叢集中分割資料。

內存數據庫支持通過碎片複製。API 操作DescribeClusters列出了與成員節點，節點名稱，端點和其他信息的碎片。

在創建 MemoryDB 集群之後，它可以被改變（縮放或縮小）。如需更多詳細資訊，請參閱「擴展」及「替換節點」。

建立新叢集時，您可以將舊叢集的資料傳送到新叢集，使其不會在一開始呈現空白狀態。如果您需要變更節點類型、引擎版本或從 Amazon ElastiCache to Redis 遷移，這樣做會很有幫助。如需詳細資訊，請參閱 製作手動快照 及 從快照還原。

支援的節點類型

內存數據庫支持以下節點類型。

記憶體最佳化

執行個體類型	基準頻寬 (Gbps)	高載頻寬 (Gbps)	增強型 I/O 多工（Redis 7.0.4+）	最低引擎版本
db.r7g.large	0.937	12.5	否	6.2
db.r7g.xlarge	1.876	12.5	否	6.2
db.r7g.2xlarge	3.75	15	是	6.2
db.r7g.4xlarge	7.5	15	是	6.2
db.r7g.8xlarge	15	N/A	是	6.2
db.r7g.12xlarge	22.5	N/A	是	6.2
db.r7g.16xlarge	30	N/A	是	6.2
db.r6g.large	0.75	10.0	否	6.2
db.r6g.xlarge	1.25	10.0	否	6.2
db.r6g.2xlarge	2.5	10.0	是	6.2
db.r6g.4xlarge	5.0	10.0	是	6.2
db.r6g.8xlarge	12	N/A	是	6.2
db.r6g.12xlarge	20	N/A	是	6.2
db.r6g.16xlarge	25	N/A	是	6.2

利用資料分層最佳化的記憶體

執行個體類型	基準頻寬 (Gbps)	高載頻寬 (Gbps)	增強型 I/O 多工（Redis 7.0.4+）	最低引擎版本
db.r6gd.xlarge	1.25	10	否	6.2
db.r6gd.2xlarge	2.5	10	否	6.2
db.r6gd.4xlarge	5.0	10	否	6.2
db.r6gd.8xlarge	12	不適用	否	6.2

一般用途節點

執行個體類型	基準頻寬 (Gbps)	高載頻寬 (Gbps)	增強型 I/O 多工（Redis 7.0.4+）	最低引擎版本
db.t4g.small	0.128	5.0	否	6.2
db.t4g.medium	0.256	5.0	否	6.2

如需 AWS 區域可用性，請參閱 Redis 定價的記憶體資料庫

所有節點類型都是在虛擬私有雲 (VPC) 中建立的。

內存數據庫保留節點

與隨需節點定價相比，預留節點可為您提供大幅 discount。預留節點不是實體節點，而是帳單 discount 適用於在您帳戶中使用隨選節點。預留節點的折扣與節點類型和 AWS 區域相關聯。

使用保留節點的一般程序如下：

- 檢閱有關可用保留節點產品的資訊
- 使用 AWS Command Line Interface 或 SDK 購買預留節點供應項目 AWS Management Console
- 檢閱現有預留節點的相關資訊

主題

- 預留節點概觀

預留節點概觀

當您購買 MemoryDB 保留節點時，您需要購買承諾產品，以便在預留節點期間，在特定節點類型上獲得折扣費率。若要使用 MemoryDB 保留節點，您可以建立新節點，就像對隨選節點一樣。您建立的新節點必須符合保留節點的規格。如果新節點的規格與您帳戶的現有保留節點相符，則會以保留節點提供的折扣費率向您收費。否則，節點會以隨需費率計費。您可以使用 AWS Management Console AWS CLI、或 MemoryDB API 來列出和購買可用的預留節點供應項目。

MemoryDB 為記憶體最佳化的 R7g、R6g 和 R6gD (含資料分層) 節點提供保留的節點。如需定價資訊，請參閱 Redis 定價的記憶體資料庫。

方案類型

預留節點提供三種類型：無預付、部分預付和全部預付款，可讓您根據預期的使用情況，針對 Redis 成本最佳化 MemoryDB。

無預付款 — 此選項可讓您存取預留節點，而無需預付款。無預付預留節點會依照期限內每小時的折扣小時費率計費，不論使用量為何，且無需預付款。

部分預付 — 此選項需要預先支付保留節點的一部分費用。期間內其餘的時數會以折扣後的每小時費率計費，無論是否有使用。

全額預付 — 在學期開始時支付全額費用，無論使用多少小時，在期限剩餘時間內都不會產生其他費用。

所有三種提供項目類型均提供一年期和三年期限。

彈性預留節點的大小

當您購買保留的節點時，您指定的一件事是節點類型，例如 db.r6g.xlarge。如需有關節點類型的詳細資訊，請參閱 Redis 定價的 MemoryDB。

如果您有一個節點，並且需要將其擴展到更大的容量，則保留的節點會自動應用到已擴展的節點。也就是說，您的預留節點會自動套用至相同節點系列中任何大小的使用量。具有相 AWS 同區域的節點可以使用大小靈活的預留節點。大小靈活的預留節點只能在其節點系列中進行擴展。例如，資料庫 .r6g.xlarge 的保留節點可以套用至 db.r6g.2xlarge，但不能套用至 db.r6gd.large，因為 db.r6g 和 db.r6gd 是不同的節點系列。

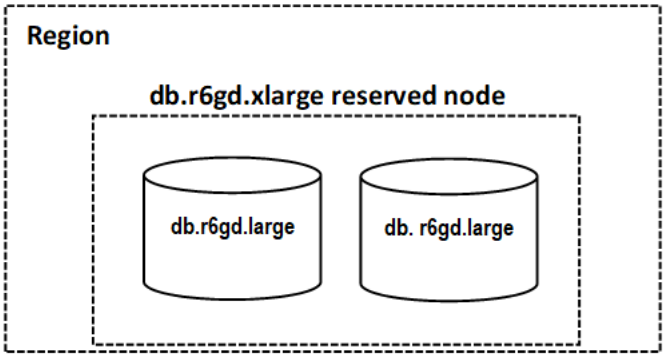
大小彈性意味著您可以在同一節點系列中的組態之間自由移動。例如，您可以從同一區域中的 r6g.xlarge 保留節點 (8 個標準化單位) 移至兩個 r6g.large 保留節點 (8 個標準化單位) (2*4 = 8 個標準化單位)，無需額外費用。 AWS

您可以使用標準化單位來比較不同保留節點大小的使用情況。例如，兩個 db.r6g.4xlarge 節點上的一個使用單位相當於一個 db.r6g.large 上的 16 個標準化使用單位。下表顯示了每個節點大小的標準化單位的數量：

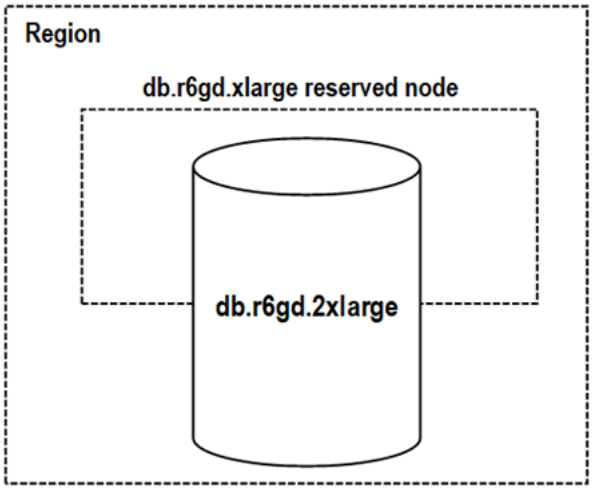
節點大小	標準化單位
小型	1
中型	2
大型	4
xlarge	8
2xlarge	16
4xlarge	32
6xlarge	48
8xlarge	64
10xlarge	80

節點大小	標準化單位
12xlarge	96
16xlarge	128

例如，您購買了 db.r6gd.xlarge 保留節點，並且在相同區域中的帳戶中有兩個執行中的 db.r6gd.large 保留節點。 AWS 在此情況下，帳單優惠會全額套用至兩個節點。



或者，如果您在相同 AWS 區域的帳戶中執行一個 db.r6gd.2xlarge 執行個體，則計費優惠會套用至預留節點使用量的 50%。



刪除保留節點

預留節點的條款涉及一年或三年的承諾。您無法取消保留的節點。但是，您可以刪除保留節點 discount 所涵蓋的節點。刪除保留節點 discount 所涵蓋之節點的程序與其他任何節點的程序相同。

如果刪除保留節點 discount 涵蓋的節點，則可以啟動另一個具有相容規格的節點。在此情況下，您仍可以在保留時間 (一或三年) 內繼續享有折扣費率。

使用保留節點

您可以使用 AWS Management Console AWS Command Line Interface、和記憶體資料庫 API 來處理保留的節點。

主控台

取得可用預留節點供應項目的定價和資訊

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在瀏覽窗格中，選擇 [保留的節點]。
3. 選擇 [購買預留節點]。
4. 針對節點類型，選擇您要部署的節點類型。
5. 在「數量」中，選擇您要部署的節點數目。
6. 在「期限」中，選擇要保留資料庫節點的時間長度。
7. 在 Offering type (方案類型) 中，選擇方案類型。

選取這些選項後，您可以在 [保留摘要] 底下查看定價資訊。

> **Important**
>
> 選擇「取消」可避免購買這些預留節點並產生任何費用。

取得有關可用預留節點供應項目的資訊之後，您可以使用這些資訊來購買訂購項目，如下列程序所示：

購買預留節點

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在瀏覽窗格中，選擇 [保留的節點]。
3. 選擇 [購買預留節點]。
4. 針對節點類型，選擇您要部署的節點類型。
5. 在「數量」中，選擇您要部署的節點數目。
6. 在「期限」中，選擇要保留資料庫節點的時間長度。

7. 在 Offering type (方案類型) 中，選擇方案類型。
8. (選用) 您可以將自己的識別碼指派給您購買的預留節點，以協助您追蹤這些節點。針對保留 ID，輸入保留節點的識別碼。

   選取這些選項後，您可以在 [保留摘要] 底下查看定價資訊。
9. 選擇 [購買預留節點]。
10. 您的預留節點已購買，然後顯示在「預留節點」清單中。

取得 AWS 帳戶預留節點的相關資訊

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在瀏覽窗格中，選擇 [保留的節點]。
3. 會顯示您帳戶的預留節點。若要查看特定保留節點的詳細資訊，請在清單中選擇該節點。然後，您可以在詳細信息中查看有關該節點的詳細信息。

AWS Command Line Interface

下列`describe-reserved-nodes-offerings`範例會傳回保留節點供應項目的詳細資訊。

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

這會產生類似下列的輸出：

```
{
    "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"
                }
```

```
            ]
        }
    ]
}
```

您也可以傳遞下列參數來限制傳回的範圍：

- `--reserved-nodes-offering-id` – 您想要購買之方案的 ID。
- `--node-type`— 節點類型篩選器值。使用此參數可僅顯示符合指定節點類型的保留區。
- `--duration`— 持續時間篩選器值，以年或秒為單位指定。使用此參數可僅顯示此期間的保留項目。
- `--offering-type`— 使用此參數可僅顯示符合指定提供項目類型的可用訂購項目。

取得有關可用預留節點供應項目的資訊之後，您可以使用這些資訊來購買提供項目。

下列`purchase-reserved-nodes-offering`範例會購買新的預留節點

若為 Linux、macOS 或 Unix：

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

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

針對 Windows：

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

- `--reserved-nodes-offering-id`代表要購買的預留節點名稱。
- `--reservation-id`是客戶指定的識別碼，以追蹤此保留項目。

> **Note**
>
> 保留 ID 是用於追蹤此保留項目的唯一客戶指定識別碼。如果未指定此參數，MemoryDB 會自動產生保留項目的識別碼。

- --node-count是要保留的節點數目。它默認為 1。

這會產生類似下列的輸出：

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

購買預留節點後，您可以取得有關預留節點的資訊。

下列describe-reserved-nodes範例會傳回此帳戶之保留節點的相關資訊。

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

這會產生類似下列的輸出：

```
{
    "ReservedNodes": [
        {
            "ReservationId": "ri-2022-12-16-00-28-40-600",
            "ReservedNodesOfferingId": "0193cc9d-7037-4d49-b332-xxxxxxxxxxxx",
            "NodeType": "db.xxx.large",
```

```
            "StartTime": 1671150737.969,
            "Duration": 94608000,
            "FixedPrice": $xxx.xx,
            "NodeCount": 1,
            "OfferingType": "Partial Upfront",
            "State": "active",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": $xx.xx,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxxx:reservednode/
ri-2022-12-16-00-28-40-600"
        }
    ]
}
```

您也可以傳遞下列參數來限制傳回的範圍：

- `--reservation-id`— 您可以將自己的識別碼指派給您購買的預留節點，以協助追蹤它們。
- `--reserved-nodes-offering-id`— 提供項目識別碼篩選值。使用此參數可僅顯示符合指定提供項目識別碼的已購買保留項目。
- `--node-type`— 節點類型篩選器值。使用此參數可僅顯示符合指定節點類型的保留區。
- `--duration`— 持續時間篩選器值，以年或秒為單位指定。使用此參數可僅顯示此期間的保留項目。
- `--offering-type`— 使用此參數可僅顯示符合指定提供項目類型的可用訂購項目。

記憶體資料庫 API

下列範例會示範如何針對保留節點使用 MemoryDB 查詢 API：

DescribeReservedNodesOfferings

傳回保留節點提供項目的詳細資訊。

```
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>
```

以下參數限制了返回內容的範圍：

- ReservedNodesOfferingId代表要購買的預留節點名稱。
- Duration— 持續時間篩選器值，以年或秒為單位指定。使用此參數可僅顯示此期間的保留項目。
- NodeType— 節點類型篩選器值。使用此參數可僅顯示符合指定節點類型的提供項目。
- OfferingType— 使用此參數可僅顯示符合指定提供項目類型的可用訂購項目。

取得有關可用預留節點供應項目的資訊之後，您可以使用這些資訊來購買提供項目。

PurchaseReservedNodesOffering

可讓您購買預留的節點供應項目。

```
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代表要購買的預留節點名稱。

- ReservationID是客戶指定的識別碼，以追蹤此保留項目。

> **Note**
>
> 保留 ID 是用於追蹤此保留項目的唯一客戶指定識別碼。如果未指定此參數，MemoryDB 會自動產生保留項目的識別碼。

- NodeCount是要保留的節點數目。它默認為 1。

購買預留節點後，您可以取得有關預留節點的資訊。

DescribeReservedNodes

傳回此帳戶之保留節點的相關資訊。

```
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>
```

以下參數限制了返回內容的範圍：

- ReservedNodesOfferingId表示保留節點的名稱。
- ReservationID— 您可以將自己的識別碼指派給您購買的預留節點，以協助追蹤它們。
- NodeType— 節點類型篩選器值。使用此參數可僅顯示符合指定節點類型的保留區。
- Duration— 持續時間篩選器值，以年或秒為單位指定。使用此參數可僅顯示此期間的保留項目。
- OfferingType— 使用此參數可僅顯示符合指定提供項目類型的可用訂購項目。

檢視預留節點的帳單

您可以在的 [帳單儀表板] 中檢視預留節點的帳單 AWS Management Console。

若要檢視保留的節點帳單

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從主機頂端的 [搜尋] 按鈕中，選擇 [帳單]。
3. 從資料面板的左側選擇「帳單」。
4. 在 [AWS 服務費用] 底下，展開 [記憶體 DB]。
5. 展開預留節點所在的 AWS 區域，例如美國東部 (維吉尼亞北部)。

Amazon MemoryDB 預留執行個體下會顯示您當月的 CreateCluster 預留節點及其小時費用。

Amazon MemoryDB CreateCluster Reserved Instances		
AmazonMemoryDB, db.r6g.large reserved instance applied	81.000 Hrs	
AmazonMemoryDB, db.r6g.4xlarge reserved instance applied	324.000 Hrs	
AmazonMemoryDB, db.r6g.4xlarge reserved instance applied	162.000 Hrs	
USD (hourly fee per AmazonMemoryDB, db.r6g.large instance	1,488.000 Hrs	
USD (hourly fee per AmazonMemoryDB, db.r6gd.2xlarge instance	744.000 Hrs	
USD 0 hourly fee per AmazonMemoryDB, db.r6g.4xlarge instance	744.000 Hrs	
USD (hourly fee per AmazonMemoryDB, db.r6gd.xlarge instance	744.000 Hrs	
USD 1 hourly fee per AmazonMemoryDB, db.r6gd.4xlarge instance	2,976.000 Hrs	

替換節點

MemoryDB 經常通過修補程序和升級來升級其車隊，通常無縫。但是，我們不時需要重新啟動 MemoryDB 節點，以將強制性的操作系統更新應用於基礎主機。必須進行這些替換才能套用升級，以強化安全、可靠性和操作效能。

您可以選擇在排程的節點替換時間之前，隨時自行管理這些替換。當您自行管理替換時，執行個體會在重新啟動節點時收到 OS 更新，而排程的節點替換將會取消。您可能會繼續收到提醒，指出節點即將進行替換。若您已手動減少維護的需求，您可以忽略這些提醒。

> **Note**
>
> 由記憶體資料庫為 Redis 自動產生的取代節點可能具有不同的 IP 位址。您必須負責檢閱應用程式組態，以確保節點與適當的 IP 位址相關聯。

下列清單會識別 MemoryDB 排程其中一個節點進行取代時可採取的動作：

記憶體資料庫節點取代選項

- 什麼都不做-如果你什麼都不做，MemoryDB 會按計劃替換節點。

  如果節點是異地同步備份叢集的成員，MemoryDB 會在修補、更新和其他維護相關節點取代期間提供改善的可用性。

  叢集提供傳入寫入要求時完成取代。
- 變更維護時段 — 對於排程的維護事件，您會收到來自 MemoryDB 的電子郵件或通知事件。在這種情況下，如果在排定的替換時間之前變更維護時段，則現在將在新的時間替換您的節點。如需詳細資訊，請參閱 修改記憶體資料庫叢集。

  > **Note**
  >
  > 只有在 MemoryDB 通知包含維護時段時，才能透過移動維護時段來變更替換時段。若通知並未包含維護時段，您便無法變更替換時間。

  例如，假設現在是 11 月 9 日星期四下午 3:00，下一個維護時段是 11 月 10 日星期五下午 5:00。以下是三種情況及其結果：
  - 您將維護時段變更為星期五下午 4:00，在目前的日期時間之後、下一個排定的維護時段之前。節點將於 11 月 10 日星期五下午 4:00 進行替換。
  - 您將維護時段變更為星期六下午 4:00，在目前的日期時間之後，以及下一個排定的維護時段之後。節點將於 11 月 11 日星期六下午 4:00 進行替換。
  - 您將維護時段變更為星期三 16:00，早於目前日期和時間。節點將於 11 月 15 日星期三下午 4:00 進行替換。

  如需說明，請參閱 管理維護作業。

管理叢集

大多數 MemoryDB 操作都是在集群級別執行的。您可以將叢集設定為含特定數量的節點和一個參數群組，以控制每個節點的屬性。叢集內的所有節點都設計為相同節點類型，並具備相同的參數和安全群組設定。

每個叢集都必須有一個叢集識別符。叢集識別符是客戶針對叢集提供的名稱。此標識符指定與 MemoryDB API 和AWS CLI命令進行交互時的特定集群。在 AWS 區域內，叢集識別碼必須是該客戶的唯一標記。

記憶體資料庫叢集的設計是為了使用 Amazon EC2 執行個體存取而設計。您只能在以 Amazon VPC 服務為基礎的虛擬私有雲端 (VPC) 中啟動 MemoryDB 叢集，但可以從外部存取它。AWS如需詳細資訊，請參閱從外部訪問內存 DB 資源AWS。

資料分層

使用 r6gd 系列節點類型的叢集會在記憶體和本機 SSD (固態硬碟) 儲存之間進行資料分層。除了將資料存放在記憶體中之外，資料分層還能藉由在每個叢集節點中使用成本較低的固態硬碟 (SSD)，為 Redis 工作負載提供全新的價格效能方案。與其他節點類型類似，寫入 r6gd 節點的資料會持久儲存在異地同步備份交易記錄中。資料分層非常適合定期存取高達 20% 的整體資料集的工作負載，以及在存取 SSD 資料時可容忍額外延遲的應用程式。

在具有資料分層的叢集上，MemoryDB 監控其存放的每個項目的最後存取時間。當可用記憶體 (DRAM) 完全耗盡時，MemoryDB 會使用最近使用的 (LRU) 演算法，將不常存取的項目從記憶體移至 SSD。當隨後存取 SSD 上的資料時，MemoryDB 會在處理請求之前自動並以異步的方式將其移回記憶體。如果您的工作負載只會定期存取其資料的子集，則資料分層是以符合成本效益的方式擴展容量的最佳方式。

請注意，使用資料分層時，金鑰本身一律會保留在記憶體中，而 LRU 則會控制記憶體與磁碟上值的位置。一般而言，建議在使用資料分層時，金鑰大小小於您值的大小。

資料分層專為盡量降低對應用程式工作負載的效能影響所設計。例如，假設 500 個位元組的字串值，與讀取在記憶體中的資料相比，可以預期請求存放在 SSD 上的資料時會有額外 450 微秒的延遲。

使用最大的資料分層節點大小 (db.r6gd.8xlarge)，可以在單一 500 個節點的叢集中存放高達 500 TB (使用 1 個僅供讀取複本時為 250 TB)。對於資料分層，MemoryDB 會為每個節點保留 19% 的 (DRAM) 記憶體，以供非資料使用。資料分層與所有 Redis 命令和 MemoryDB 中支援的資料結構相容。不需要任何用戶端變更就能使用此功能。

主題

- 最佳實務
- 限制
- 資料層分定價
- 監控
- 使用資料分層
- 在資料分層啟用的情況下，將資料從快照還原到叢集

最佳實務

建議遵循下列最佳實務：

- 資料分層非常適合定期存取高達 20% 的整體資料集的工作負載，以及在存取 SSD 資料時可容忍額外延遲的應用程式。
- 使用資料分層節點上可用的 SSD 容量時，建議值的大小大於金鑰大小。值大小不能大於 128MB；否則將不會移至磁碟。項目在 DRAM 和 SSD 之間移動時，金鑰將一律保留在記憶體中，而且只有值會移至 SSD 層。

限制

資料分層具有下列限制：

- 使用的節點類型必須來自 r6gd 系列，該系列在下列區域可用：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 和 sa-east-1。
- 無法將 r6gd 叢集的快照還原到另一個叢集，除非該叢集也使用 r6gd。
- 無法將快照匯出到 Amazon S3 以用於資料分層叢集。
- 不支援無叉 (forkless) 儲存。
- 不支援從資料分層叢集 (例如，使用 r6gd 節點類型的叢集) 擴展到未使用資料分層的叢集 (例如，使用 r6g 節點類型的叢集)。
- 資料分層僅支援volatile-lru、allkeys-lru 和 noeviction 最大記憶體政策。
- 大於 128 MiB 的項目不會移至固態硬碟。

資料層分定價

與 R6g 節點 (僅記憶體) 相比，R6gd 節點的總容量 (記憶體 + SSD) 多出 5 倍，而且在以最大使用率執行時，可協助您節省 60% 以上的儲存成本。如需詳細資訊，請參閱 MemoryDB 定價。

監控

MemoryDB 提供專為監控使用資料分層的效能叢集而設計的指標。若要監控與 SSD 相比的 DRAM 中項目比例，您可以使用CurrItems指標記憶體資料庫的度量。您可以按以下方式計算百分比：(CurrItems with Dimension: Tier = Memory * 100) / (CurrItems with no dimension filter)。記憶體中的項目百分比下降到 5% 以下時，建議您考慮擴展 MemoryDB 叢集。

如需詳細資訊，請參閱位於的使用資料分層的 MemoryDB 叢集指標記憶體資料庫的度量。

使用資料分層

使用 AWS Management Console 使用資料分層

在建立叢集時，可以從 r6gd 系列中選取節點類型來使用資料分層，例如 db.r6gd.xlarge。選取該節點類型會自動啟用資料分層。

如需有關建立叢集的詳細資訊，請參閱 步驟 1：建立叢集。

使用 AWS CLI 啟用資料分層

使用建立叢集時AWS CLI，可以從 r6gd 系列中選取節點類型來使用資料分層，例如 db.r6gd.xlarge，然後設定`--data-tiering`參數。

從 r6gd 系列中選取節點類型時，無法選擇退出資料分層。如果您設定 `--no-data-tiering` 參數，操作將會失敗。

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

執行此操作之後，將會看到類似如下的回應：

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "creating",
```

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

在資料分層啟用的情況下，將資料從快照還原到叢集

在資料分層啟用的情況下，可以使用 (主控台)、(AWSCLI) 或 (MemoryDB API) 將快照還原到新的叢集。當使用 r6gd 系列中的節點類型建立叢集時，會啟用資料分層。

在資料分層啟用的情況下，將資料從快照還原到叢集 (主控台)

若要在資料分層啟用的情況下，將快照還原到新叢集，請按照以下步驟操作：從快照還原 (主控台)

請注意，若要啟用資料從 r6gd 系列節點類型，您需要從 r6gd 系列中選取節點類型。

在資料分層啟用的情況下，將資料從快照還原到叢集 (AWSCLI)

使用建立叢集時AWS CLI，若從 r6gd 系列中選取節點類型，例如 db.r6gd.xlarge，然後設定*--data-tiering*參數，即預設為使用資料從 r6gd.xlarge。

從 r6gd 系列中選取節點類型時，無法選擇退出資料分層。如果您設定 `--no-data-tiering` 參數，操作將會失敗。

若為 Linux、macOS 或 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
```

若為 Linux、macOS 或 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
```

執行此操作之後，將會看到類似如下的回應：

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

```
}
```

準備叢集

接下來，您可以找到有關使用 MemoryDB 控制台，或 MemoryDB API 創建集群的說明。AWS CLI

每當您建立叢集時，最好進行一些準備工作，因此您不需要立即升級或進行變更。

主題

- 判斷要求

判斷要求

準備

釐清下列問題的答案有助於提高叢集的建立流暢度：

- 開始建立叢集之前，請務必在同一個 VPC 中建立子網路群組。或者，您可以使用提供的預設子網路群組。如需詳細資訊，請參閱子網路和子網路群組。

  記憶體資料庫的設計可從內部AWS使用 Amazon EC2 進行存取。但是，如果您在以 Amazon VPC 為基礎的 VPC 中啟動，則可以從外部提供存取權。AWS如需詳細資訊，請參閱從外部訪問內存 DB 資源AWS。
- 您是否需要自訂任何參數值？

  如果需要，請建立自訂參數群組。如需詳細資訊，請參閱建立參數群組。
- 您是否需要建立 VPC 安全性群組？

  如需詳細資訊，請參閱 VPC 中的安全性。
- 您要如何實作容錯能力？

  如需詳細資訊，請參閱緩解故障。

主題

- 記憶體和處理器要求
- 記憶體資料庫叢集配置
- 增強型 I/O 多工處理

- 擴展要求
- 存取要求
- 區域和可用區域

記憶體和處理器要求

內存數據庫的 Redis 的基本構建塊是節點。節點在碎片中配置以形成集群。當您判斷要為叢集使用何種節點類型時，請一併考量叢集的節點組態和您要存放的資料量。

記憶體資料庫叢集配置

記憶體資料庫叢集由 1 到 500 個碎片組成。MemoryDB 群集中的數據跨集群中的碎片進行分區。您的應用程式會使用稱為端點的網路位址與 MemoryDB 叢集連線。除了節點端點之外，MemoryDB 叢集本身也有一個稱為叢集端點的端點。您的應用程式可以使用此端點從叢集讀取或寫入叢集，並決定要從哪個節點讀取或寫入到 MemoryDB。

增強型 I/O 多工處理

如果您執行的是 Redis 7.0 版或更高版本，您將透過增強的 I/O 多工處理獲得額外的加速，其中每個專用網路 IO 執行緒會將多個用戶端的命令管道至 Redis 引擎，並利用 Rediss 有效率地批次處理命令的能力。如需詳細資訊，請參閱超快效能和the section called “支援的節點類型”.

擴展要求

所有叢集都可以擴展到較大的節點類型。當您擴展 MemoryDB 叢集時，您可以在線上執行，讓叢集保持可用，或者您可以從快照植入新叢集，並避免讓新叢集開始為空。

如需詳細資訊，請參閱本指南中的 擴展 。

存取要求

根據設計，記憶體資料庫叢集是從 Amazon EC2 執行個體存取的。MemoryDB 叢集的網路存取僅限於建立叢集的帳戶。因此，您必須先授權對叢集的輸入，才能從 Amazon EC2 執行個體存取叢集。如需詳細說明，請參閱本指南的步驟 2：授權對叢集的存取。

區域和可用區域

透過將 MemoryDB 叢集定位在應用程式附近的AWS區域中，您可以減少延遲。如果您的叢集有多個節點，將節點安置在不同可用區域中可降低故障對叢集的影響。

如需詳細資訊，請參閱下列內容：

- 選擇區域與可用區域
- 緩解故障

建立叢集

Redis 的內存數據庫提供了三種方法來創建一個集群。如需詳細資訊，請參閱步驟 1：建立叢集。

檢視叢集的詳細資訊

您可以使用 MemoryDB 主控台或 MemoryDB API 來檢視有關一或多個叢集的詳細資訊。AWS CLI

檢視 MemoryDB 叢集 (主控台) 的詳細資料

下列程序詳細說明如何使用 MemoryDB 主控台檢視 MemoryDB 叢集的詳細資料。

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看叢集的詳細資訊，請選擇叢集名稱左邊的圓鈕，然後選擇 [檢視詳細資訊]。您也可以直接按一下叢集來檢視叢集詳細資訊頁面。

   [叢集詳細資料] 頁面會顯示叢集的詳細資訊，包括叢集端點。您可以使用「叢集詳細資訊」頁面中的多個頁籤來檢視更多詳細資訊。
3. 選擇 [碎片和節點] 索引標籤，以查看叢集的碎片清單，以及每個碎片中的節點數目。
4. 若要檢視節點上的特定資訊，請展開下表中的碎片。或者，您也可以使用搜尋方塊搜尋碎片。

   這樣做會顯示每個節點的相關資訊，包括其可用區域、插槽/金鑰空間和狀態。
5. 選擇「指標」標籤以監視其各自的處理作業，例如「CPU 使用率」和「引擎 CPU 使用率」。如需詳細資訊，請參閱記憶體資料庫的度量。
6. 選擇 [網路和安全性] 索引標籤，以查看子網路群組和安全性群組的詳細資料。

   a. 在子網路群組中，您可以看到子網路群組的名稱、子網路所屬 VPC 的連結以及子網路群組的 Amazon 資源名稱 (ARN)。

   b. 在安全性群組中，您可以看到安全性群組 ID、名稱和說明。
7. 選擇 [維護和快照] 索引標籤以查看快照設定的詳細資料。

   a. 在快照中，您可以查看是否已啟用自動快照、快照保留期和快照視窗。

   b. 在快照中，您會看到此叢集的任何快照清單，包括快照名稱、大小、碎片數目和狀態。

   如需詳細資訊，請參閱快照和還原 。
8. 選擇維護和快照索引標籤，以查看「維護時段」的詳細資訊，以及任何擱置的 ACL、重新碎片或服務更新。如需詳細資訊，請參閱管理維護作業。
9. 選擇 [服務更新] 索引標籤，以查看適用於此叢集之任何服務更新的詳細資訊。如需詳細資訊，請參閱Redis 內存數據庫中的服務更新。

10. 選擇 [標記] 索引標籤，查看與此叢集相關聯之任何資源或成本配置標記的詳細資訊。如需詳細資訊，請參閱標記快照。

檢視叢集的詳細資訊 (AWSCLI)

您可以使用 AWS CLI `describe-clusters` 命令來檢視叢集的詳細資訊。如果省略 `--cluster-name` 參數，則會傳回多個叢集 (最多 `--max-results` 個) 的詳細資訊。如果包含 `--cluster-name` 參數，則會傳回指定叢集的詳細資訊。您可以使用 `--max-results` 參數限制傳回的記錄數量。

以下程式碼會列出 `my-cluster` 的詳細資訊。

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

以下程式碼清單會列出最多 25 個叢集的詳細資訊。

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

Example

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

下面的 JSON 輸出顯示了響應：

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

```
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": 1629230643.961,
                            "Endpoint": {
                                "Address": "my-cluster-0001-001.my-
cluster.abcdef.memorydb.us-east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "CreateTime": 1629230644.025,
                            "Endpoint": {
                                "Address": "my-cluster-0001-002.my-
cluster.abcdef.memorydb.us-east-1.amazonaws.com",
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 2
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.abcdef.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "default",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:000000000:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
```

```
            "MaintenanceWindow": "sat:06:30-sat:07:30",
            "SnapshotWindow": "04:00-05:00",
            "ACLName": "open-access",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true,
        }
```

如需詳細資訊，請參閱記憶體AWS CLI資料庫主題。describe-clusters

檢視叢集的詳細資料 (記憶體資料庫 API)

您可以使用 MemoryDB API DescribeClusters 動作檢視叢集的詳細資料。如果包含 ClusterName 參數，則會傳回指定叢集的詳細資訊。如果省略 ClusterName 參數，則會傳回最多 MaxResults 個叢集 (預設值為 100) 的詳細資訊。MaxResults 的值不可小於 20 或大於 100。

以下程式碼會列出 my-cluster 的詳細資訊。

```
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>
```

以下程式碼清單會列出最多 25 個叢集的詳細資訊。

```
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>
```

如需詳細資訊，請參閱記憶體資料庫 API 參考主題。DescribeClusters

修改記憶體資料庫叢集

除了從叢集新增或移除節點之外，您還可能需要對現有叢集進行其他變更，例如新增安全性群組、變更維護時段或參數群組。

建議您將維護時段落在使用量最低的時段。您可能需要不時進行調整。

當您變更叢集的參數時，變更會立即套用至叢集。無論是變更叢集的參數群組本身或是叢集的參數群組內的參數值，均適用此情況。

您也可以更新叢集的引擎版本。例如，您可以選取新的引擎次要版本，MemoryDB 將立即開始更新叢集。

使用 AWS Management Console

修改叢集

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在右上角清單中，選擇您要修改之叢集所在的 AWS 區域。
3. 從左側導覽中，前往 [叢集]。從叢集詳細資料中，使用圓鈕選取叢集，然後移至動作，然後移至修改。
4. 便會顯示 「修改」頁面。
5. 在「修改」視窗中，進行所需的修改。選項包括：

   - 描述
   - 子網路群組
   - VPC 安全群組
   - 節點類型

   > **Note**
   >
   > 如果叢集使用 r6gd 系列中的節點類型，則只能從該系列中選擇不同的節點大小。如果從 r6gd 系列中選擇節點類型，則將自動啟用資料分層。如需詳細資訊，請參閱資料分層。

   - Redis 的版本兼容性
   - 啟用自動快照

- 快照保留期
- 快照視窗
- Maintenance window (維護時段)
- SNS 通知的主題

6. 選擇儲存變更。

您也可以移至叢集詳細資訊頁面，然後按一下修改來修改叢集。如果要修改叢集的特定段落，可以移至「叢集詳細資訊」頁面中的個別頁籤，然後按一下修改。

使用 AWS CLI

您可以使用 AWS CLI `update-cluster` 操作修改現有的叢集。若要修改叢集的組態值，請指定叢集 ID、要變更的參數以及參數的新值。下方範例會變更名稱為 `my-cluster` 之叢集的維護時段，並立即套用變更。

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

如需詳細資訊，請參閱AWS CLI命令參考中的更新叢集。

使用記憶體資料庫 API

您可以使用記憶體資料庫 API UpdateCluster作業修改現有叢集。若要修改叢集的組態值，請指定叢集 ID、要變更的參數以及參數的新值。下方範例會變更名稱為 `my-cluster` 之叢集的維護時段，並立即套用變更。

```
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>
```

從叢集新增/移除節點

您可以使用AWS Management Console、或 MemoryDB API 從叢集中新增或移除節點。AWS CLI

使用 AWS Management Console

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從叢集清單中，選擇要新增或移除節點的叢集名稱。
3. 在「碎片和節點」選項卡下，選擇「添加/刪除節點」
4. 在 [新增節點數目] 中，輸入您想要的節點數目。
5. 選擇 Confirm (確認) 。

> **Important**
>
> 如果將節點數設定為 1，將不再啟用異地同步備份。您也可以選擇啟用自動容錯移轉。

使用 AWS CLI

1. 識別您要移除的節點名稱。如需詳細資訊，請參閱檢視叢集的詳細資訊。
2. 搭配使用 `update-cluster` CLI 操作與要移除的節點清單，如下列範例所示。

   若要使用命令列界面移除叢集中的節點，請搭配使用 `update-cluster` 命令與下列參數：

   - `--cluster-name`您要從中移除節點的叢集識別碼。
   - `--replica-configuration`— 可讓您設定複本的數量：
     - `ReplicaCount`— 設定此內容以指定您想要的複本節點數目。
   - `--region` 指定您想要從中移除節點之叢集的 AWS 區域。

   若為 Linux、macOS 或 Unix：

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

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

針對 Windows：

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

如需詳細資訊，請參閱 AWS CLI 主題 update-cluster。

使用記憶體資料庫 API

若要使用 MemoryDB API 移除節點，請使用叢集名稱和要移除的節點清單呼叫 UpdateCluster API 作業，如下所示：

- ClusterName您要從中移除節點的叢集識別碼。
- ReplicaConfiguration— 可讓您設定複本的數量：
  - ReplicaCount— 設定此內容以指定您想要的複本節點數目。
- Region 指定您想要從中移除節點之叢集的 AWS 區域。

如需詳細資訊，請參閱UpdateCluster。

存取叢集

Redis 執行個體是設計為透過 Amazon EC2 執行個體來存取。

您可以從同一個 Amazon VPC 中的 Amazon EC2 執行個體來存取內存 DB 節點。或者，您可以透過使用 VPC 互連，從不同 Amazon VPC 中的 Amazon EC2 存取內存 DB 節點。

主題

- 授予對您叢集的存取
- 從外部訪問內存 DB 資源AWS

授予對您叢集的存取

您只能從同一個 Amazon VPC 中執行的 Amazon EC2 執行個體連接至內存 DB 叢集。在此情況下，您需要授權透過網路輸入至叢集。

授權從 Amazon VPC 安全群組透過網路輸入至叢集

1. 請登入 AWS Management Console，並在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在左側導覽窗格中，在網路和安全，選擇安全群組。
3. 從安全群組的清單中，選擇要用於 Amazon VPC 的安全群組。除非您建立了安全叢集供 MemyDB 使用，否則此安全用户組將會命名為預設值。
4. 選擇 Inbound (傳入) 標籤，然後執行下列動作：

    a. 選擇 Edit (編輯)。

    b. 選擇 Add rule (新增規則)。

    c. 在 Type (類型) 欄中，選擇 Custom TCP rule (自訂 TCP 規則)。

    d. 在 Port range (連接埠範圍) 方塊中，輸入要用於叢集節點的連接埠號碼。此號碼必須與您啟動叢集時指定的號碼相同。Redis 的預設連接埠為**6379**。

    e. 在 中來源方塊中，選擇Anywhere的連接埠範圍 (0.0.0/0)，讓任何您在 Amazon VPC 內啟動的 Amazon EC2 執行個體都能連接至內存 DB 節點。

    > **Important**
    >
    > 將 MemyDB 叢集開啟到 0.0.0.0/0 不會向網際網路公開叢集，因為它沒有任何公有 IP 地址，因此無法從 VPC 外部進行存取。不過，預設安全群組可能會套用到客戶帳戶

中的其他 Amazon EC2 執行個體，而這些執行個體可能有公有 IP 地址。如果他們正巧在預設連接埠上執行某些項目，就可能會意外公開該服務。因此，建議您建立一個由 MeryDB 專用的 VPC 安全叢集。如需詳細資訊，請參閱自訂安全群組。

f. 選擇 Save (儲存)。

當您在 Amazon VPC 中啟動 Amazon EC2 執行個體時，該執行個體就能夠連接至您的 MemyDB 叢集。

從外部訪問內存 DB 資源AWS

MemyDB 是一種設計用於 VPC 的內部服務。由於網際網路流量和安全性考量的延遲，因此不鼓勵外部存取。但是，若測試或開發用途需要對 MemoryDB 的外部存取，則可以透過 VPN 完成。

使用AWSClient VPN 來允許外部存取內存 DB 節點，具有下列好處：

- 有限存取核准的使用者或身份驗證金鑰；
- VPN 用戶端和 AWS VPN 端點之間的加密流量；
- 有限存取特定子網路或節點；
- 輕鬆撤銷使用者的存取或身份驗證金鑰；
- 稽核連線；

以下程序示範如何：

主題

- 建立憑證授權機構
- 設定 AWS Client VPN 元件
- 設定 VPN 用戶端

建立憑證授權機構

您可以使用不同的技術或工具來建立憑證授權單位 (CA)。我們建議由 OpenVPN 專案提供的 easy-rsa 公用程式。無論您選擇哪個選項，請確保金鑰安全無虞。下列程序會下載 easy-rsa 指令碼、建立憑證授權單位和金鑰來驗證第一個 VPN 用戶端：

- 若要建立初始憑證，請開啟終端機並執行下列動作：
  - `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`

  包含憑證的 pki 子目錄將建立於 easy-rsa 下。

- 將伺服器憑證提交至 AWS Certificate Manager (ACM)：
  - 在 ACM 主控台上，選取 Certificate Manager。
  - 選取 Import certificate (匯入憑證)。
  - 在 Certificate body (憑證內文) 欄位裡輸入 `easy-rsa/pki/issued/server.crt` 檔案中可用的公有金鑰憑證。
  - 在 Certificate private key (憑證私有金鑰) 欄位裡貼上 `easy-rsa/pki/private/server.key` 中可用的私有金鑰。請確定選取 `BEGIN AND END PRIVATE KEY` 之間的所有直線 (包括 `BEGIN` 和 `END` 直線)。
  - 在 Certificate chain (憑證鏈) 欄位中貼上 `easy-rsa/pki/ca.crt` 檔案中可用的 CA 公有金鑰。
  - 選取 Review and import (檢閱和匯入)。
  - 選取 Import (匯入)。

  若要使用 AWS CLI 將伺服器的憑證提交至 ACM，請執行下列命令：`aws acm import-certificate --certificate fileb://easy-rsa/pki/issued/server.crt --private-key file://easy-rsa/pki/private/server.key --certificate-chain file://easy-rsa/pki/ca.crt --region region`

  請注意憑證 ARN 以供日後使用。

設定 AWS Client VPN 元件

使用 AWS 主控台

在 AWS 主控台上，選取 Services (服務)，然後選取 VPC。

在 Virtual Private Network (虛擬私有網路) 下，選取 Client VPN Endpoints (客戶端 VPN 端點)，然後執行下列動作：

設定 AWS Client VPN 元件

- 選取 Create Client VPN Endpoint (建立客戶端 VPN 端點)。
- 指定下列選項：
  - Client IPv4 CIDR (客戶端 IPv4 CIDR)：使用具有至少 /22 範圍網路遮罩的私有網路。請確定選取的子網路不會與 VPC 網路的位址衝突。範例：10.0.0/22.
  - 在 Server certificate ARN (伺服器憑證 ARN) 中，選取先前匯入的憑證 ARN。
  - 選取 Use mutual authentication (使用交互身份驗證)。

- 在 Client certificate ARN (用戶端憑證 ARN) 中，選取先前匯入之憑證的 ARN。
- 選取 Create Client VPN Endpoint (建立客戶端 VPN 端點)。

使用 AWS CLI

執行以下 命令：

```
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
```

輸出範例：

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

將目標網路與 VPN 端點建立關聯

- 選取新的 VPN 端點，然後選取 Associations (關聯) 標籤。
- 選取 Associations (關聯) 並指定下列選項。
  - VPC：選擇內存 DB 叢集的 VPC。
  - 選取其中一個內存 DB 叢集的網路。如果有疑問，請檢子網路群組在內存數據庫儀錶板上。
  - 選取 Associations (關聯)。如有必要，請針對剩餘的網路重複這些步驟。

使用 AWS CLI

執行以下 命令：

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

輸出範例：

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

檢閱 VPN 安全性群組

VPN 端點會自動採用 VPC 的預設安全群組。檢查傳入和傳出規則，並確認安全羣組是否允許從 VPN 網路 (在 VPN 端點設定上定義) 到服務連接埠上 MemyDB 網路的流量 (預設為 Redis 的 6379)。

如果您需要變更指派給 VPN 端點的安全群組，請依照下列步驟進行：

- 選取目前安全群組。
- 選取 Apply Security Group (套用安全群組)。
- 選取新的安全群組。

使用 AWS CLI

執行以下 命令：

```
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
```

輸出範例：

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

> **Note**
>
> 內存 DB 安全用户端也需要允許來自 VPN 用户端的流量。根據 VPC 網路，用戶端的位址將以 VPN 端點位址遮蔽。因此，在 MemyDB 安全羣組上建立傳入規則時，請考慮使用 VPC 網路 (而非 VPN 用户端的網路)。

授權目標網路的 VPN 存取

在 Authorization (授權) 標籤上，選取 Authorize Ingress (授權輸入) 並指定下列項目：

- 要啟用訪問的目標網絡：使用 0.0.0.0/0 允許存取任何網路 (包括網際網路) 或限制 MemyDB 網路/主機。
- 在 Grant access to: (授予存取：) 下，選取 Allow access to all users (允許存取所有使用者)。

- 選取 Add Authorization Rules (新增授權規則)。

使用 AWS CLI

執行以下 命令：

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

輸出範例：

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

允許從 VPN 用戶端存取網際網路

如果您需要透過 VPN 瀏覽網際網路，則需要建立額外的路由。選取 Route Table (路由表) 標籤，然後選取 Create Route (建立路由)。

- 路由目的地：0.0.0.0/0
- 目標 VPC 子網 ID：選取其中一個可存取網際網路的相關子網路。
- 選取 Create Route (建立路由)。

使用 AWS CLI

執行以下 命令：

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

輸出範例：

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

設定 VPN 用戶端

在 AWS 用戶端 VPN 儀表板上，選取最近建立的 VPN 端點，然後選取 Download Client Configuration (下載用戶端組態)。複製組態檔案，以及檔案 `easy-rsa/pki/issued/client1.domain.tld.crt` 和 `easy-rsa/pki/private/client1.domain.tld.key`。編輯組態檔案，並變更或新增下列參數：

- 憑證：新增一個指向 client1.domain.tld.crt 檔案的參數憑證的新行。使用檔案的完整路徑。範例：cert /home/user/.cert/client1.domain.tld.crt
- 憑證：金鑰：新增一個指向 client1.domain.tld.key 檔案的參數金鑰的新行。使用檔案的完整路徑。範例：key /home/user/.cert/client1.domain.tld.key

使用下列命令建立 VPN 連接：sudo openvpn --config downloaded-client-config.ovpn

撤銷存取

如果您需要讓來自特定客戶端金鑰的存取失效，則需要在 CA 中撤銷該金鑰。然後將撤銷清單提交到 AWS Client VPN。

用 easy-rsa 撤銷密鑰：

- cd easy-rsa
- ./easyrsa3/easyrsa revoke client1.domain.tld
- 輸入「是」以繼續，或輸入任何其他輸入以中止。

  Continue with revocation: `yes` ... * `./easyrsa3/easyrsa gen-crl
- 已建立更新的 CRL。CRL 檔案：/home/user/easy-rsa/pki/crl.pem

將撤銷清單匯入 AWS Client VPN：

- 在 AWS Management Console 上，選取 Services (服務)，然後選取 VPC (VPC)。
- 選取 Client VPN Endpoints (用戶端 VPN 端點)。
- 選取用戶端 VPN 端點，然後選取 Actions (動作) -> Import Client Certificate CRL (匯入用戶端憑證 CRL)。
- 貼上 crl.pem 檔案的內容：

使用 AWS CLI

執行以下 命令：

```
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
```

輸出範例：

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

尋找連線端點

您的應用程式會使用端點連線到叢集。端點是叢集的唯一位址。使用叢集叢集端點所有操作。

以下各節將引導您探索所需的端點。

尋找記憶體 DB 叢集的端點 (AWS Management Console)

尋找記憶體 DB 叢集的端點

1. 登入AWS Management Console並打開 Redis 控制台的內存數據庫https://console.aws.amazon.com/memorydb/。
2. 從導覽窗格中選擇 Clusters (叢集)。

   隨即出現叢集畫面與叢集清單。選擇您要連線的叢集。
3. 若要尋找叢集的端點，請選擇叢集的名稱 (而非選項按鈕)。
4. 所以此叢集端點顯示於叢集詳細資訊。若要複製它，請選擇複製端點左側的圖示。

尋找記憶體 DB 叢集的端點 (AWSCLI)

您可以使用`describe-clusters`探索叢集的端點。命令會傳回叢集的端點。

下列作業會擷取端點，在此範例中會以*##*，針對叢集`mycluster`。

它會傳回下列 JSON 回應：

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

針對 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
        }
    ]
}
```

如需詳細資訊，請參閱「」describe-clusters。

尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

您可以使用 Redis API 的記憶體資料庫來探索叢集的端點。

尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

您可以使用 MemoryDB API 來探索叢集的端點`DescribeClusters`動作。動作會傳回叢集的端點。

下列操作會擷取叢集的端點`mycluster`。

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

如需詳細資訊，請參閱「」DescribeClusters。

使用碎片

碎片是一到 6 個節點的集合。您可以建立具有較多碎片數量和較少複本數目的叢集，每個叢集最多可達 500 個節點。此叢集組態的範圍可以從 500 個碎片和 0 個複本到 100 個碎片和 4 個複本，這是允許的複本最大數量。叢集的資料會分割到叢集的各個碎片中。如果一個碎片中有超過一個節點，碎片會實作複寫，其中一個節點為讀取/寫入主要節點，其他節點則為僅供讀取複本節點。

使用叢集時AWS Management Console，您需指定叢集中的碎片數量，以及碎片中的節點數。如需詳細資訊，請參閱建立記憶體資料庫叢集。

碎片中的每個節點都具有相同的運算、儲存體及記憶體規格。MemoryDB API 可讓您控制叢集中的屬性，例如節點數、安全設定，以及系統維護時間。

如需詳細資訊，請參閱 MemoryDB 的離線重新分片和碎片重新平衡功能 及 MemoryDB 的線上重新分片和碎片重新平衡功能。

尋找碎片的名字

您可以使用AWS Management Console、AWS CLI或 MemoryDB API 尋找碎片的名稱。

使用 AWS Management Console

下列程序會使用AWS Management Console來尋找 MemoryDB 叢集的碎片名稱。

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中選擇叢集。
3. 在「名稱」下選擇要查找其碎片名稱的集群。
4. 在「碎片和節點」選項卡下，查看「名稱」下的碎片列表。您還可以展開每個節點以查看其節點的詳細信息。

使用 AWS CLI

若要尋找 MemoryDB 叢集的碎片 (碎片) 名稱，請使用具有下列選用參數`describe-clusters`的AWS CLI作業。

- **`--cluster-name`**使用時會將輸出限制為指定叢集的詳細資訊。如果省略此參數，系統會傳回最多 100 個叢集的詳細資訊。
- **`--show-shard-details`**傳回碎片的詳細資訊，包括其名稱。

此命令將傳回 `my-cluster` 的詳細資訊。

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

它會傳回下列 JSON 回應：

加上分行符號的用意是便於閱讀。

```
{
    "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
        }
    ]
}
```

使用 API

若要尋找 MemoryDB 叢集的碎片 ID，請使用`DescribeClusters`具有下列選用參數的 API 作業。

- **`ClusterName`**使用時會將輸出限制為指定叢集的詳細資訊。如果省略此參數，系統會傳回最多 100 個叢集的詳細資訊。
- **`ShowShardDetails`**傳回碎片的詳細資訊，包括其名稱。

Example

此命令將傳回 `my-cluster` 的詳細資訊。

若為 Linux、macOS 或 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>
```

管理您的記憶體資料庫實作

在本節中，您可以找到有關如何管理 MemoryDB 實現的各種組件的詳細信息。

主題

- 雷迪斯引擎版本
- JSON 入門
- 標記您的內存 DB 資源
- 管理維護作業
- 最佳實務
- 了解記憶體資料庫複寫
- 快照和還原
- 擴展
- 使用參數群組設定引擎參數
- 教學課程：設定 Lambda 函數以存取 Amazon VPC 中的記憶體資料庫

雷迪斯引擎版本

本節涵蓋支援的 Redis 引擎版本。

主題

- 適用於 Redis 7.1 版的記憶體資料庫 (增強)
- 適用於 Redis 7.0 版的記憶體資料庫 (增強版)
- 適用於 Redis 6.2 版本的內存數據庫（增強）
- 升級引擎版本

適用於 Redis 7.1 版的記憶體資料庫 (增強)

適用於 Redis 7.1 版的 MemoryDB 在特定區域的預覽中新增了對向量搜尋功能的支援，以及重大錯誤修正和效能增強功能。

- 矢量搜索功能：矢量搜索可以與現有的 MemoryDB 功能一起使用。不使用向量搜尋的應用程式不會受到其存在的影響。下列區域提供 MemoryDB 7.1 版以上 Redis 的向量搜尋預覽：美國東部 (維吉尼

亞北部和俄亥俄州)、美國西部 (奧勒岡)、歐洲 (愛爾蘭) 和亞太區域 (東京)。有關如何啟用向量搜尋預覽和相關功能，請參閱此處的文件。

> **Note**
>
> 記憶體資料庫的 Redis 7.1 版與 OSS Redis 的版本 7.0 相容。如需有關 Redis 7.0 發行版本的詳細資訊，請參閱上的 Redis 7.0 版本說明。 GitHub

適用於 Redis 7.0 版的記憶體資料庫 (增強版)

適用於 Redis 7.0 的記憶體資料庫增加了許多改進和新功能的支援：

- Redis 函數：Redis 7 的 MemoryDB 增加了對 Redis 函數的支持，並提供了一個託管的體驗，使開發人員能夠使用存儲在 MemoryDB 集群上的應用程序邏輯執行 LUA 腳本，而不需要客戶端重新發送腳本到服務器與每個連接。
- ACL 改進：Redis 7 的內存數據庫添加了對 Redis 的訪問控制列表（ACL）的下一個版本的支持。使用 Redis 7 的 MemoryDB，客戶端現在可以在 Redis 中的特定鍵或密鑰空間上指定多組權限。
- 分片發布/訂閱：適用於 Redis 7 的 MemoryDB 增加了在啟用集群模式（CME）下運行 MemoryDB 時以分片方式運行 Redis 發布/訂閱功能的支持。Redis 發佈/訂閱功能讓發佈者能夠向頻道中任意數量的訂閱者傳送訊息。使用適用於 Redis 7 的 Amazon MemoryDB，通道會繫結至 MemoryDB 叢集中的碎片，因此無需在碎片之間傳播通道資訊。這會導致改進的可擴展性。
- 增強型 I/O 多工處理：適用於 Redis 7 版的 MemoryDB 引入了增強的 I/O 多工處理功能，對於具有許多並行用戶端連線至 MemoryDB 叢集的高輸送量工作負載，可提供更高的輸送量並減少延遲。例如，當使用 r6g.4xlarge 節點叢集並執行 5200 個並行用戶端時，與 Redis 第 6 版的 MemoryDB 相比，您可以提高多達 46% 的輸送量 (每秒讀取和寫入作業)，而 P99 延遲最多可減少 21%。

如需有關 Redis 7.0 發行版本的詳細資訊，請參閱上的 Redis 7.0 版本說明。 GitHub

適用於 Redis 6.2 版本的內存數據庫（增強）

MemoryDB 引入了 Redis 引擎的下一個版本，其中包括自動版本升級支持使用存取控制清單 (ACL) 驗證使用者，客戶端緩存和顯著的操作改進。

Redis 引擎版本 6.2.6 也引入了對原生 JavaScript 物件表示法 (JSON) 格式的支援，這是一種簡單、無結構描述的方式來編碼 Redis 叢集內的複雜資料集。有了 JSON 支援，您可以針對透過 JSON 運作的應用程式，運用效能和 Redis API。如需詳細資訊，請參閱 JSON 入門。此外，還包含 JSON 相關度

量JsonBasedCmds，用於監視此資 CloudWatch 料類型的使用情況。如需詳細資訊，請參閱 記憶體資料庫的度量。

使用 Redis 6 時，MemoryDB 將為每個 Redis OSS 次要版本提供單一版本，而不是提供多個修補程式版本。這是為了最大限度地減少必須從多個次要版本中進行選擇時的混淆和歧義。MemoryDB 還將自動管理正在運行的集群的次要版本和修補程序版本，以確保提高性能和增強的安全性。這將透過標準的客戶通知管道，透過服務更新行銷活動處理。如需詳細資訊，請參閱 Redis 內存數據庫中的服務更新。

如果您在創建過程中未指定引擎版本，MemoryDB 將自動為您選擇首選的 Redis 版本。另一方面，如果您使用指定引擎版本6.2，MemoryDB 將自動調用可用的 Redis 6.2 的首選修補程序版本。

例如，當您建立叢集時，可將--engine-version參數設定為6.2。叢集會在建立時以目前可用的偏好修補程式版本啟動。任何具有完整引擎版本值的請求都將被拒絕，將拋出異常，並且進程將失敗。

調用 DescribeEngineVersions API 時，EngineVersion參數值將設置為 6.2，並在EnginePatchVersion字段中返回實際的完整引擎版本。

如需有關 Redis 6.2 發行版本的詳細資訊，請參閱上的 Redis 6.2 版本說明。 GitHub

升級引擎版本

MemoryDB 預設會透過服務更新自動管理執行中叢集的修補程式版本。如果將叢集的AutoMinorVersionUpgrade內容設定為 false，您也可以選擇退出 auto 次要版本升級。不過，您無法選擇退出自 auto 修補程式版本升級。

您可以控制在 auto 升級開始之前，是否將支援叢集的通訊協定相容軟體升級為 MemoryDB 支援的新版本，以及何時升級。這一層控制可讓您維持特定版本的相容性、在部署至生產環境前先利用您的應用程式測試新版本，並根據自己的期限和時間表執行版本升級。

您可以透過下列方式啟動叢集的引擎版本升級：

- 通過更新它並指定新的引擎版本。如需詳細資訊，請參閱 修改記憶體資料庫叢集。
- 套用對應引擎版本的服務更新。如需詳細資訊，請參閱 Redis 內存數據庫中的服務更新。

注意下列事項：

- 您可以升級到更新版本的引擎，但無法降級到舊版引擎。如果您要使用舊版引擎，您必須刪除現有的叢集，並使用舊版引擎重新建立一個。

- 建議您定期升級至最新主要版本，因為大部分主要改進功能不會向後移植至舊版。隨著 MemoryDB 將可用性擴展到新的 AWS 區域，MemoryDB 支持當時的兩個最新`MAJOR.MINOR`版本的新區域。例如，如果一個新的 AWS 區域啟動，而 Redis 版本的最新`MAJOR.MINOR`記憶體資料庫為 7.0 和 6.2，Redis 的記憶體資料庫將支援新區域中的 7.0 和 6.2 版本。 AWS 隨著 Redis 的較新`MAJOR.MINOR`版本的內存數據庫被釋放，內存數據庫將繼續添加對 Redis 版本新發布的內存數據庫的支持。若要進一步瞭解如何選擇 MemoryDB 區域，請參閱。支援的地區和端點
- 引擎版本管理功能是為了讓您能夠盡可能控制執行修補的方式，但是，MemoryDB 保留在系統或軟件出現嚴重安全漏洞的情況下代表您修補叢集的權利。
- MemoryDB 將為每個 Redis OSS 次要版本提供單一版本，而不是提供多個修補程式版本。這是為了最大限度地減少必須從多個版本中進行選擇時的混淆和歧義。MemoryDB 還將自動管理正在運行的集群的次要版本和修補程序版本，以確保提高性能和增強的安全性。這將透過標準的客戶通知管道，透過服務更新行銷活動處理。如需詳細資訊，請參閱 Redis 內存數據庫中的服務更新。
- 您可以在最短的停機時間內升級叢集版本。叢集在整個升級過程中都可供讀取，而在過程的多數時間也可供寫入，除了在容錯移轉操作中會有幾秒可能無法寫入。
- 建議您在低傳入寫入流量期間執行引擎升級。

  具有多個碎片的簇被處理和修補，如下所示：

  - 每個碎片隨時只會執行一個升級作業。
  - 在每個碎片中，都會先處理所有複本，再處理主要複本。如果某個碎片中的複本較少，則該碎片中的主要複本可能會在其他碎片的複本處理完成前就已處理。
  - 跨所有碎片時，則會循序處理主要節點。一次只會升級一個主要節點。

主題

- 如何升級引擎版本
- 解決 Redis 引擎升級被封鎖的問題

如何升級引擎版本

您可以使用 MemoryDB 主控台、或 MemoryDB API 修改叢集，並指定較新的引擎版本 AWS CLI，以啟動叢集的版本升級。如需詳細資訊，請參閱下列主題。

- 使用 AWS Management Console
- 使用 AWS CLI
- 使用記憶體資料庫 API

解決 Redis 引擎升級被封鎖的問題

如下表所示，如果您的 Redis 向上擴展操作為擱置中，即表示 Redis 引擎升級操作遭到封鎖。

待定作業	封鎖的作業
向上擴展	立即升級引擎
引擎升級	立即向上擴展
向上擴展與升級引擎	立即向上擴展
	立即升級引擎

JSON 入門

MemoryDB 支援原生JavaScript物件標記法 (JSON) 格式，這是一種簡單、無結構描述的方式，對 Redis 叢集內的複雜資料集進行編碼。您可以在 Redis 叢集內，使用JavaScript物件標記法 (JSON) 格式，原生儲存和存取資料，並更新儲存在這些叢集中的 JSON 資料，不需管理自訂程式碼來序列化和還原序列化。

除了針對透過 JSON 操作的應用程式利用 Redis API 之外，您現在可以有效率地擷取和更新 JSON 文件的特定部分，而不需要操作整個物件，進而改善效能並降低成本。您也可以使用 Goessner 式 `JSONPath` 查詢，搜尋 JSON 文件內容。

使用支援的引擎版本建立叢集之後，JSON 資料類型和相關聯的命令會自動可用。這是 API 相容且 RGB 相容於 Redisjson 模組的第 2 版，因此您可以輕鬆地將現有的 JSON 型 Redis 應用程式遷移到記憶體資料庫中。如需支援的 Redis 命令詳細資訊，請參閱支援的命令。

JSON 相關度量`JsonBasedCmds`已納入CloudWatch以監視此資料類型的使用情況。如需詳細資訊，請參閱 MemoryDB 指標。

> **Note**
>
> 若要使用 JSON，您必須執行 Redis 引擎 6.2.6 或更新版本。

主題

- JSON 數據類型概述

- 支援的命令

JSON 數據類型概述

內存數據庫支持一些 Redis 的命令與 JSON 數據類型的工作。以下是 JSON 數據類型的概述和所支持 Redis 命令的詳細列表。

術語

術語	描述
JSON 文件	指的是一個紅色的 JSON 密鑰的值
JSON 值	指的是 JSON 文檔的子集，包括代表整個文檔的根。值可以是容器或容器中的條目
JSON 元素	相當於 JSON 值

支援的 JSON 標準

JSON 格式符合 RFC 7159 和 ECMA-404 JSON 資料交換標準。支援 JSON 文字中的 UTF-8 Unicode。

根元素

根元素可為任何 JSON 資料類型。請注意，在舊版 RFC 4627 中，只允許將物件或陣列當作根值。由於更新至 RFC 7159，JSON 文件的根可為任何 JSON 資料類型。

文件大小限制

JSON 文件以針對快速存取和修改進行最佳化的格式儲存在內部。這種格式通常會導致消耗更多的記憶體，而不是相同文件的對等序列化表示。單一 JSON 文件的記憶體使用量限制為 64MB，這是記憶體內資料結構的大小，而不是 JSON 字串。使用`JSON.DEBUG MEMORY`命令可以檢查 JSON 文檔消耗的內存量。

JSON ACL

- JSON 資料類型已完全整合至 Redis 的存取控制清單 (ACL) 功能中。與現有的每個資料類型類別 (@string、@hash 等) 類似，會新增一個新類別 @json，以簡化對 JSON 命令和資料的存取管理作

業。沒有其他現有 Redis 命令屬於 @json 類別。所有 JSON 命令都會強制執行任何索引鍵空間或命令限制和許可。

- 已更新五個現有 Redis ACL 類別，加入新的 JSON 命令：@read、@write、@fast、@slow 和 @admin。下表指出 JSON 命令到適當的類別的映射。

ACL

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

巢狀深度限制

JSON 物件或陣列具有本身是另一個 JSON 物件或陣列的元素時，該內部物件或陣列稱之為在外部物件或陣列中「巢狀」。巢狀深度上限為 128。任何建立巢狀深度大於 128 文件的嘗試，都會遭到拒絕，並顯示錯誤。

命令語法

大多數命令都需有 Redis 索引鍵名稱當作第一個引數。部分命令也有路徑引數。如果路徑引數為可選，未提供，路徑引數預設為根。

標記法：

- 必要的引數用尖括號括起來，例如 <key>
- 可選參數用方括號括起來，例如 [path]
- 其他可選參數由... 表示，例如 [json...]

路徑語法

JSON-Redis 支持兩種路徑語法：

- 增強的語法 — 遵循戈斯納描述的 JSONPath 語法，如下表所示。為清楚說明，我們重新排序並修改表格中的描述。
- 受限語法 – 查詢功能有限。

> **Note**
>
> 某些命令的結果是敏感的，使用哪種類型的路徑語法。

如果查詢路徑以 '$' 開頭，它會使用增強型語法。否則，將使用受限語法。

增強的語法

符號/表達式	描述
$	根元素

符號/表達式	描述
. 或 []	子操作員
..	遞歸下降
*	萬用字元。物件或陣列中的所有元素。
[]	數組下標運算符。索引以 0 為基礎。
[,]	聯合運營商
[start:end:step]	陣列切片運算符
?()	將濾鏡 (Script) 運算式套用至目前的陣列或物件
()	篩選條件 expression
@	用於參照正在處理的當前節點的過濾器表達式
==	等於，用於篩選器運算式。
!=	不等於，用於篩選器運算式。
>	大於，用於篩選器運算式。
>=	大於或等於，用於篩選條件表達式。
<	小於，用於篩選器運算式。
<=	小於或等於，用於篩選條件表達式。
&&	邏輯 AND，用於組合多個過濾器表達式。
||	邏輯 OR，用於組合多個過濾器表達式。

範例

下面的例子是建立在 Goessner 的示例 XML 數據，我們通過添加其他字段進行了修改。

```
{ "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
    }
  }
}
```

路徑	描述
$.store.book[*].author	商店中所有書籍的作者
$..author	所有作者
$.store.*	商店的所有成員
$["store"].*	商店的所有成員
$.store..price	商店裡所有東西的價格
$..*	JSON 結構的所有遞迴成員
$..book[*]	所有書籍
$..book[0]	第一本書
$..book[-1]	最後一本書
$..book[0:2]	前兩本書
$..book[0,1]	前兩本書
$..book[0:4]	從索引 0 到 3 的書籍（結束索引不包括在內）
$..book[0:4:2]	索引 0，2 處的書籍
$..book[?(@.isbn)]	所有具有 ISBN 編號的書籍
$..book[?(@.price<10)]	所有書籍都比 $10 便宜
'$..book[?(@.price < 10)]'	所有書籍都比 10 美元便宜。（如果路徑包含空格，則必須引用該路徑）
'$..book[?(@["price"] < 10)]'	所有書籍都比 $10 便宜
'$..book[?(@.["price"] < 10)]'	所有書籍都比 $10 便宜
$..book[?(@.price>=10&&@.price<=100)]	所有書籍在 10 美元至 100 美元的價格範圍內，包括在內

路徑	描述
'$..book[?(@.price>=10 && @.price<=100)]'	所有書籍在 10 美元至 100 美元的價格範圍內，包括在內。（如果路徑包含空格，則必須引用該路徑）
$..book[?(@.sold==true||@.in-stock==false)]	所有書籍已售出或缺貨
'$..book[?(@.sold == true || @.in-stock == false)]'	所有書籍已售出或缺貨。（如果路徑包含空格，則必須引用該路徑）
'$.store.book[?(@.["category"] == "fiction")]'	小說類別中的所有書籍
'$.store.book[?(@.["category"] != "fiction")]'	非小說類別的所有書籍

更多過濾器表達式示例：

```
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]"
```

受限語法

符號/表達式	描述
. 或 []	子操作員
[]	數組下標運算符。索引以 0 為基礎。

範例

路徑	描述
.store.book[0].author	第一本書的作者
.store.book[-1].author	最後一本書的作者
.address.city	城市名稱
["store"]["book"][0]["title"]	第一本書的標題
["store"]["book"][-1]["title"]	最後一本書的標題

> **Note**
>
> 本文件中引用的所有 Goessner 內容均受創用 CC 授權規範。

常見錯誤字首

每個錯誤訊息都有一個字首。以下是常用錯誤前置詞的清單：

字首	描述
ERR	一般錯誤
LIMIT	超出大小限制錯誤。例如，超出文件大小限制或巢狀深度限制
NONEXISTENT	鍵或路徑不存在
OUTOFBOUNDARIES	數組索引超出界限
SYNTAXERR	語法錯誤
WRONGTYPE	錯誤的值類型

JSON 相關指標

提供下列 JSON 資訊指標：

Info	描述
json_total_memory_bytes	分配給 JSON 物件的總記憶體
json_num_documents	Redis 中的文件總數

要查詢核心指標，請運行 Redis 命令：

```
info json_core_metrics
```

內存數據庫如何與 JSON 交互

下面說明了記憶數據庫如何與 JSON 數據類型進行交互。

運算子優先順序

評估用於篩選的條件表達式時，&& 優先，然後評估 ||，就像大多數語言一樣。括號內的操作將首先執行。

路徑巢狀上限行為

記憶體 DB 的最大路徑巢狀限制為 128。`$.a.b.c.d...` 等值只能達到 128 個等級。

處理數值

JSON 對整數和浮點數沒有單獨的數據類型。均稱為數字。

當接收到 JSON 號碼時，它會以兩種格式之一存儲。如果該數字符合 64 位有符號整數，則將其轉換為該格式；否則，它將被存儲為字符串。對兩個 JSON 數字（例如 JSON.NUMINCRBY 和 JSON.NUMMULTBY）進行的算術運算嘗試保持盡可能多的精度。如果兩個操作數和結果值適合 64 位有符號整數，則執行整數算術。否則，輸入操作數被轉換為 64 位 IEEE 雙精度浮點數，執行算術運算，並將結果轉換回字符串。

算術命令 `NUMINCRBY` 和 `NUMMULTBY`：

- 如果兩個數字都是整數，並且結果超出 int64 的範圍，它將自動成為雙精度浮點數。
- 如果至少有一個數字是浮點數，則結果將是一個雙精度浮點數。
- 如果結果超出 double 的範圍，該命令將返回一個`OVERFLOW`錯誤。

> **Note**
>
> 在 Redis 引擎版本 6.2.6.R2 之前，當輸入時接收到 JSON 數字時，它會轉換為兩個內部二進制表示之一：64 位有符號整數或 64 位 IEEE 雙精度浮點。不會保留原始字串和全部格式化。因此，數字當作 JSON 回應的一部分輸出時，會從內部二進位表示法，轉換為使用一般格式化規則的可列印字串。這些規則可能導致產生的字串與接收的字串不同。
>
> - 如果兩個數字都是整數，並且結果超出範圍的範圍`int64`，會自動變成 64 位元 IEEE 雙精確度浮點數。
> - 如果其中至少一個數字是浮點數，結果會是 64 位元 IEEE 雙精確度浮點數。
> - 如果結果超出 64 位元 IEEE 雙精確度的範圍，會發生 OVERFLOW 錯`OVERFLOW`誤。

如需可用命令的詳細清單，請參閱支援的命令。

嚴格語法評估

即使路徑的子集包含有效路徑，MemoryDB 也不允許使用無效語法的 JSON 路徑。這是為了我們的客戶保持正確行為。

支援的命令

支援下列 JSON 命令：

主題

- 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

追加一個或多個值到路徑中的陣列值。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- json（必填）-JSON 值被追加到數組

傳回

如果路徑是增強型語法：

- 整數陣列，代表陣列上每個路徑上的新長度。
- 如果值不是陣列，其相應的傳回值為 null。
- 如果輸入 json 引數之一不是有效的 JSON 字串，會發生 SYNTAXERR 錯誤。
- 如果沒有路徑，會發生 NONEXISTENT 錯誤。

如果路徑是受限語法：

- 整數，新陣列長度。
- 如果選取多個陣列值，命令會傳回上次更新陣列的新長度。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。
- 如果輸入 json 引數之一不是有效的 JSON 字串，會發生 SYNTAXERR 錯誤。
- 如果沒有路徑，會發生 NONEXISTENT 錯誤。

範例

增強型路徑語法：

```
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\"]]"
```

受限路徑語法：

```
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

搜尋路徑中陣列中第一次出現的純量 JSON 值。

- 將索引四捨五入到陣列的開頭和結尾，處理超出範圍的錯誤。
- 如果開頭 > 結尾，傳回 -1 (找不到)。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- JSON 標量（必填）-要搜索的標量值; JSON 標量指的是不是對象或數組的值。即字符串，數字，布爾值和 null 是標量值。
- 啟動（可選）-開始索引，包括在內。如果未提供，預設為 0。
- end（可選）-結束索引，排他性。如果未提供，預設為 0，表示包含最後一個元素。0 或 -1 表示包含最後一個元素。

傳回

如果路徑是增強型語法：

- 整數陣列。每個值都是路徑的陣列中相符元素的索引。如果找不到，則值為 -1。
- 如果值不是陣列，其相應的傳回值為 null。

如果路徑是受限語法：

- 如果找不到，則為整數、相符元素的索引或 -1。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。

範例

增強型路徑語法：

```
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
```

受限路徑語法：

```
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

插入一個或多個值到索引之前的路徑的數組值。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- 索引（必填）-數組索引，在其之前插入值。
- json（必填）-JSON 值被追加到數組

傳回

如果路徑是增強型語法：

- 整數陣列，代表陣列上每個路徑上的新長度。
- 如果值是空陣列，其相應的傳回值為 null。
- 如果值不是陣列，其相應的傳回值為 null。
- 如果索引引數超出範圍，會發生 OUTOFBOUNDARIES 錯誤。

如果路徑是受限語法：

- 整數，新陣列長度。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。
- 如果索引引數超出範圍，會發生 OUTOFBOUNDARIES 錯誤。

範例

增強型路徑語法：

```
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\"]]"
```

受限路徑語法：

```
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

獲取在路徑的數組值的長度。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 整數陣列，代表每個路徑的陣列長度。
- 如果值不是陣列，其相應的傳回值為 null。
- 如果沒有文件索引鍵，則為 null。

如果路徑是受限語法：

- 大量字串陣列。每個元素都是物件中的索引鍵名稱。
- 整數，陣列長度。
- 如果選取多個物件，命令會傳回第一個陣列的長度。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。
- 如果沒有路徑，會發生 WRONGTYPE 錯誤。
- 如果沒有文件索引鍵，則為 null。

範例

增強型路徑語法：

```
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)
```

受限路徑語法：

```
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

從數組中刪除並返回索引處的元素。彈出空陣列會傳回 null。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根
- index（可選）-數組中的位置開始彈出。
  - 如果未提供，預設為 -1，表示最後一個元素。
  - 負值表示從最後一個元素數起的位置。
  - 超出範圍的索引會四捨五入到各自的陣列範圍。

傳回

如果路徑是增強型語法：

- 批量字符串數組，代表每個路徑上的彈出值。
- 如果值是空陣列，其相應的傳回值為 null。
- 如果值不是陣列，其相應的傳回值為 null。

如果路徑是受限語法：

- 批量字符串，代表彈出的 JSON 值
- 如果陣列是空的，則為 null。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。

範例

增強型路徑語法：

```
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\"]]"
```

受限路徑語法：

```
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

在路徑修剪數組，使其成為子數組 [開始，結束]，兩者都包括在內。

- 如果陣列是空的，不必做任何事，會傳回 0。
- 如果開頭 <0，則將其視為 0。
- 如果結尾 >= 大小 (陣列的大小)，則將其視為 size-1。
- 如果開頭 >= 大小或開頭 > 結尾，清空陣列並傳回 0。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- 啟動（必填）-開始索引，包括在內。
- end（必填）-結束索引，包括在內。

傳回

如果路徑是增強型語法：

- 整數陣列，代表陣列上每個路徑上的新長度。

- 如果值是空陣列，其相應的傳回值為 null。
- 如果值不是陣列，其相應的傳回值為 null。
- 如果索引引數超出範圍，會發生 OUTOFBOUNDARIES 錯誤。

如果路徑是受限語法：

- 整數，新陣列長度。
- 如果陣列是空的，則為 null。
- 如果路徑上的值不是陣列，會發生 WRONGTYPE 錯誤。
- 如果索引引數超出範圍，會發生 OUTOFBOUNDARIES 錯誤。

範例

增強型路徑語法：

```
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\"]]"
```

受限路徑語法：

```
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

清除陣列或路徑上的物件。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

- 整數，已清除的容器數目。
- 清除空陣列或物件計為清除 0 個容器。

> **Note**
>
> Piror 6.6.R2 版，清除一個空陣列或物件計為清除 1 個容器。

- 清除非容器值會傳回 0。
- 如果路徑中找不到任何陣列或物件值，則命令會傳回 0。

範例

```
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

報告信息。支援的子命令如下：

- 記憶體 <key>[路徑] — 以 JSON 值的位元組報告記憶體使用情況。如果未提供，路徑預設為根。
- 深度 <key>[路徑] — 報告 JSON 文件的最大路徑深度。

> **Note**
>
> 這個子命令只能使用 Redis 引擎 6.2.6.R2 或更新版本。

- 欄位 <key>[路徑] — 報告指定文件路徑上的欄位數目。如果未提供，路徑預設為根。每個非容器 JSON 值都計為一個欄位。物件和陣列遞迴計為每個內含 JSON 值的一個欄位。除根容器外，每個容器值都計為一個附加欄位。
- 幫助-打印命令的幫助消息。

語法

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

取決於子命令：

MEMORY

- 如果路徑是增強型語法：
  - 返回一個整數數組，代表每個路徑中 JSON 值的內存大小（以字節為單位）。
  - 如果沒有 Redis 索引鍵，會發生 WRONGTYPE 陣列。
- 如果路徑是受限語法：
  - 返回一個整數，內存大小以字節為單位的 JSON 值。
  - 如果沒有 Redis 索引鍵，則返回 null。

DEPTH

- 傳回整數，代表 JSON 文件的最大路徑深度。
- 如果沒有 Redis 索引鍵，傳回 null。

FIELDS

- 如果路徑是增強型語法：
  - 返回一個整數數組，代表在每個路徑 JSON 值的字段的數量。

  - 如果沒有 Redis 索引鍵，會發生 WRONGTYPE 陣列。
- 如果路徑是受限語法：
  - 返回一個整數，JSON 值的字段數。
  - 如果沒有 Redis 索引鍵，則返回 null。

幫助-返回幫助消息的數組。

範例

增強型路徑語法：

```
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
```

受限路徑語法：

```
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

刪除文件金鑰中路徑上的 JSON 值。如果路徑是根，相當於從 Redis 刪除索引鍵。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

- 刪除的元素數目。
- 如果沒有 Redis 索引鍵，則為 0。
- 如果 JSON 路徑無效或不存在，則為 0。

範例

增強型路徑語法：

```
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\":[]}"
```

受限路徑語法：

```
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

的別名 JSON.DEL

JSON.GET

在一個或多個路徑返回序列化的 JSON。

語法

```
JSON.GET <key>
```

```
[INDENT indentation-string]
[NEWLINE newline-string]
[SPACE space-string]
[NOESCAPE]
[path ...]
```

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 縮進/換行符/空格（可選）-控制返回的 JSON 字符串的格式，即「漂亮的打印」。每個字符串的默認值是空字符串。他們可以在任何組合被壓制。可以按任何順序指定。
- NOESCAPE-可選的，允許出現舊版兼容性，並且沒有其他效果。
- path（可選）— 零個或多個 JSON 路徑，如果沒有給出任何路徑，則默認為根路徑。路徑引數必須放在最後。

傳回

增強型路徑語法：

如果提供一個路徑：

- 返回值數組的序列化字符串。
- 如果未選取任何值，此命令會傳回空陣列。

如果提供多個路徑：

- 返回一個字符串化的 JSON 對象，其中每個路徑都是一個關鍵字。
- 如果混合增強型和受限路徑語法，結果會按照增強型語法。
- 如果沒有路徑，則其對應的值會是空陣列。

範例

增強型路徑語法：

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

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

受限路徑語法：

```
 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

從多個文檔密鑰在路徑上獲取序列化的 JSONS。對於不存在的密鑰或 JSON 路徑，則返回 null。

語法

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

- key (必要) – 文件類型的一或多個 Redis 索引鍵。
- 路徑 (必要) — JSON 路徑

傳回

- 批量字符串數組。陣列的大小等於命令中的索引鍵數量。陣列的每個元素都會填入 (a) 位於路徑的序列化 JSON，或者 (b) 如果鍵不存在或路徑不存在於文檔中或路徑無效（語法錯誤），則為 Null。
- 如果有任何指定的索引鍵，且不是 JSON 索引鍵，命令會傳回 WRONGTYPE 錯誤。

範例

增強型路徑語法：

```
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\"]"
```

受限路徑語法：

```
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

以指定數字遞增路徑上的數字值。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- 數字（必填）-一個數字

傳回

如果路徑是增強型語法：

- 代表每個路徑結果值的批量字符串數組。
- 如果值不是數字，則其對應的傳回值為 null。
- 如果無法剖析數字，會發生 WRONGTYPE 錯誤。
- 如果結果超出 64 位元 IEEE 雙精確度的範圍，會發生 OVERFLOW 錯誤。
- 如果沒有文件索引鍵，則為 NONEXISTENT。

如果路徑是受限語法：

- 代表結果值的批量字符串。
- 如果選取多個值，命令會傳回上次所更新值的結果。
- 如果路徑上的值不是數字，會發生 WRONGTYPE 錯誤。
- 如果無法剖析數字，會發生 WRONGTYPE 錯誤。
- 如果結果超出 64 位元 IEEE 雙精確度的範圍，會發生 OVERFLOW 錯誤。
- 如果沒有文件索引鍵，則為 NONEXISTENT。

範例

增強型路徑語法：

```
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}}"
```

受限路徑語法：

```
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

將路徑上的數字值乘以給定的數字。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑
- 數字 (必填) -一個數字

傳回

如果路徑是增強型語法：

- 代表每個路徑結果值的批量字符串數組。
- 如果值不是數字，則其對應的傳回值為 null。
- 如果無法剖析數字，會發生 WRONGTYPE 錯誤。

- 如果結果超出 64 位元 IEEE 雙精確度的範圍，會發生 OVERFLOW 錯誤。
- 如果沒有文件索引鍵，則為 NONEXISTENT。

如果路徑是受限語法：

- 代表結果值的批量字符串。
- 如果選取多個值，命令會傳回上次所更新值的結果。
- 如果路徑上的值不是數字，會發生 WRONGTYPE 錯誤。
- 如果無法剖析數字，會發生 WRONGTYPE 錯誤。
- 如果結果超出 64 位元 IEEE 雙精確度的範圍，會發生 OVERFLOW 錯誤。
- 如果沒有文件索引鍵，則為 NONEXISTENT。

範例

增強型路徑語法：

```
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]"
```

受限路徑語法：

```
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

獲取路徑上對象值的索引鍵數目。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 整數陣列，代表每個路徑的物件長度。
- 如果值不是物件，其相應的傳回值為 null。
- 如果沒有文件索引鍵，則為 null。

如果路徑是受限語法：

- 整數，物件中的索引鍵數目。
- 如果選取多個物件，命令會傳回第一個物件的長度。
- 如果路徑上的值不是物件，會發生 WRONGTYPE 錯誤。
- 如果沒有路徑，會發生 WRONGTYPE 錯誤。
- 如果沒有文件索引鍵，則為 null。

範例

增強型路徑語法：

```
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)
```

受限路徑語法：

```
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

在路徑的對象值中獲取鍵名稱。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 大量字串陣列。每個元素都是相符物件中的索引鍵陣列。
- 如果值不是物件，其相應的傳回值是空白值。
- 如果沒有文件索引鍵，則為 null。

如果路徑是受限語法：

- 大量字串陣列。每個元素都是物件中的索引鍵名稱。
- 如果選取多個物件，命令會傳回第一個物件的索引鍵。
- 如果路徑上的值不是物件，會發生 WRONGTYPE 錯誤。
- 如果沒有路徑，會發生 WRONGTYPE 錯誤。
- 如果沒有文件索引鍵，則為 null。

範例

增強型路徑語法：

```
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"
```

受限路徑語法：

```
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

返回在 Redis 的序列化協議（RESP）給定路徑的 JSON 值。如果值是容器，則響應為 RESP 數組或嵌套數組。

- JSON null 映射至 RESP null 大量字串。
- JSON 布爾值映射到相應的 RESP 簡單字符串。
- 整數映射至 RESP 整數。
- 64 位 IEEE 雙精確度浮點數映射至 RESP 大量字串。
- JSON 字串映射至 RESP 大量字串。
- JSON 陣列被表示為 RESP 陣列，其中第一個元素是簡單的字符串 [，其次是數組的元素。
- JSON 對象表示為 RESP 數組，其中第一個元素是簡單的字符串 {，後跟鍵-值對，每個都是一個 RESP 批量字符串。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 陣列的陣列。每個陣列元素呈現一個路徑上值的 RESP 形式。
- 如果沒有文件索引鍵，則為空陣列。

如果路徑是受限語法：

- 陣列，代表路徑上值的 RESP 形式。
- 如果沒有文件索引鍵，則為 null。

範例

增強型路徑語法：

```
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"
```

受限路徑語法：

```
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

在路徑上設定 JSON 值。

如果路徑呼叫物件成員：

- 如果父元素不存在，則該命令將返回不存在的錯誤。
- 如果父元素存在但不是對象，則該命令將返回 ERROR。
- 如果有父元素且為物件：
  - 如果沒有成員，只會在父物件是路徑中的最後一個子系時，將新成員附加至父物件。否則，該命令將返回不存在的錯誤。
  - 如果有該成員，其值將以 JSON 值取代。

如果路徑呼叫陣列索引：

- 如果父元素不存在，該命令將返回一個不存在的錯誤。
- 如果父元素存在但不是數組，則該命令將返回 ERROR。
- 如果父元素存在但索引超出邊界，則該命令將返回 OUTOFBOUNDANDS 錯誤。
- 如果有父元素且索引有效，該元素將以新的 JSON 值取代。

如果路徑呼叫物件或陣列，該值 (物件或陣列) 將以新的 JSON 值取代。

語法

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

[NX | XX] 您可以在其中包含 [NX | XX] 識別碼的 0 個或 1 個

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (必要) — JSON 路徑。針對新的 Redis 索引鍵，JSON 路徑必須為根 "."。

- NX (選用) — 如果路徑為根，請只在沒有 Redis 索引鍵時才設定值，如插入新文件。如果路徑不是根，只有在路徑不存在時才設置該值，即將值插入到文檔中。
- XX (選用) — 如果路徑為根，請只在有 Redis 索引鍵時才設定值，並在 Redis 索引鍵時才設定值，也就是說，取代現有文件。如果路徑不是根，則僅在路徑存在時才設置該值，即更新現有值。

傳回

- 成功時有簡單字串 'OK'。
- 如果不符合 NX 或 XX 條件，即為 null。

範例

增強型路徑語法：

```
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]}"
```

受限路徑語法：

```
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

追加一個字符串到路徑中的 JSON 字符串。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根
- json_string（必需）-一個字符串的 JSON 表示。請注意，JSON 字符串必須加引號，即「foo" '。

傳回

如果路徑是增強型語法：

- 整數數組，代表字符串在每個路徑的新長度。
- 如果路徑上的值不是字符串，則其對應的傳回值為 null。
- SYNTAXERR 如果輸入 json 引數不是有效 JSON 字串，會發生 WRONGTYPE 錯誤。
- NONEXISTENT 如果沒有路徑，則為錯誤。

如果路徑是受限語法：

- 整數，字串的新長度。
- 如果選取多個字串值，該命令會傳回上次所更新字串的新長度。
- 如果路徑上的值不是字串，會發生 WRONGTYPE 錯誤。
- 如果輸入 json 引數不是有效 JSON 字串，會發生 WRONGTYPE 錯誤。
- 如果沒有路徑，會發生 NONEXISTENT 錯誤。

範例

增強型路徑語法：

```
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)
```

受限路徑語法：

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

JSON.STRLEN

獲取路徑中 JSON 字符串值的長度。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 整數數組，代表字符串值在每個路徑的長度。
- 如果值不是字串，其對應的傳回值為 null。
- 如果沒有文件索引鍵，則為 null。

如果路徑是受限語法：

- 整數，字串的長度。
- 如果選取多個字串值，該命令會傳回第一個字串的長度。
- 如果路徑上的值不是字串，會發生 WRONGTYPE 錯誤。
- 如果沒有路徑，會發生 NONEXISTENT 錯誤。
- 如果沒有文件索引鍵，則為 null。

範例

增強型路徑語法：

```
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)
```

受限路徑語法：

```
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

在路徑上切換真假之間的布爾值。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 整數數組（0-假，1-真），代表在每個路徑結果的布爾值。
- 如果值不是布林值，其對應的傳回值為 null。
- 如果沒有文件索引鍵，則為 NONEXISTENT。

如果路徑是受限語法：

- 字符串（「真」/「假」）表示生成的布爾值。
- 如果沒有文件索引鍵，則為 NONEXISTENT。
- WRONGTYPE如果路徑上的值不是布林值，會發生 WRONGTYPE 錯誤。

範例

增強型路徑語法：

```
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)
```

受限路徑語法：

```
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

在給定路徑的值的報告類型。

語法

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

- key (必要) — JSON 文件類型的 Redis 索引鍵
- 路徑 (選用) — JSON 路徑。如果未提供，預設為根

傳回

如果路徑是增強型語法：

- 字符串數組，代表在每個路徑的值的類型。該類型是 {"null", "boolean", "string", "number", "integer", "object" and "array"} 之一。
- 如果沒有路徑，其對應的傳回值為 null。
- 如果沒有文件索引鍵，則為空陣列。

如果路徑是受限語法：

- 字串，值的類型
- 如果沒有文件索引鍵，則為 null。

- 如果 JSON 路徑無效或不存在，則為 null。

範例

增強型路徑語法：

```
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
```

受限路徑語法：

```
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
```

標記您的內存 DB 資源

為協助您管理叢集和其他 MemeryDB 資源，您可以用標籤形式將您自己的中繼資料指派給每個資源。標籤可讓您以不同的方式分類您的 AWS 資源，例如依據目的、擁有者或環境。當您有許多相同類型的資源時，這將會很有用，因為—您可以依據先前指派的標籤，快速識別特定的資源。本主題說明標籤並示範如何建立它們。

> **Warning**
>
> 根據最佳實務，建議您不要在標籤中包含敏感資料。

標籤基本概念

標籤是您指派給 AWS 資源的標籤。每個標籤皆包含由您定義的一個金鑰與一個選用值。標籤可讓您以不同的方式分類您的 AWS 資源，例如依據用途或擁有者。例如，您可以為帳戶的 MemeryDB 叢集定義一組標籤，協助您追蹤各個叢集的擁有者與使用者組。

我們建議您為每種資源類型建立符合您需求的標籤金鑰。使用一致的標籤金鑰組可讓您更輕鬆的管理您的資源。您可以根據您新增的標籤搜尋和篩選資源。如需如何實作有效資源標記策略的詳細資訊，請參見AWS標記最佳實務的白皮書。

標籤對 MemeryDB 來說不具任何語意意義，並會嚴格解譯為字元字串。此外，標籤不會自動指派給您的資源。您可以編輯標籤金鑰和值，並且可以隨時從資源移除標籤。您可以將標籤的值設為 `null`。若您將與現有標籤具有相同鍵的標籤新增到該資源，則新值會覆寫舊值。如果您刪除資源，也會刪除任何該資源的標籤。

您可以使用AWS Management Console，AWS CLI和內存數據庫 API。

若使用 IAM，您可以控制 AWS 帳戶中的哪些使用者具有建立、編輯和刪除標籤的許可。如需詳細資訊，請參閱 資源層級許可。

您可以標記的資源

您可以為帳戶中現有的大多數 MemoryDB 資源新增標籤。下表列出支援標籤建立的資源。若您使用 AWS Management Console，可以利用標籤編輯器來對資源套用標籤。有些資源畫面可讓您在建立資源時指定資源的標籤；例如，具有 Name 索引鍵和您指定值的標籤。在大多數的案例中，主控台會立即在建立資源後套用標籤 (而非在資源建立過程時)。控制台可以根據名稱標籤，但此標籤對 MemoryDB 服務來說不具有任何語意意義。

此外，有些資源建立動作可讓您在建立資源時指定資源的標籤。若標籤無法在資源建立時套用，我們會轉返資源建立程序。這可確保資源不是具有標籤建立，就是不會建立，因此無論何時都不會有不具有標籤的資源。藉由在建立時為資源建立標籤，您可以消除在資源建立後執行自訂標籤指令碼的必要。

如果您使用 Amazon MemoryDB API，AWSCLI 或AWS開發套件，您可以使用Tags參數來應用標籤。這些類別為：

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

下表說明可標記的內存數據庫資源，以及可在使用內存數據庫 API 建立時標記的資源。AWSCLI 或 AWSSDK。

內存 DB 資源的標記支援

支援標籤	支援在建立時標記
是	是
是	是
是	是
是	是
是	是

支援標籤	支援在建立時標記
是	是

您可以在您的 IAM 政策中將標籤式的資源層級許可套用到支援在建立時新增標籤的 MemeryDB API 動作，以實作可在建立時為資源建立標籤之使用者和叢集的細微控制。您的資源從建立時便已獲得妥善的保護，標籤會立即套用到您的資源。因此，控制資源使用情況的任何標籤式資源層級許可都會立即生效。您可以更準確的追蹤和報告您的資源。您可以強制新資源使用標籤，並控制哪些標籤金鑰和值會在您的資源上設定。

如需詳細資訊，請參閱 為資源加上標籤的範例。

如需為資源加上標籤以便計費的詳細資訊，請參閱「使用成本配置標籤監控成本」。

標記集羣和快照

以下規則適用於屬於請求作業一部分的標記程序：

- CreateCluster：
  - 如果提供 `--cluster-name`：

    如果請求中包含標籤，就會對叢集進行標籤。
  - 如果提供 `--snapshot-name`：

    如果請求中包含標籤，叢集只會使用這些標籤進行標記。如果請求中不含任何標籤，就會將快照標籤新增至叢集。
- CreateSnapshot：
  - 如果提供 `--cluster-name`：

    如果請求中包含標籤，就只會將這些請求標籤新增至快照。如果請求中不含任何標籤，就會將叢集標籤新增至快照。
  - 針對自動快照：

    標籤會從叢集標籤傳播。
- CopySnapshot：

如果請求中包含標籤，就只會將這些請求標籤新增至快照。如果請求中不含任何標籤，就會將來源快照標籤新增至複製的快照。

- TagResource和UntagResource：

  將從資源中添加/刪除標籤。

標籤限制

以下基本限制適用於標籤：

- 每一資源最多標籤數 – 50
- 對於每一個資源，每個標籤金鑰必須是唯一的，且每個標籤金鑰只能有一個值。
- 索引鍵長度上限 - 128 個 UTF-8 Unicode 字元。
- 值長度上限 - 256 個 UTF-8 Unicode 字元。
- 雖然 MemoryDB 允許標籤中的任何字元，但其他服務可能是限制性的。服務間允許的字元包括：可用 UTF-8 表示的英文字母、數字和空格，還有以下字元：+ - = . _ : / @
- 標籤金鑰與值皆區分大小寫。
- 此 `aws:` 字首已保留供 AWS 使用。如果標籤具有此字首的標籤金鑰，則您無法編輯或刪除標籤的金鑰或值。具 `aws:` 字首的標籤，不算在受資源限制的標籤計數內。

您無法僅根據標籤終止、停止或刪除資源。您必須指定資源識別符。例如，若要刪除您套用稱為 `DeleteMe` 標籤金鑰的快照，您必須搭配快照的資源識別符 (例如 `DeleteSnapshot`) 使用 `snap-1234567890abcdef0` 動作。

如需有關您可以標記的 MemoryDB 資源的詳細資訊，請參閱您可以標記的資源。

為資源加上標籤的範例

- 新增標籤至叢集。

```
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"
```

- 使用標籤建立叢集。

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

```
--cluster-name testing-tags \
--description cluster-test \
--subnet-group-name test \
--node-type db.r6g.large \
--acl-name open-access \
--tags Key="project",Value="XYZ" Key="memorydb",Value="Service"
```

- 使用標籤建立快照。

  在此情況下，如果您根請求新增了標籤，即使叢集包含標籤，快照也只會接收請求標籤。

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

使用成本配置標籤監控成本

將成本配置標籤新增至 Redis 中的 MemoryDB 中的資源時，您可以將發票上的費用依資源標籤值分組，藉此追蹤成本。

內存數據庫成本配置標籤是您所定義並與內存數據庫資源建立關聯的索引鍵值組。索引鍵與值皆會區分大小寫。您可以使用標籤索引鍵來定義類別，而標籤值可為該類別中的某個項目。例如，您可以定義標籤索引鍵 `CostCenter`，且標籤值為 `10010`，指出資源是指派給 10010 成本中心。您也可利用 `Environment` 之類的索引鍵，和 `test` 或 `production` 之類的值，以使用標籤來指定用於測試或生產的資源。我們建議您使用一組一致的標籤索引鍵，讓您更輕鬆地追蹤與資源關聯的成本。

您可以使用成本配置標籤來整理您的 AWS 帳單，以反映您自己的成本結構。方式是註冊以取得包含標籤鍵值的 AWS 帳戶帳單。接著，若要查看合併資源的成本，請根據具有相同標籤鍵值的資源來整理您的帳單資訊。例如，您可以使用特定應用程式名稱來標記數個資源，然後整理帳單資訊以查看該應用程式跨數項服務的總成本。

您也可以結合標籤來以更高的細節層次追蹤成本。例如，若要按區域追蹤您的服務成本，您可以使用標籤索引鍵 `Service` 和 `Region`。在某個資源上，您會有值 `MemoryDB` 和 `Asia Pacific (Singapore)`，而在另一個資源上，則有值 `MemoryDB` 和 `Europe (Frankfurt)`。您可以依區域劃分來查看您的總 MemoryDB 成本。如需詳細資訊，請參閱《AWS Billing 使用者指南》中的使用成本分配標籤。

您可以將 MemyDB 成本配置標籤新增至 MemoryDB 叢集。新增、列出、修改、複製或移除標籤時，該操作只會套用至指定的叢集。

成本配置標籤的特性

- 成本配置標籤會套用至內存數據庫資源，這些資源是在 CLI 和 API 操作中以 ARN 的形式指定。資源類型會是「叢集」。

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

  ARN 範例：`arn:aws:memorydb:us-east-1:1234567890:cluster/my-cluster`
- 標籤金鑰是標籤必要的名稱。索引鍵的字串值的長度可以是 1 到 128 個 Unicode 字元，不可在前面加上 `aws:`。此字串只能包含一組 Unicode 字母、數字、空格、底線 (_)、句點 (.)、冒號 (:)、反斜線 (\ )、等號 (=)、加號 (+)、連字號 (-) 或 @ 符號 (@)。
- 標籤值為標籤的選用值。值的字串值長度可以是 1 到 256 個 Unicode 字元，不可在前面加上 `aws:`。此字串只能包含一組 Unicode 字母、數字、空格、底線 (_)、句點 (.)、冒號 (:)、反斜線 (\ )、等號 (=)、加號 (+)、連字號 (-) 或 @ 符號 (@)。
- 內存 DB 資源的上限為 50 個標籤。
- 標籤組中的值不必是唯一的。例如，您可以有一個標籤組，其中的索引鍵 `Service` 和 `Application` 都有值 `MemoryDB`。

AWS 不會將任何語意意義套用至標籤。標籤會嚴格解譯為字元字串。AWS不會對任何 MemoryDB 資源自動設定任何標籤。

使用 AWS CLI 管理成本配置標籤

您可以使用 AWS CLI 來新增、修改或移除成本配置標籤。

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

主題

- 使用 AWS CLI 列出標籤
- 使用 AWS CLI 新增標籤
- 使用 AWS CLI 修改標籤
- 使用 AWS CLI 移除標籤

使用 AWS CLI 列出標籤

您可以使用AWS CLI若要列出現有 MemoryDB 資源上的標籤，方法是使用列出標籤operation.

下面的代碼使用AWS CLI列出內存數據庫羣集上的標籤`my-cluster`在 us-east-1 區域中。

針對 Linux、macOS 或 Unix：

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

針對 Windows：

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

此操作的輸出看起來應該類似以下，這是資源上所有標籤的清單。

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

如果資源上沒有標籤，輸出會是空的TagList。

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

如需詳細資訊，請參閱 。AWS CLI適用於內存 DB列出標籤。

使用 AWS CLI 新增標籤

您可以使用AWS CLI若要將標籤新增至現有的 MemoryDB 資源，方法是使用tag-resourceCLI 操作。如果標籤索引鍵不存在於資源上，則索引鍵和值會新增至資源。如果索引鍵已存在於資源上，則與該索引鍵相關聯的值會更新為新的值。

下面的代碼使用AWS CLI添加密鑰`Service`和`Region`的值`memorydb`和`us-east-1`分別添加到羣集`my-cluster`在 us-east-1 區域中。

針對 Linux、macOS 或 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
```

針對 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
```

此操作的輸出看起來應該類似以下，這是在操作後資源上所有標籤的清單。

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

如需詳細資訊，請參閱 。AWS CLI適用於內存 DBtag-resource。

您也可以使用AWS CLI若要在建立新叢集時將標籤新增至叢集，方法是使用操作create-cluster。

使用 AWS CLI 修改標籤

您可以使用AWS CLI來修改 MemoryDB 叢集上的標籤。

若要修改標籤：

- 使用tag-resource來新增標籤和值，或是變更與現有標籤關聯的值。
- 使用untag-resource來從資源移除指定的標籤。

這兩項操作的輸出會是指定叢集上標籤和其值的清單。

使用 AWS CLI 移除標籤

您可以使用AWS CLI來從 MemoryDB 叢集移除標籤，方法是使用untag-resourceoperation.

下面的代碼使用AWS CLI刪除帶有鍵的標籤`Service`和`Region`從羣集`my-cluster`在 us-east-1 區域中。

針對 Linux、macOS 或 Unix：

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

針對 Windows：

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

此操作的輸出看起來應該類似以下，這是在操作後資源上所有標籤的清單。

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

如需詳細資訊，請參閱 。AWS CLI適用於內存 DBuntag-resource。

使用 MemoryDB API 管理成本配置標籤

您可以使用 MemoryDB API 來新增、修改或移除成本配置標籤。

成本配置標籤會套用至叢集的 MemoryDB。要加標籤的叢集是使用 ARN (Amazon Resource Name) 來指定。

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

主題

- 使用 MemoryDB API 列出標籤
- 使用 MemoryDB API 新增標籤
- 使用 MemoryDB API 修改標籤
- 使用 MemoryDB API 移除標籤

使用 MemoryDB API 列出標籤

您可以使用 MemoryDB API 來列出現有資源上的標籤，方法是使用ListTagsoperation.

下列程式碼使用 MemoryDB API 來列出資源上的標籤`my-cluster`在 us-east-1 區域中。

```
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>
```

使用 MemoryDB API 新增標籤

您可以使用 MemoryDB API 來將標籤新增至現有的 MemoryDB 叢集，方法是使用TagResourceoperation. 如果標籤索引鍵不存在於資源上，則索引鍵和值會新增至資源。如果索引鍵已存在於資源上，則與該索引鍵相關聯的值會更新為新的值。

下面的代碼使用內存 DB API 添加密鑰`Service`和`Region`的值`memorydb`和`us-east-1`分別添加到資源`my-cluster`在 us-east-1 區域中。

```
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>
```

如需詳細資訊，請參閱「」TagResource。

使用 MemoryDB API 修改標籤

您可以使用 MemoryDB API 來修改 MemoryDB 叢集上的標籤。

若要修改標籤的值：

- 使用 TagResource 操作來新增標籤和值，或是變更現有標籤的值。
- 使用 UntagResource 來從資源移除標籤。

這兩項操作的輸出會是指定資源上標籤和其值的清單。

使用 MemoryDB API 移除標籤

您可以使用 MemoryDB API 來從現有 MemoryDB 叢集移除標籤，方法是使用UntagResourceoperation.

下面的代碼使用內存 DB API 來刪除帶有密鑰的標籤`Service`和`Region`從羣集`my-cluster`在 us-east-1 區域中。

```
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>
```

管理維護作業

每個叢集都有每週一次的維護時段，會在此期間套用任何系統變更。若您在建立或修改快取叢集時，未指定偏好的維護時段，MemyDB 會隨機選取一週中某一天指派您區域維護時段內 60 分鐘的維護時段。

60 分鐘的維護時段隨機選自每個區域的 8 小時時段。以下資料表列出每個區域的時段，預設維護時段會從此時段中指派。您可以選擇區域維護時段以外的偏好維護時段。

區域代碼	區域名稱	區域維護時段
ap-northeast-1	亞太區域 (東京)	下午 1 時至 9 時 (UTC)
ap-northeast-2	Asia Pacific (Seoul) Region	中午 12 時至下午 8 時 (UTC)
ap-south-1	Asia Pacific (Mumbai) Region	17:30-1:30 (UTC)
ap-southeast-1	亞太區域 (新加坡)	下午 2 時至 10 時 (UTC)
ap-east-1	亞太區域 (香港) 區域	下午 1 時至 9 時 (UTC)
ap-southeast-2	Asia Pacific (Sydney) Region	中午 12 時至下午 8 時 (UTC)
cn-north-1	中國 (北京) 區域	下午 2 時至 10 時 (UTC)
cn-northwest-1	中國 (寧夏) 區域	下午 2 時至 10 時 (UTC)
eu-west-3	歐洲 (巴黎) 區域	下午 11 時 59 分至上午 7 時 29 分 (UTC)
eu-central-1	歐洲區域 (法蘭克福)	下午 11 時至次日上午 7 時 (UTC)
eu-west-1	歐洲 (愛爾蘭) 區域	下午 10 時至次日上午 6 時 (UTC)
eu-west-2	Europe (London) Region	下午 11 時至次日上午 7 時 (UTC)
sa-east-1	南美洲區域 (聖保羅)	01:00-09:00 (UTC)
ca-central-1	Canada (Central) Region	上午 3 時至上午 11 時 (UTC)
us-east-1	US East (N. Virginia) Region	上午 3 時至上午 11 時 (UTC)

區域代碼	區域名稱	區域維護時段
us-east-1	US East (Ohio) Region	04:00-12:00 (UTC)
us-west-1	US West (N. California) Region	上午 6 時至下午 2 時 (UTC)
us-west-2	美國西部區域 (奧勒岡)	上午 6 時至下午 2 時 (UTC)

變更叢集的維護時段

維護時段應落在使用量最低的時段，因此可能需要不時進行調整。您可以修改叢集來指定時間範圍，最多 24 小時，您已請求的所有維護活動會在此期間進行。在此期間會進行所有您請求的延遲或待處理叢集修改。

其他資訊

如需維護時段和節點取代的資訊，請參閱下列內容：

- 替換節點 - 管理節點更換
- 修改記憶體資料庫叢集 - 變更叢集的維護時段

最佳實務

以下提供 Redis 的建建議最實務。遵循這些內容，可改善您叢集的效能和可靠性。

主題

- 受限制的 Redis 命令
- Redis 內存數據庫中的恢復能力
- 提供發佈/訂閱閱讀和增實務
- 最佳實務：線上叢集大小調整

受限制的 Redis 命令

為了提供受管服務體驗，MemoryDB 會限制存取某些需要進階權限的命令。下列指令無法使用：

- 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

Redis 內存數據庫中的恢復能力

AWS 全球基礎設施是以 AWS 區域與可用區域為中心建置的。AWS區域提供多個分開且隔離的實際可用區域，它們以低延遲、高輸送量和高度備援聯網功能相互連結。透過可用區域，您所設計與操作的應用程式和資料庫，就能夠在可用區域之間自動容錯移轉，而不會發生中斷。可用區域的可用性、容錯能力和擴充能力，均較單一或多個資料中心的傳統基礎設施還高。

如需有關 AWS 區域與可用區域的詳細資訊，請參閱 AWS 全域基礎設施。

除了AWS全球基礎設施，Redis 提供數種功能，可協助支援資料的彈性和快照需求。

主題

- 緩解故障

緩解故障

規劃 Redis 實施時，您應該規劃讓故障對您的應用程式和數據的影響最小。本節中的主題涵蓋您可以採取用來保護您的應用程式和資料不受故障的方法。

緩解故障：內存數據庫集

MemeryDB 羣集包含您應用程式可以對其同時進行寫入和讀取的單一主要節點，以及 0 到 5 個唯讀複本節點。但是，我們強烈建議至少使用 1 個副本以實現高可用性。無論何時將數據寫入主要節點，它會保留到事務日誌並於複本節點上異步更新。

當僅供讀取複本故障時

1. 內存 DB 會偵測到故障的複本。
2. 內存 DB 使故障的節點離線。
3. 內存 DB 啟動並於相同 AZ 中佈建替代節點。
4. 新節點會與事務日誌同步。

在這段時間，您的應用程式可以繼續使用其他節點來讀取和寫入。

內存數據庫多可用區域

如果您的內存 DB 集閤中激活多可用區，將偵測並自動取代故障的主要節點。

1. 內存 DB 偵測到主要節點故障。

2. MemoryDB 在確保副本與故障的主複製副本保持一致後故障轉移到副本。
3. 內存 DB 會在故障主要節點的可用區域中旋轉複本。
4. 新節點會與事務日誌同步。

容錯移轉至複本節點的速度一般會較建立和佈建新主要節點來得快。這表示您的應用程式可以較早繼續寫入至您的主要節點。

如需詳細資訊，請參閱利用異地同步備份將 MemoryDB 中的停機時間。

提供發佈/訂閱閱讀和增實務

使用 Redis 7 或更新版本時，我們建議使用碎片發佈/訂閱。您也可以使用增強型 I/O 多工來改善輸送量和延遲，這在使用 Redis 7 版或更新版本時會自動提供，而且不需要變更用戶端。它非常適合發佈/訂閱工作負載，這些工作負載通常為具有多個用戶端連線的輸送量繫結。

最佳實務：線上叢集大小調整

「重新碎片」涉及將碎片或節點新增到您的叢集或從叢集移除碎片或節點，並重新分配鍵的空間。多項事物會對重新碎片操作造成影響，例如叢集上的負載、記憶體使用率，以及整體資料大小。為了取得最佳體驗，我們建議您遵循統一工作負載模式分佈的整體叢集最佳實務。此外，我們建議您採取以下步驟。

在初始化重新碎片前，我們建議以下內容：

- 測試您的應用程式 - 在預備環境中測試應用程式於重新分片期間的行為 (若可能的話)。
- 提早取得擴展問題的通知 - 重新分片是一項需要大量運算的操作。因此，我們建議在重新分片期間，將多核心執行個體的 CPU 使用率保持在 80% 以下，在單核心執行個體上保持低於 50%。在應用程式開始觀察到擴展問題之前，監控制存在 MemoryDb 指標準，並在應用程式 可進行追蹤的有用指標包括 `CPUUtilization`、`NetworkBytesIn`、`NetworkBytesOut`、`CurrConnections`、`NewConnections` 及 `BytesUsedForMemoryDB`。
- 在向內擴展前確保有足夠的可用記憶體 - 若您要向內擴展，請確保碎片上保留的可用記憶體至少是待移除碎片所使用記憶體的 1.5 倍。
- 在離峰時段期間起始重新分片程序 - 此做法有助於減少重新分片作業期間的延遲，以及對用戶端造成的輸送量影響。它也有助於更快完成重新碎片，因為有更多的資源可用於重新分配位置。
- 檢閱用戶端逾時行為 - 有些用戶端可能會在線上叢集調整大小期間產生較高的延遲。使用較高的逾時設定您的用戶端程式庫，有助於解決該情況，即使伺服器上負載較高，系統仍有時間連線。在某些情況下，您可能想要對伺服器開啟大量連線。在這些情況下，請考慮將指數退避新增到重新連線邏輯。這麼做可幫助防止爆量的新連線同時衝擊伺服器。

在重新碎片期間，我們建議以下內容：

- 避免費時命令 -避免執行任何需要大量運算或輸入/輸出的操作，例如 `KEYS` 和 `SMEMBERS` 命令。我們建議此方法，因為這些操作會增加叢集上的負載，並會影響叢集的效能。請改為使用 `SCAN` 和 `SSCAN` 命令。

- 遵循 Lua 最佳實務 - 避免長時間執行的 Lua 指令碼，並且一律預先宣告用於 Lua 指令碼的索引鍵。我們建議此方法來判斷 Lua 指令碼並未使用跨位置命令。確認用於 Lua 指令碼中的鍵屬於相同位置。

在重新碎片之後，請注意以下內容：

- 若目標碎片上的可用記憶體不足，向內擴展可能會僅部分成功。若發生這種結果，請檢閱可用記憶體，並視需要重試操作。
- 擁有大量項目的位置不會進行遷移。特別是擁有大於 256 MB 項目位置的後序列化不會進行遷移。
- FLUSHALL在重新分片操作期間，Lua 腳本中不支持和FLUSHDB命令。

了解記憶體資料庫複寫

MemoryDB 實作複寫，使用資料分割到最多 500 個碎片。

叢集中的每個碎片都具備單一讀/寫主節點，以及最多 5 個僅供讀取複本節點。每個主節點最高可承受 100 MB /秒。您可以建立具有較多碎片數量和較少複本數目的叢集，每個叢集最多可達 500 個節點。此叢集組態的範圍可以從 500 個碎片和 0 個複本到 100 個碎片和 4 個複本，這是允許的複本最大數量。

一致性

在 MemoryDB 中，主節點是強烈一致的。成功的寫入作業會在返回給用戶端之前，持久地儲存在分散式異地同步備份交易記錄中。對主要選項的讀取作業永遠傳回反映所有先前成功寫入作業之影響的最多up-to-date資料。如此強大的一致性會在主要容錯移轉之間保留。

在 MemoryDB，複本節點為最終一致性。複本的讀取作業 (使用READONLY命令) 可能不一定會反映最近成功寫入作業的影響，而延遲度量會發佈到CloudWatch。不過，來自單一複本的讀取作業會依序一致。成功的寫入作業會以在主複本上執行的相同順序，在每個複本上生效。

叢集中的複寫

碎片中的每個僅供讀取複本都會維護碎片主節點中的資料副本。使用交易記錄檔的非同步複寫機制可用來保持僅供讀取複本與主要複本同步。應用程式可從叢集內的任何節點進行讀取。應用程式只能寫入主要節點。僅供讀取複本可增強讀取延展性。由於 MemoryDB 將數據存儲在持久的事務日誌中，因此沒有數據將丟失的風險。資料會分割片中的每個碎片中。

應用程式會使用 MemoryDB 叢集的叢集端點與叢集中的節點連線。如需詳細資訊，請參閱尋找連線端點。

MemoryDB 叢集是地區性的，並且只能包含來自一個區域的節點。若要改善容錯能力，您必須在該區域內的多個可用區域佈建主要和僅供讀取複本。

強烈建議所有 MemoryDB 叢集使用提供異地同步備份的複寫功能。如需詳細資訊，請參閱利用異地同步備份將 MemoryDB 中的停機時間。

利用異地同步備份將 MemoryDB 中的停機時間

有許多執行個體可能需要取代主要節點；這些執行個體包括特定類型的計劃維護，以及主節點或可用區域故障的不太可能發生事件。

節點故障的回應取決於哪個節點失敗。但是，在所有情況下，MemoryDB 都可確保在節點更換或容錯移轉期間不會遺失任何資料。例如，如果複本失敗，則會取代失敗的節點，並從交易記錄檔同步資料。如果主節點故障，將觸發容錯移轉至一致的複本，以確保在容錯移轉期間不會遺失任何資料。現在可從新的主要節點提供寫入作業。然後會取代舊的主要節點，並從交易記錄檔同步處理。

如果主節點在單一節點碎片上失敗 (無複本)，MemoryDB 會停止接受寫入，直到主節點被取代並從交易記錄檔同步為止。

節點更換可能會導致叢集停機，但如果異地同步備份處於作用中狀態，停機時間將最小化。主節點的角色將自動容錯移轉到其中一個複本。沒有必要創建和佈建一個新的主節點，因為 MemoryDB 將透明地處理這個問題。此容錯移轉及複本提升可確保您能在提升完成時立即繼續寫入新的主要節點。

如果由於維護更新或服務更新而起始計劃的節點更換，請注意在叢集提供傳入寫入要求時，已計劃的節點取代已完成。

MemoryDB 叢集上的異地同步備份可改善您的容錯能力。特別是在叢集的主要節點因任何原因無法存取或失敗的情況下，情況尤其如此。MemoryDB 叢集上的異地同步備份要求每個碎片具有多個節點，並且會自動啟用。

主題

- 具有異地同步備份回應的故障案例
- 測試自動容錯移轉

具有異地同步備份回應的故障案例

如果異地同步備份處於作用中狀態，則故障的主要節點容錯移轉至可用的複本。複本會自動與交易記錄檔同步處理，並成為主要節點，這比建立和重新佈建新的主要節點快得多。此程序通常只需要幾秒鐘，您便能再次寫入叢集。

異地同步備份處於作用中狀態時，MemoryDB 會持續監控主節點的狀態。若主要節點故障，便會根據故障的類型執行以下其中一個動作。

主題

- 只有主節點故障的故障案例

- 主節點和某些複本失敗時的失敗情況
- 整個叢集故障的故障案例

只有主節點故障的故障案例

如果只有主節點失敗，複本將自動成為主要節點。然後，會在與故障主要複本相同的可用區域中建立並佈建取代複本。

當只有主節點發生故障時，MemoryDB 異地同步備份會執行下列動作：

1. 失敗的主要節點會離線。
2. up-to-date複本會自動成為主要副本。

   寫入可以在容錯移轉程序完成後立即恢復，通常只需幾秒鐘。
3. 已啟動並佈建取代複本。

   取代複本會在故障主要節點所在的可用區域中啟動，以維護節點的分佈。
4. 複本會與交易記錄檔同步。

如需尋找叢集端點的資訊，請參閱下列主題：

- 尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

主節點和某些複本失敗時的失敗情況

如果主要複本和至少一個複本失敗，則會將up-to-date複本提升為主要叢集。也會在與故障節點相同的可用區域中建立和佈建新複本。

當主節點和某些複本失敗時，MemoryDB 異地同步備份會執行下列動作：

1. 失敗的主要節點和失敗的複本會離線。
2. 可用的複本將成為主要節點。

   寫入可以在容錯移轉完成後立即恢復，通常只需幾秒鐘。
3. 建立及佈建替換用的複本。

   替換用的複本會在失敗節點所在的可用區域內建立，維持節點的分佈。

4. 所有節點都會與交易記錄同步。

如需尋找叢集端點的資訊，請參閱下列主題：

- 尋找記憶體 DB 叢集的端點 (AWSCLI)
- 尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

整個叢集故障的故障案例

若所有項目都失敗，便會在原始節點的相同可用區域內重新建立及佈建所有節點。

沒有資料遺失在這個案例中，因為資料保存在交易記錄檔中。

當整個叢集發生故障時，MemoryDB 異地同步備份會執行下列動作：

1. 失敗的主要節點和複本會離線。
2. 會建立並佈建取代主要節點，並與交易記錄同步。
3. 建立和佈建取代複本，並與交易記錄檔同步。

   替代項目會在失敗節點所在的可用區域內建立，維持節點的分佈。

如需尋找叢集端點的資訊，請參閱下列主題：

- 尋找記憶體 DB 叢集的端點 (AWSCLI)
- 尋找記憶體資料庫叢集的端點 (記憶體資料庫 API)

測試自動容錯移轉

您可以使用 MemoryDB 主控台、和 MemoryDB API 來測試自動容錯移轉。AWS CLI

在測試時，請注意下列事項：

- 在任何 24 小時期間內，您最多可以使用五次此操作。
- 如果您在不同叢集中的碎片上呼叫此作業，則可以同時進行呼叫。
- 在某些情況下，您可能會在相同 MemoryDB 叢集中的不同碎片上多次呼叫此作業。在這種情況下，必須先完成第一個節點取代，才能夠執行後續呼叫。
- 若要判斷節點取代是否完成，請使用 Redis 的主控台、或 MemoryDB API 的記憶體資料庫來AWS CLI檢查事件。查找與以下相關的事件`FailoverShard`，這裡列出的可能發生順序：

  1. 集群消息：`FailoverShard API called for shard <shard-id>`
  2. 集群消息：`Failover from primary node <primary-node-id> to replica node <node-id> completed`
  3. 集群消息：`Recovering nodes <node-id>`
  4. 集群消息：`Finished recovery for nodes <node-id>`

  如需詳細資訊，請參閱下列內容：

  - DescribeEvents在內存數據庫 API 參考
- 此 API 是專為在 MemoryDB 容錯移轉的情況下測試應用程式的行為而設計的。並非設計成啟動容錯移轉以解決叢集問題的操作工具。此外，在某些情況下，例如大規模操作活動，AWS 可能會封鎖此 API。

主題

- 使用 AWS Management Console 測試自動容錯移轉
- 使用 AWS CLI 測試自動容錯移轉
- 使用記憶體資料庫 API 測試自動容錯移轉

使用 AWS Management Console 測試自動容錯移轉

使用下列程序，透過主控台測試自動容錯移轉。

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。

2. 選擇您要測試的叢集左側的圓鈕。此叢集必須至少有一個複本節點。
3. 在 Details (詳細資訊) 區域中，確認此叢集已啟用異地同步備份。若叢集尚未啟用多個可用區，請選擇不同叢集，或是修改此叢集以啟用多個可用區。如需詳細資訊，請參閱修改記憶體資料庫叢集。
4. 選擇叢集名稱。
5. 在 [碎片和節點] 頁面上，針對您要測試容錯移轉的碎片，選擇碎片的名稱。
6. 對於節點，選擇「容錯移轉主要」。
7. 選擇 Continue (繼續) 來容錯移轉主要節點，或是 Cancel (取消) 來取消操作而不容錯移轉主要節點。

   在容錯移轉程序期間，主控台會繼續將節點的狀態顯示為「可用」。若要追蹤容錯移轉測試的進度，請從主控台導覽窗格選擇 Events (事件)。在 Events (事件) 標籤上，觀察指出您容錯移轉已啟動的事件 (`FailoverShard API called`) 並完成 (`Recovery completed`)。

使用 AWS CLI 測試自動容錯移轉

您可以使用AWS CLI作業容錯移轉碎片，在任何啟用異地同步備份的叢集上測試自動容錯移轉。

參數

- `--cluster-name` - 必要。要測試的叢集。
- `--shard-name` - 必要。您要測試自動容錯移轉的碎片名稱。在連續 24 小時期間內，您最多可以測試五個碎片。

下列範例會使用AWS CLI來呼叫 `failover-shard` MemoryDB 叢集`0001`中的碎片。`my-cluster`

若為 Linux、macOS 或 Unix：

```
aws memorydb failover-shard \
   --cluster-name my-cluster \
   --shard-name 0001
```

針對 Windows：

```
aws memorydb failover-shard ^
   --cluster-name my-cluster ^
```

```
    --shard-name 0001
```

若要追蹤您容錯移轉的進度，請使用 AWS CLI `describe-events` 操作。

它將返回以下 JSON 響應：

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

如需詳細資訊，請參閱下列內容：

- 容錯移轉分片
- describe-events

使用記憶體資料庫 API 測試自動容錯移轉

下列範例會呼叫`FailoverShard`叢集`0003`中的`memorydb00`碎片。

Example 測試自動容錯移轉

```
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>
```

若要追蹤容錯移轉的進度，請使用 MemoryDB `DescribeEvents` API 作業。

如需詳細資訊，請參閱下列內容：

- FailoverShard
- DescribeEvents

變更複本的數量

您可以使用AWS Management Console、或 MemoryDB API 來動態增加或減少 MemoryDB 叢集中的僅供讀取複本數量。AWS CLI所有碎片必須具有相同數量的複本。

增加叢集中複本的複本數量

您可以將 MemoryDB 叢集中複本的複本數量。每個碎片最多五個。您可以使用AWS Management Console、或 MemoryDB API 來執行此作業。AWS CLI

主題

- 使用 AWS Management Console
- 使用 AWS CLI
- 使用 ReplicyDB API

使用 AWS Management Console

若要增加 MemoryDB 叢集 (主控台) 中的複本數目，請參閱。從叢集新增/移除節點

使用 AWS CLI

若要增加 MemoryDB 叢集中的複本數量，請使用`update-cluster`命令搭配下列參數：

- `--cluster-name` - 必要。識別您希望增加複本數量的叢集。
- `--replica-configuration` - 必要。允許您設定複本的複本數量。若要增加複本計數，請將`ReplicaCount`屬性設定為此作業結束後，您希望此碎片片中擁有的複本數。

Example

以下範例會`my-cluster`將叢集中複本的複本數量。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --replica-configuration \
        ReplicaCount=2
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --replica-configuration ^
        ReplicaCount=2
```

它會傳回下列 JSON 回應：

```
{
    "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
    }
}
```

若要在已更新叢集的狀態從更新變更為可用時檢視其詳細資訊，請使用下列命令：

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

它將返回以下 JSON 響應：

```
{
    "Clusters": [
        {
            "Name": "my-cluster",
            "Status": "available",
            "NumberOfShards": 1,
            "Shards": [
                {
                    "Name": "0001",
                    "Status": "available",
                    "Slots": "0-16383",
                    "Nodes": [
                        {
                            "Name": "my-cluster-0001-001",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-002",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1b",
                            "CreateTime": "2021-08-21T20:22:12.405000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                                "Port": 6379
                            }
                        },
                        {
                            "Name": "my-cluster-0001-003",
                            "Status": "available",
                            "AvailabilityZone": "us-east-1a",
                            "CreateTime": "2021-08-22T12:59:31.844000-07:00",
                            "Endpoint": {
                                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
```

```
                                "Port": 6379
                            }
                        }
                    ],
                    "NumberOfNodes": 3
                }
            ],
            "ClusterEndpoint": {
                "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
                "Port": 6379
            },
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupName": "default.memorydb-redis6",
            "ParameterGroupStatus": "in-sync",
            "SubnetGroupName": "my-sg",
            "TLSEnabled": true,
            "ARN": "arn:aws:memorydb:us-east-1:xxxxxxexamplearn:cluster/my-cluster",
            "SnapshotRetentionLimit": 0,
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "SnapshotWindow": "04:30-05:30",
            "ACLName": "my-acl",
            "DataTiering": "false",
            "AutoMinorVersionUpgrade": true
        }
    ]
}
```

如需使用 CLI 增加複本數量的詳細資訊，請參閱命令參考中的更新叢集。AWS CLI

使用 ReplicyDB API

若要增加 MemoryDB 碎片中的複本數目，請使用具有下列參數的UpdateCluster動作：

- ClusterName - 必要。識別您希望增加複本數量的叢集。
- ReplicaConfiguration - 必要。允許您設定複本的複本數量。若要增加複本計數，請將ReplicaCount屬性設定為此作業結束後，您希望此碎片片中擁有的複本數。

Example

以下範例會`sample-cluster`將叢集中複本的複本數量。範例完成後，每個碎片中都有三個複本。無論這是具有單一碎片的 MemoryDB 叢集還是具有多個碎片的 MemoryDB 叢集，都會套用此數字。

```
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>
```

如需使用 API 增加複本數量的詳細資訊，請參閱UpdateCluster。

減少叢集中複本的複本數量

您可以減少 MemoryDB 叢集中複本的複本數。您可以將複本的複本數量，但如果您的主節點故障，您無法容錯移轉到複本。

您可以使用AWS Management Console、AWS CLI或 MemoryDB API 來減少叢集中複寫的複本數量。

主題

- 使用 AWS Management Console
- 使用 AWS CLI
- 使用 ReplicyDB API

使用 AWS Management Console

若要減少 MemoryDB 叢集 (主控台) 中的複本數目，請參閱。從叢集新增/移除節點

使用 AWS CLI

若要減少 MemoryDB 叢集中複本的複本數量，請使用`update-cluster`命令搭配下列參數：

- `--cluster-name` - 必要。識別要減少複本數目的叢集。
- `--replica-configuration` - 必要。

  `ReplicaCount`-設定此屬性指定您希望的複本節點數。

Example

以下範例會`--replica-configuration`使用叢集`my-cluster`中複寫的複本數量。

若為 Linux、macOS 或 Unix：

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

針對 Windows：

```
aws memorydb update-cluster ^
```

```
    --cluster-name my-cluster ^
    --replica-configuration ^
        ReplicaCount=1 ^
```

它將返回以下 JSON 響應：

```
{
    "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
    }
}
```

若要在已更新叢集的狀態從更新變更為可用時檢視其詳細資訊，請使用下列命令：

若為 Linux、macOS 或 Unix：

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

針對 Windows：

```
aws memorydb describe-clusters ^
```

```
    --cluster-name my-cluster
    --show-shard-details
```

它將返回以下 JSON 響應：

```
{
    "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
        }
    ]
}
```

如需使用 CLI 減少複本數量的詳細資訊，請參閱命令參考中的更新叢集。AWS CLI

使用 ReplicyDB API

若要減少 MemoryDB 叢集中的複本數目，請搭配下列參數使用UpdateCluster動作：

- `ClusterName` - 必要。識別要減少複本數目的叢集。
- `ReplicaConfiguration` - 必要。允許您設定複本的複本數量。

  `ReplicaCount`-設定此屬性指定您希望的複本節點數。

Example

以下範例會ReplicaCount使用叢集sample-cluster中複本的複本數量。範例完成後，每個碎片中都有一個複本。無論這是具有單一碎片的 MemoryDB 叢集還是具有多個碎片的 MemoryDB 叢集，都會套用此數字。

```
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>
```

如需使用 API 減少複本數量的詳細資訊，請參閱UpdateCluster。

快照和還原

適用於 Redis 的 MemoryDB 叢集會自動將資料備份到異地同步備份交易記錄，但您可以選擇定期或隨選建立叢集的 point-in-time 快照。這些快照可用於在上一點重新建立叢集，或植入全新的叢集。快照集包含叢集的中繼資料以及叢集中的所有資料。所有快照都寫入 Amazon Simple Storage Service (Amazon S3)，該服務提供耐用的儲存。您可以隨時透過建立新的 MemoryDB 叢集並使用快照中的資料填入叢集來還原資料。透過記憶體資料庫，您可以使用 AWS Command Line Interface (AWS CLI) 和記憶體資料庫 API 來管理快照集。AWS Management Console

主題

- 快照限制
- 快照成本
- 排程自動快照
- 製作手動快照
- 建立最終快照
- 說明快照
- 複製快照
- 匯出快照
- 從快照還原
- 使用外部建立的快照植入新叢集
- 標記快照
- 刪除快照

快照限制

計劃或製作快照時，請考慮下列限制：

- 對於 MemoryDB 叢集，所有支援的節點類型都可以使用快照和還原。
- 在任何連續的 24 小時期間內，每個叢集最多可建立 20 個手動快照。
- MemoryDB 僅支持在集群級別上拍攝快照。MemoryDB 不支援在碎片或節點層級拍攝快照。
- 在快照過程中，您無法在叢集上執行任何其他 API 或 CLI 作業。
- 如果您刪除叢集並要求最終快照集，MemoryDB 一律會從主節點擷取快照集。這樣可確保您在刪除叢集之前擷取最新的資料。

快照成本

使用 MemoryDB，您可以免費為每個使用中的 MemoryDB 叢集儲存一個快照。所有區域的額外快照儲存空間按每月 0.085/GB 的費率計費。AWS建立快照或將資料從快照還原至 MemoryDB 叢集無需支付資料傳輸費用。

排程自動快照

對於任何 MemoryDB 叢集，您都可以啟用自動快照。啟用自動快照後，MemoryDB 會每天建立叢集的快照。不會對叢集造成任何影響，而且變更會立即生效。如需詳細資訊，請參閱從快照還原。

排程自動快照時，您應該規劃下列設定：

- 快照視窗 — MemoryDB 開始建立快照的每一天期間。快照視窗的最小長度為 60 分鐘。您可以在最方便的任何時間設定快照視窗，也可以設定在一天中避免在特別高使用率期間執行快照的時間。

  如果您沒有指定快照視窗，MemoryDB 會自動指派一個快照視窗。
- 快照保留限制 — 快照在 Amazon S3 中保留的天數。例如，如果您將保留限制設定為 5，則今天拍攝的快照會保留 5 天。當保留限制到期時，系統會自動刪除快照集。

  最大快照保留限制為 35 天。如果快照保留限制設為 0，則會停用叢集的自動快照。即使禁用了自動快照，MemoryDB 數據仍然是完全耐用的。

您可以在使用 MemoryDB 主控台、或 MemoryDB API 建立 MemoryDB 叢集時啟用或停用自動快照集。AWS CLI您可以在建立 MemoryDB 叢集時，核取 [快照] 段落中的啟用自動備份方塊來啟用自動快照。如需詳細資訊，建立記憶體資料庫叢集。

製作手動快照

除了自動快照之外，您還可以隨時建立手動快照。不同於自動快照會在指定保留期間後自動刪除，手動快照沒有保留期限，之後就會自動刪除快照。您必須手動刪除任何手動快照。即使您刪除叢集或節點，該叢集或節點的任何手動快照也會保留。如果您不想再保留手動快照，您必須自行明確刪除它。

手動快照對於測試和歸檔非常有用。例如，假設您開發了一組用於測試的基準資料。您可以創建數據的手動快照，並隨時恢復它。測試修改資料的應用程式之後，您可以建立新叢集並從基準快照還原來重設資料。叢集就緒時，您可以根據基準資料再次測試應用程式，並視需要經常重複此程序。

除了直接建立手動快照之外，您還可以使用下列其中一種方式建立手動快照：

- 複製快照— 來源快照是自動還是手動建立並不重要。
- 建立最終快照— 在刪除叢集之前立即建立快照。

其他重要議題

- 快照限制
- 快照成本

您可以使用AWS Management Console、或 MemoryDB API 建立節點的手動快照。AWS CLI

建立手動快照 (主控台)

若要建立叢集的快照 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從左側導覽窗格中，選擇 [叢集]。

   會出現「記憶體資料庫叢集」畫面。
3. 選擇您要備份的 MemoryDB 叢集名稱左側的圓形按鈕。
4. 選擇操作，然後選擇拍攝快照。
5. 在 [快照] 視窗中，在 [快照名稱] 方塊中輸入快照的名稱。建議您使用名稱指出已備份的叢集，以及建立快照的日期和時間。

   叢集命名限制條件如下：

- 必須包含 1-40 個英數字元或連字號。
- 必須以字母開頭。
- 不能連續包含兩個連字號。
- 結尾不能是連字號。

6. 在「加密」下，選擇要使用預設加密金鑰還是客戶管理的金鑰。如需詳細資訊，請參閱記憶體中的傳輸中加密 (TLS)。
7. 在「標籤」下方，選擇性地新增標籤以搜尋並篩選快照或追蹤AWS成本。
8. 選擇 Take Snapshot (擷取快照)。

   叢集的狀態會變更為「快照中」。當狀態恢復為可用時，即表示快照已完成。

建立手動快照 (AWSCLI)

若要使用建立叢集的手動快照AWS CLI，請使用具有下列參數的`create-snapshot`AWS CLI作業：

- `--cluster-name`— 用來做為快照來源的 MemoryDB 叢集的名稱。備份 MemoryDB 叢集時，請使用此參數。

  叢集命名限制條件如下：
  - 必須包含 1-40 個英數字元或連字號。
  - 必須以字母開頭。
  - 不能連續包含兩個連字號。
  - 結尾不能是連字號。

- `--snapshot-name` - 要建立的快照名稱。

相關主題

如需詳細資訊，請參閱 AWS CLI 命令參考中的 `create-snapshot`。

建立手動快照 (記憶體資料庫 API)

若要使用 MemoryDB API 建立叢集的手動快照集，請搭配下列參數使用 `CreateSnapshot` MemoryDB API 作業：

- `ClusterName`— 用來做為快照來源的 MemoryDB 叢集的名稱。備份 MemoryDB 叢集時，請使用此參數。

  叢集命名限制條件如下：
  - 必須包含 1-40 個英數字元或連字號。
  - 必須以字母開頭。
  - 不能連續包含兩個連字號。
  - 結尾不能是連字號。
- `SnapshotName` - 要建立的快照名稱。

相關主題

如需詳細資訊，請參閱CreateSnapshot。

建立最終快照

您可以使用 MemoryDB 主控台、或 MemoryDB API 建立最終快照集。AWS CLI

建立最終快照 (主控台)

當您使用 MemoryDB 主控台刪除 MemoryDB 叢集時，您可以建立最終快照集。

若要在刪除 MemoryDB 叢集時建立最終快照，請在刪除頁面上選擇是，並在指定快照的名稱。步驟 4：刪除叢集

建立最終快照 (AWSCLI)

您可以在使用刪除 MemoryDB 叢集時建立最終快照集。AWS CLI

刪除記憶體資料庫叢集時

若要在刪除叢集時建立最終快照，請使用具有下列參數的`delete-cluster`AWS CLI作業：

- `--cluster-name` - 正在刪除的叢集名稱。
- `--final-snapshot-name`— 最終快照的名稱。

下列程式碼會在刪除叢集`bkup-20210515-final`時取得最終快照`myCluster`。

若為 Linux、macOS 或 Unix：

```
aws memorydb delete-cluster \
        --cluster-name myCluster \
        --final-snapshot-name bkup-20210515-final
```

針對 Windows：

```
aws memorydb delete-cluster ^
        --cluster-name myCluster ^
        --final-snapshot-name bkup-20210515-final
```

如需詳細資訊，請參閱AWS CLI命令參考中的刪除叢集。

建立最終快照 (記憶體資料庫 API)

您可以在使用 MemoryDB API 刪除記憶體資料庫叢集時建立最終快照集。

刪除記憶體資料庫叢集時

若要建立最終快照集，請搭配下列參數使用 DeleteCluster MemoryDB API 作業。

- ClusterName - 正在刪除的叢集名稱。
- FinalSnapshotName— 快照的名稱。

下列記憶體資料庫 API 作業會在刪除叢集bkup-20210515-final時建立快照。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>
```

如需詳細資訊，請參閱DeleteCluster。

說明快照

下列程序說明如何顯示快照清單。您也可以視需要檢視特定快照的詳細資料。

描述快照（控制台）

使用顯示快照 AWS Management Console

1. 登入主控台
2. 從左側導覽窗格中選擇 [快照]。
3. 使用搜尋來篩選手動、自動或所有快照。
4. 若要查看特定快照的詳細資訊，請選擇快照名稱左邊的圓鈕。選擇「作業」，然後選擇「檢視明
5. 您也可以在「檢視詳細資訊」頁面中執行其他快照動作，例如複製、還原或刪除。您也可以將標籤新增至快照

說明快照 (AWSCLI)

若要顯示快照清單，以及選擇性地顯示特定快照的詳細資料，請使用 `describe-snapshots` CLI 作業。

範例

以下操作使用參數列`--max-results`出最多 20 個與您的帳戶相關聯的快照。省略參數`--max-results`清單最多 50 個快照。

```
aws memorydb describe-snapshots --max-results 20
```

下列作業會使用參數`--cluster-name`，僅列出與叢集相關聯的快照`my-cluster`。

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

下列作業會使用參數`--snapshot-name`來顯示快照的詳細資訊`my-snapshot`。

```
aws memorydb describe-snapshots --snapshot-name my-snapshot
```

如需詳細資訊，請參閱說明快照。

描述快照 (記憶體資料庫 API)

若要顯示快照清單，請使用此`DescribeSnapshots`作業。

範例

以下操作使用參數列MaxResults出最多 20 個與您的帳戶相關聯的快照。省略參數MaxResults清單最多 50 個快照。

```
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>
```

下列作業會使用參數列ClusterName出與叢集相關聯的所有快照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>
```

下列作業會使用參數SnapshotName來顯示快照的詳細資訊MyBackup。

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeSnapshots
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &SnapshotName=MyBackup
```

```
    &Timestamp=20210801T220302Z
    &Version=2021-01-01
    &X-Amz-Algorithm=Amazon4-HMAC-SHA256
    &X-Amz-Date=20210801T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20210801T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

如需詳細資訊，請參閱DescribeSnapshots。

複製快照

無論是自動還是手動建立快照，您都可以製作任何快照的副本。複製快照時，除非特別覆寫，否則會將與來源相同的 KMS 加密金鑰用於目標。您還可以導出快照，以便從 MemoryDB 外部訪問它。如需匯出快照的指引，請參閱匯出快照。

下列程序說明如何複製快照。

複製快照 (主控台)

若要複製快照 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看快照清單，請從左側導覽窗格中選擇 [快照]。
3. 從快照清單中，選擇要複製之快照名稱左側的圓鈕。
4. 選擇「動作」，然後選擇「複製」
5. 在 [複製快照] 頁面中，執行下列動作：
    a. 在 [新增快照名稱] 方塊中，輸入新快照的名稱。
    b. 將選用的 Target S3 Bucket (目標 S3 儲存貯體) 方塊留白。此欄位應僅用於匯出快照，且需要特殊的 S3 許可。如需匯出快照的資訊，請參閱匯出快照。
    c. 選擇要使用預設AWS KMS加密金鑰還是使用自訂金鑰。如需詳細資訊，請參閱記憶體中的傳輸中加密 (TLS)。
    d. 或者，您也可以將標籤新增至快照副本。
    e. 請選擇 Copy (複製)。

複製快照 (AWSCLI)

若要複製快照，請使用此`copy-snapshot`作業。

參數

- `--source-snapshot-name`— 要複製的快照名稱。
- `--target-snapshot-name`— 快照副本的名稱。
- `--target-bucket`— 保留用於匯出快照。建立快照複本時，請勿使用此參數。如需詳細資訊，請參閱匯出快照。

下列範例會建立自動快照的複本。

若為 Linux、macOS 或 Unix：

```
aws memorydb copy-snapshot \
    --source-snapshot-name automatic.my-primary-2021-03-27-03-15 \
    --target-snapshot-name my-snapshot-copy
```

針對 Windows：

```
aws memorydb copy-snapshot ^
    --source-snapshot-name automatic.my-primary-2021-03-27-03-15 ^
    --target-snapshot-name my-snapshot-copy
```

如需詳細資訊，請參閱複製快照。

複製快照 (記憶體資料庫 API)

若要複製快照，請使用具有下列參數的copy-snapshot作業：

參數

- SourceSnapshotName— 要複製的快照名稱。
- TargetSnapshotName— 快照副本的名稱。
- TargetBucket— 保留用於匯出快照。建立快照複本時，請勿使用此參數。如需詳細資訊，請參閱匯出快照。

下列範例會建立自動快照的複本。

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>
```

如需詳細資訊，請參閱CopySnapshot。

匯出快照

適用於 Redis 的 MemoryDB 支援將您的記憶體快照匯出至 Amazon Simple Storage Service (Amazon S3) 儲存貯體，讓您可以從外部 MemoryDB 存取快照。匯出的 MemoryDB 快照與開放原始碼 Redis 完全相容，而且可以使用適當的 Redis 版本或工具載入。您可以使用記憶體資料庫主控台、或 MemoryDB API 匯出快照集。AWS CLI

如果您需要在其他AWS區域啟動叢集，匯出快照會很有幫助。您可以匯出某個 AWS 區域中的資料、將 .rdb 檔案複製到新的 AWS 區域，然後使用該 .rdb 檔案植入新叢集，而不是等待透過使用來填入新叢集。如需植入新叢集的資訊，請參閱使用外部建立的快照植入新叢集。您也可能為了離線處理 .rdb 檔案，而想要匯出叢集資料。

Important

- 您要將其複製到的目的地的 MemoryDB 快照和 Amazon S3 儲存貯體必須位於同一AWS個區域。

  雖然複製到 Amazon S3 儲存貯體的快照已加密，但我們強烈建議您不要授與其他人存放快照之 Amazon S3 儲存貯體的存取權。
- 使用資料分層的叢集不支援將快照匯出到 Amazon S3。如需詳細資訊，請參閱資料分層。

在將快照匯出到 Amazon S3 儲存貯體之前，您必須在與快照相同的AWS區域中擁有一個 Amazon S3 儲存貯體。授與存儲桶的內存數據庫訪問權限。前兩個步驟示範如何執行此操作。

Warning

下列情況會以您不想要的方式公開資料：

- 當其他人可以存取您將快照匯出到的 Amazon S3 儲存貯體時。

  若要控制對快照的存取，請僅允許存取您要存取資料的對象存取 Amazon S3 儲存貯體。如需管理 Amazon S3 儲存貯體存取權的詳細資訊，請參閱 Amazon S3 開發人員指南中的管理存取權。
- 當其他人有使用 CopySnapshot API 操作的權限時。

  具有使用 `CopySnapshot` API 操作許可的使用者或群組可以建立自己的 Amazon S3 儲存貯體並將快照複製到他們。若要控制快照的存取權限，請使用 AWS Identity and Access

Management (IAM) 政策來控制誰有權使用 CopySnapshot API。如需有關使用 IAM 來控制使用 MemoryDB API 作業的詳細資訊，請參閱 MemoryDB 使用者指南適用於 Redis 的記憶體資料庫中的身分識別和存取管理中的〈〉。

主題

- 步驟 1：建立 Amazon S3 儲存貯體
- 步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體
- 步驟 3：匯出記憶體資料庫快照

步驟 1：建立 Amazon S3 儲存貯體

下列程序使用 Amazon S3 主控台建立 Amazon S3 儲存貯體，您可以在其中匯出和存放記憶體資料庫快照。

建立 Amazon S3 儲存貯體

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 選擇 Create Bucket (建立儲存貯體)。
3. 在 Create a Bucket - Select a Bucket Name and Region (建立儲存貯體 - 選取儲存貯體名稱和區域) 中，執行下列動作：
   a. 在 Bucket Name (儲存貯體名稱) 中，輸入 Amazon S3 儲存貯體的名稱。
   b. 從 Region (區域) 清單中為您的 Amazon S3 儲存貯體選擇 AWS 區域。此AWS區域必須與您要匯出的 MemoryDB 快照集所在的AWS區域相同。
   c. 選擇建立。

如需有關建立 Amazon S3 儲存貯體的詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的建立儲存貯體。

步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體

2019 年 3 月 20 日前推出的 AWS 區域預設為啟用。您可以立即開始使用這些 AWS 區域。2019 年 3 月 20 日之後引入的區域預設為停用狀態。您必須先啟用或選擇加入這些區域，才能使用這些區域，如管理AWS區域中所述。

授與某個區域中 S3 儲存貯體的記憶體資料庫存取權 AWS

若要在AWS區域中的 Amazon S3 儲存貯體上建立適當的許可，請執行下列步驟。

授與 S3 儲存貯體的記憶體資料庫存取權

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 選擇您要複製快照的目標 Amazon S3 儲存貯體的名稱。這應該是您在步驟 1：建立 Amazon S3 儲存貯體中建立的 S3 儲存貯體。
3. 選擇 [權限] 索引標籤，然後在 [權限] 底下選擇 [值區
4. 更新政策以授予 MemoryDB 執行操作所需的權限：

   - 將 `[ "Service" : "region-full-name.memorydb-snapshot.amazonaws.com" ]` 新增至 `Principal`。
   - 新增下列將快照匯出至 Amazon S3 儲存貯體所需的許可。
     - `"s3:PutObject"`
     - `"s3:GetObject"`
     - `"s3:ListBucket"`
     - `"s3:GetBucketAcl"`
     - `"s3:ListMultipartUploadParts"`
     - `"s3:ListBucketMultipartUploads"`

   以下是已更新政策可能有的外觀範例。

```
{
    "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/*"
            ]
        }
    ]
}
```

步驟 3：匯出記憶體資料庫快照

現在，您已經建立了 S3 儲存貯體，並授與 MemoryDB 許可以存取它。將 S3 物件擁有權變更為已啟用 ACL-儲存貯體擁有者偏好。接下來，您可以使用 MemoryDB 主控台、AWS CLI 或 MemoryDB API 將快照匯出至其中。以下假設您具備 S3 專屬的其他 IAM 許可。

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "s3:GetBucketLocation",
   "s3:ListAllMyBuckets",
   "s3:PutObject",
   "s3:GetObject",
   "s3:DeleteObject",
   "s3:ListBucket"
  ],
  "Resource": "arn:aws:s3:::*"
 }]
}
```

匯出記憶體資料庫快照 (主控台)

下列程序使用 MemoryDB 主控台將快照匯出到 Amazon S3 儲存貯體，以便您可以從 MemoryDB 外部存取快照。Amazon S3 儲存貯體必須與記憶體資料庫快照位於相同的AWS區域。

將記憶體資料庫快照匯出至 Amazon S3 儲存貯體

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看快照清單，請從左側導覽窗格中選擇 [快照]。
3. 從快照清單中，選擇要匯出之快照名稱左側的圓鈕。
4. 請選擇 Copy (複製)。
5. 在 Create a Copy of the Backup? (是否建立備份複本？) 中，執行下列動作：

   a. 在 [新增快照名稱] 方塊中，輸入新快照的名稱。

      該名稱必須介於 1 到 1,000 個字元之間，而且能夠以 UTF-8 編碼。

      MemoryDB 會新增分片識別碼，`.rdb`以及您在此處輸入的值。例如，如果您輸入`my-exported-snapshot`，MemoryDB 會建立。`my-exported-snapshot-0001.rdb`

   b. 從「目標 S3 位置」清單中，選擇您要將快照複製到其中的 Amazon S3 儲存貯體的名稱 (您在其中建立的儲存貯體步驟 1：建立 Amazon S3 儲存貯體)。

      目標 S3 位置必須是快照AWS區域中具有以下許可的 Amazon S3 儲存貯體，匯出程序才能成功。

      - 物件存取權 - Read (讀取) 和 Write (寫入)。
      - 許可存取權 - Read (讀取)。

      如需詳細資訊，請參閱步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體。

   c. 請選擇 Copy (複製)。

> **Note**
>
> 如果您的 S3 儲存貯體沒有 MemoryDB 將快照匯出到該儲存貯體所需的許可，您會收到下列其中一個錯誤訊息。返回步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體以新增指定的權限，然後重試匯出快照。
>
> - 尚未在 S3 儲存貯體上被授與 %s 的讀取權限。
>
>   解決方式：新增儲存貯體的 Read (讀取) 許可。
> - 尚未在 S3 儲存貯體上被授與寫入權限 %s。

解決方式：新增儲存貯體的 Write (寫入) 許可。

- 尚未被授與 S3 儲存貯體上的讀取 ACP 權限 %s。

  解決方式：新增儲存貯體的 Read (讀取) 許可存取。

如果要將快照複製到另一個AWS區域，請使用 Amazon S3 複製快照。如需詳細資訊，請參閱 Amazon 簡單儲存服務使用者指南中的複製物件。

匯出記憶體資料庫快照 (CLI) AWS

使用具有下列參數的 `copy-snapshot` CLI 操作將快照匯出到 Amazon S3 儲存貯體：

參數

- `--source-snapshot-name`— 要複製的快照名稱。
- `--target-snapshot-name`— 快照副本的名稱。

  該名稱必須介於 1 到 1,000 個字元之間，而且能夠以 UTF-8 編碼。

  MemoryDB 添加了一個分片標識符和`.rdb`您在這裡輸入的值。例如，如果您輸入`my-exported-snapshot`，MemoryDB 會建立。`my-exported-snapshot-0001.rdb`
- `--target-bucket`— 您要匯出快照的 Amazon S3 儲存貯體的名稱。系統會在指定的值區中建立快照副本。

  `--target-bucket`必須是快照AWS區域中具有下列許可的 Amazon S3 儲存貯體，匯出程序才能成功。

  - 物件存取權 - Read (讀取) 和 Write (寫入)。
  - 許可存取權 - Read (讀取)。

  如需詳細資訊，請參閱步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體。

以下操作將快照複製到我的 S3 桶。

若為 Linux、macOS 或 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
```

針對 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**
>
> 如果您的 S3 儲存貯體沒有 MemoryDB 將快照匯出到該儲存貯體所需的許可，您會收到下列其中一個錯誤訊息。返回步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體以新增指定的權限，然後重試匯出快照。
>
> - 尚未在 S3 儲存貯體上被授與 %s 的讀取權限。
>
>   解決方式：新增儲存貯體的 Read (讀取) 許可。
> - 尚未在 S3 儲存貯體上被授與寫入權限 %s。
>
>   解決方式：新增儲存貯體的 Write (寫入) 許可。
> - 尚未被授與 S3 儲存貯體上的讀取 ACP 權限 %s。
>
>   解決方式：新增儲存貯體的 Read (讀取) 許可存取。

如需詳細資訊，請參閱 AWS CLI 命令參考中的 `copy-snapshot`。

如果要將快照複製到另一個AWS區域，請使用 Amazon S3 副本。如需詳細資訊，請參閱 Amazon 簡單儲存服務使用者指南中的複製物件。

匯出記憶體資料庫快照 (記憶體資料庫 API)

使用具有這些參數的 `CopySnapshot` API 操作，將快照匯出到 Amazon S3 儲存貯體。

參數

- `SourceSnapshotName`— 要複製的快照名稱。
- `TargetSnapshotName`— 快照副本的名稱。

該名稱必須介於 1 到 1,000 個字元之間，而且能夠以 UTF-8 編碼。

MemoryDB 會新增分片識別碼，`.rdb`以及您在此處輸入的值。例如，如果您輸入 `my-exported-snapshot`，則會得到 `my-exported-snapshot-0001.rdb`。

- `TargetBucket`— 您要匯出快照的 Amazon S3 儲存貯體的名稱。系統會在指定的值區中建立快照副本。

  `TargetBucket`必須是快照AWS區域中具有下列許可的 Amazon S3 儲存貯體，匯出程序才能成功。

  - 物件存取權 - Read (讀取) 和 Write (寫入)。
  - 許可存取權 - Read (讀取)。

  如需詳細資訊，請參閱步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體。

下列範例會將自動快照複製到 Amazon S3 儲存貯體`my-s3-bucket`。

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**
>
> 如果您的 S3 儲存貯體沒有 MemoryDB 將快照匯出到該儲存貯體所需的許可，您會收到下列其中一個錯誤訊息。返回步驟 2：將記憶體資料庫存取權授予您的 Amazon S3 儲存貯體以新增指定的權限，然後重試匯出快照。

- 尚未在 S3 儲存貯體上被授與 %s 的讀取權限。

  解決方式：新增儲存貯體的 Read (讀取) 許可。
- 尚未在 S3 儲存貯體上被授與寫入權限 %s。

  解決方式：新增儲存貯體的 Write (寫入) 許可。
- 尚未被授與 S3 儲存貯體上的讀取 ACP 權限 %s。

  解決方式：新增儲存貯體的 Read (讀取) 許可存取。

如需詳細資訊，請參閱CopySnapshot。

如果要將快照複製到另一個AWS區域，請使用 Amazon S3 副本將匯出的快照複製到另一個AWS區域的 Amazon S3 儲存貯體。如需詳細資訊，請參閱 Amazon 簡單儲存服務使用者指南中的複製物件。

從快照還原

您可以隨時將資料從 MemoryDB 或 ElastiCache Redis .rdb 快照檔案還原至新叢集。

Redis 的恢復過程中的內存數據庫支持以下內容：

- 從您 ElastiCache 為 Redis 建立的一或多個 .rdb 快照檔案遷移至 MemoryDB 叢集。

  您必須將 .rdb 檔案放在 S3 中，才能執行還原。
- 在新叢集中指定一些碎片，這些碎片與叢集中用來建立快照檔案的碎片數目不同。
- 為新叢集指定不同節點類型 (大型或小型)。若要縮減為更小的節點類型，請確定新的節點類型有足夠的記憶體，可供您的資料和 Redis 額外負荷使用。
- 設定新 MemoryDB 叢集的插槽，與用來建立快照檔案的叢集不同。

> **Important**
>
> - 記憶體資料庫叢集不支援多個資料庫。因此，當還原至 MemoryDB 時，如果 .rdb 檔案參考多個資料庫，您的還原就會失敗。
> - 您無法從使用資料分層 (例如 r6gd 節點類型) 的叢集將快照還原到不使用資料分層 (例如，r6g 節點類型) 的叢集中。

從快照還原叢集時是否進行任何變更，都是由您所做的選擇所控制。使用 MemoryDB 主控台進行還原時，您可以在 [還原叢集] 頁面中進行這些選擇。您可以在使用AWS CLI或 MemoryDB API 進行還原時設定參數值來進行這些選擇。

在還原作業期間，MemoryDB 會建立新叢集，然後使用快照檔案中的資料填入叢集。完成此程序後，叢集就會預熱並準備好接受要求。

> **Important**
>
> 在繼續之前，請確定您已建立要從中還原的叢集快照。如需詳細資訊，請參閱製作手動快照。如果您要從外部建立的快照還原，請參閱使用外部建立的快照植入新叢集。

下列程序說明如何使用 MemoryDB 主控台、或 MemoryDB API 將快照還原至新叢集。AWS CLI

從快照還原 (主控台)

將快照還原到新叢集 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在功能窗格中，選擇 [快照]。
3. 在快照清單中，選擇您要還原的快照名稱旁邊的按鈕。
4. 選擇動作，然後選擇還原
5. 在 [叢集配置] 下，輸入下列內容：
   a. 叢集名稱 — 必要。新叢集的名稱。
   b. 說明 — 選用。新叢集的描述。
6. 完成「子網路群組」區段：
   - 對於子網路群組，請建立新的子網路群組，或從可用清單中選擇要套用至此叢集的現有子網路群組。如果您要創建一個新的：
     - 輸入名稱
     - 輸入說明
     - 如果您啟用多個可用區，子網路群組必須至少包含兩個位於不同可用區域的子網路。如需詳細資訊，請參閱子網路和子網路群組。
     - 如果您要建立新的子網路群組，但沒有現有的 VPC，系統會要求您建立 VPC。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。
7. 完成「叢集設定」區段：
   a. 如需 Redis 版本相容性，請接受預設值`6.0`。
   b. 對於連接埠，請接受 6379 的預設 Redis 連接埠，或者，如果您有理由使用不同的連接埠，請輸入連接埠號碼。
   c. 對於「參數群組」，請接受`default.memorydb-redis6`參數群組。

      參數群組可控制叢集的執行時間參數。如需參數群組的詳細資訊，請參閱雷迪斯特定參數。
   d. 針對節點類型，請選擇您想要的節點類型 (及其相關記憶體大小) 的值。

      如果您選擇 r6gd 節點類型系列的成員，您將自動啟用叢集中的資料分層。如需詳細資訊，請參閱資料分層。

e. 對於碎片數量，請選擇您要用於此叢集的碎片數目。

您可以動態變更叢集中的碎片數目。如需詳細資訊，請參閱擴展 MemoryDB 叢集。

f. 針對 Replicas per shard (每個碎片的複本)，選擇您要讓每個碎片具備的僅供讀取複本節點數目。

存在以下限制;。

- 如果您已啟用多個可用區，請確保每個碎片至少有一個複本。
- 使用主控台建立叢集時，每個碎片的複本數都相同。

g. 選擇 Next (下一步)

h. 完成「進階設定」區段：

i. 在 Security groups (安全群組) 中，選擇要用於此叢集的安全群組。安全群組可做為防火牆來控制叢集的網路存取。您可以為 VPC 使用預設安全群組，或建立新的安全群組。

如需安全群組的詳細資訊，請參閱 Amazon VPC 使用者指南中的 VPC 的安全群組。

ii. 資料的加密方式如下：

- Encryption at rest (靜態加密) - 啟用存放在磁碟上的資料加密功能。如需詳細資訊，請參閱靜態加密。

> **Note**
>
> 您可以選擇 Customer Managed AWS Master Key (客戶受管 AWS 主金鑰) 並選擇金鑰，以提供不同的加密金鑰。

- Encryption in-transit (傳輸中加密) - 啟用傳輸中資料加密功能。其預設為啟用。如需詳細資訊，請參閱傳輸中加密。

如果您選擇沒有加密，則將以默認用戶創建一個名為「開放訪問」的開放訪問控制列表。如需詳細資訊，請參閱使用存取控制清單 (ACL) 驗證使用者。

iii. 若為快照，請選擇性地指定快照保留期間和快照視窗。依預設，會選取 [啟用自動快照]。

iv. 針對 「維護」視窗，選擇性地指定維護時段 維護時段是 MemoryDB 每週為叢集排程系統維護的時間 (通常為 1 小時)。您可以允許 MemoryDB 選擇維護時段的日期和時間（無偏好），也可以自己選擇日期，時間和持續時間（指定維護時段）。如果您從清單中選

擇 Specify maintenance window (指定維護時段)，請為您的維護時段選擇 Start day (開始日)、Start time (開始時間) 和 Duration (持續時間)。所有時間均以 UCT 時間表示。

如需詳細資訊，請參閱管理維護作業。

v. 針對 Notifications (通知)，選擇現有的 Amazon Simple Notification Service (Amazon SNS) 主題，或選擇手動輸入 ARN，並輸入主題的 Amazon 資源名稱 (ARN)。Amazon SNS 可讓您推播通知到已與網際網路連線的智慧裝置。預設是停用通知。如需詳細資訊，請參閱 https://aws.amazon.com/sns/。

i. 對於標籤，您可以選擇性地套用標籤來搜尋和篩選叢集或追蹤AWS成本。

j. 檢閱所有項目和選項，然後進行任何所需的更正。準備就緒後，請選擇 Create cluster (建立叢集) 以啟動叢集，或 Cancel (取消) 取消操作。

一旦叢集的狀態變為可用，您就可以為其授予 EC2 存取權限、連線至叢集並開始使用叢集。如需更多詳細資訊，請參閱「步驟 2：授權對叢集的存取」及「步驟 3：Connect 到叢集」。

> **Important**
>
> 在您的叢集可用之後，系統就會按叢集作用中時間每個小時或部分小時計費 (即使您並未主動使用亦同)。若要停止此叢集產生費用，您必須將其刪除。請參閱 步驟 4：刪除叢集。

從快照 (AWSCLI) 還原

使用任一`create-cluster`作業時，請務必包含參數，`--snapshot-name`或`--snapshot-arns`將快照中的資料植入新叢集。

如需詳細資訊，請參閱下列內容：

- 建立叢集 (AWS CLI)在內存數據庫用戶指南中。
- 在命令參AWS CLI考中創建集群。

從快照還原 (記憶體資料庫 API)

您可以使用記憶體資料庫 API 作業還原記憶體資料庫快照集。`CreateCluster`

使用此`CreateCluster`作業時，請務必包含參數，`SnapshotName`或`SnapshotArns`將快照中的資料植入新叢集。

如需詳細資訊，請參閱下列內容：

- 建立叢集 (記憶體資料庫 API)在內存數據庫用戶指南中。
- CreateCluster在內存數據庫 API 參考中。

使用外部建立的快照植入新叢集

當您建立新的 MemoryDB 叢集時，您可以使用 Redis .rdb 快照集檔案中的資料植入叢集。

若要從 MemoryDB 快照集或 Redis 快照集植入新的記憶體資料庫叢集， ElastiCache 請參閱。從快照還原

當您使用 Redis .rdb 檔案植入新的 MemoryDB 叢集時，您可以執行下列動作：

- 指定新叢集中的碎片數目。此數目可以與叢集中用來建立快照檔案的碎片數目不同。
- 為新叢集指定不同的節點類型 — 大於或小於建立快照的叢集中使用的節點類型。若要縮減為更小的節點類型，請確定新的節點類型有足夠的記憶體，可供您的資料和 Redis 額外負荷使用。

> **Important**
>
> - 您必須確保快照資料不會超過節點的資源。
>
> 如果快照太大，則產生的叢集狀態為`restore-failed`。如果發生此情況，您必須刪除叢集並重新開始。
>
> 如需節點類型與規格的完整清單，請參閱內存數據庫節點類型的特定參數。
> - 您只能對使用 Amazon S3 伺服器端加密 (SSE-S3) 的 Redis .rdb 檔案進行加密。如需詳細資訊，請參閱使用伺服器端加密保護資料。

第 1 步：在外部集群上創建 Redis 快照

若要建立快照以植入您的 MemoryDB 叢集

1. 連線到您現有的 Redis 執行個體。
2. 運行 Redis 的`BGSAVE`或`SAVE`操作來創建一個快照。記下您的 .rdb 檔案位置。

   `BGSAVE` 是非同步的，不會封鎖其他用戶端的處理。如需詳細資訊，請參閱 Redis 網站上的 BGSAVE。

   `SAVE` 是同步的，並會封鎖其他處理序直到完成為止。如需詳細資訊，請參閱 Redis 網站上的 SAVE。

如需有關建立快照的其他資訊，請參閱 Redis 網站上的 Redis 持續性。

步驟 2：建立 Amazon S3 儲存貯體和資料夾

建立快照檔案後，您需要將其上傳到 Amazon S3 儲存貯體中的資料夾。若要執行此操作，您必須先擁有 Amazon S3 儲存貯體，且該儲存貯體中有資料夾。如果您已有具備適當許可的 Amazon S3 儲存貯體和資料夾，您可以跳到「步驟 3：將您的快照上傳到 Amazon S3」。

建立 Amazon S3 儲存貯體

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 依照 Amazon Simple Storage Service 使用者指南中的建立儲存貯體提供的指引操作，建立 Amazon S3 儲存貯體。

   Amazon S3 儲存貯體的名稱必須具 DNS 合規性。否則，MemoryDB 將無法訪問您的備份文件。DNS 合規的規則如下：

   - 名稱長度須為 3 到 63 個字元。
   - 名稱必須是一連串一或多個標籤，並以句號 (.) 分隔，其中每個標籤：
     - 以小寫字母或數字開頭。
     - 以小寫字母或數字結尾。
     - 僅包含小寫字母、數字和破折號。
   - 不得使用 IP 地址格式 (例如 192.0.2.0)。

   我們強烈建議您在新的 MemoryDB 叢集所在的相同AWS區域中建立 Amazon S3 儲存貯體。當 MemoryDB 從 Amazon S3 讀取您的 .rdb 檔案時，此方法可確保最高的資料傳輸速度。

   > **Note**
   >
   > 為了讓您的資料盡可能保持安全，請盡可能限制您 Amazon S3 儲存貯體的許可。同時，權限仍然需要允許存儲桶及其內容用於植入新的 MemoryDB 叢集。

在 Amazon S3 儲存貯體中新增資料夾

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。

2. 選擇您要上傳 .rdb 檔案的目的地儲存貯體名稱。
3. 選擇 Create folder (建立資料夾)。
4. 輸入您的新資料夾名稱。
5. 選擇儲存。

   記下儲存貯體名稱和資料夾名稱。

步驟 3：將您的快照上傳到 Amazon S3

現在，上傳您在第 1 步：在外部集群上創建 Redis 快照中建立的 .rdb 檔案 上傳到在 步驟 2：建立 Amazon S3 儲存貯體和資料夾 中建立的 Amazon S3 儲存貯體和資料夾。如需有關此工作的詳細資訊，請參閱上載物件。在步驟 2 到 3 之間，選擇您已建立的資料夾名稱。

將 .rdb 檔案上傳到 Amazon S3 資料夾

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 選擇您在步驟 2 中建立的 Amazon S3 儲存貯體名稱。
3. 選擇您在步驟 2 中建立的資料夾名稱。
4. 選擇上傳。
5. 選擇 Add files (新增檔案)。
6. 瀏覽至您要上傳的一或多個檔案，然後選擇一或多個檔案。若要選擇多個檔案，請按住 Ctrl 鍵並選擇每個檔案名稱。
7. 選擇 Open (開啟)。
8. 確認 [上傳] 頁面中列出的一或多個檔案是否正確，然後選擇 [上傳]。

記下 .rdb 檔案的路徑。例如，如果您的儲存貯體名稱為 `myBucket` 且路徑為 `myFolder/redis.rdb`，請輸入 `myBucket/myFolder/redis.rdb`。您需要此路徑才能使用此快照中的資料植入新叢集。

如需其他資訊，請參閱 Amazon 簡單儲存服務使用者指南中的儲存貯體命名規則。

步驟 4：授與 .rdb 檔案的記憶體讀取權限

2019 年 3 月 20 日前推出的 AWS 區域預設為啟用。您可以立即開始使用這些 AWS 區域。2019 年 3 月 20 日之後引入的區域預設為停用狀態。您必須先啟用或選擇加入這些區域，才能使用這些區域，如管理AWS區域中所述。

授與 .rdb 檔案的記憶體資料庫讀取存取權

若要將 MemoryDB 讀取存取權授與快照檔案

1. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 選擇包含您 .rdb 檔案的 S3 儲存貯體名稱。
3. 選擇包含您 .rdb 檔案的資料夾名稱。
4. 選擇 .rdb 快照檔案的名稱。所選檔案的名稱將會顯示在頁面頂端的標籤上方。
5. 選擇 Permissions (許可) 索引標籤標籤。
6. 在 Permissions (許可) 中，選擇 Bucket policy (儲存貯體政策)，然後選擇 Edit (編輯)。
7. 更新政策以授予 MemoryDB 執行操作所需的權限：

   - 將 `[ "Service" : "region-full-name.memorydb-snapshot.amazonaws.com" ]` 新增至 `Principal`。
   - 新增下列將快照匯出至 Amazon S3 儲存貯體所需的許可：
     - `"s3:GetObject"`
     - `"s3:ListBucket"`
     - `"s3:GetBucketAcl"`

   以下是已更新政策可能有的外觀範例。

```
{
    "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. 選擇儲存。

第 5 步：使用 .rdb 文件數據種子內存數據庫集群

現在您已經準備好建立 MemoryDB 叢集，並使用 .rdb 檔案中的資料植入叢集。若要建立叢集，請遵循中的指示建立記憶體資料庫叢集。

您用來告訴 MemoryDB 在哪裡可以找到您上傳到 Amazon S3 的 Redis 快照的方法取決於您用來建立叢集的方法：

使用 .rdb 檔案資料植入記憶體資料庫叢集

- 使用記憶體資料庫主控台

  選擇 Redis 引擎之後，展開 Advanced Redis settings (進階 Redis 設定) 區段，並找到 Import data to cluster (將資料匯入叢集)。在 Seed RDB file S3 location (植入 RDB 檔案 S3 位置) 方塊中，輸入檔案的 Amazon S3 路徑。如果您有多個 .rdb 檔案，請以逗號分隔清單輸入每個檔案的路徑。Amazon S3 路徑看起來像 *myBucket*/*myFolder*/*myBackupFilename*.rdb。

- 使用 AWS CLI

  如果您使用 `create-cluster` 或 `create-cluster` 操作，請使用參數 `--snapshot-arns` 來指定每個 .rdb 檔案的完整 ARN。例如 `arn:aws:s3:::`*myBucket*/*myFolder*/*myBackupFilename*.rdb。ARN 必須解析為您存放在 Amazon S3 中的快照檔案。

- 使用記憶體資料庫 API

如果您使用`CreateCluster`或 `CreateCluster` MemoryDB API 作業，請使用參數`SnapshotArns`為每個 .rdb 檔案指定完整的 ARN。例如 `arn:aws:s3:::myBucket/myFolder/myBackupFilename.rdb`。ARN 必須解析為您存放在 Amazon S3 中的快照檔案。

在建立叢集的過程中，快照中的資料會寫入叢集。您可以透過檢視 MemoryDB 事件訊息來監視進度。要做到這一點，請參閱 MemoryDB 控制台，然後選擇事件。您也可以使用 AWS MemoryDB 命令列介面或記憶體資料庫 API 來取得事件訊息。

標記快照

您可以將自己的中繼資料以標記的形式指派給每個快照。標籤可讓您以不同的方式對快照進行分類，例如，依目的、擁有者或環境。當您有許多相同類型的資源時，這將會很有用，因為—您可以依據先前指派的標籤，快速識別特定的資源。如需詳細資訊，請參閱您可以標記的資源。

成本分配標籤可讓您依標籤值分組發票上的費用，來追蹤多個 AWS 服務中的成本。若要進一步了解成本分配標籤，請參閱使用成本分配標籤。

使用 MemoryDB 主控台、或 MemoryDB APIAWS CLI，您可以在快照上新增、列出、修改、移除或複製成本分配標籤。如需詳細資訊，請參閱使用成本配置標籤監控成本。

刪除快照

自動快照會在其保留限制到期時自動刪除。如果您刪除叢集，其所有自動快照也會一併刪除。

MemoryDB 提供刪除 API 操作，可讓您隨時刪除快照，無論快照是自動還是手動創建。由於手動快照沒有保留限制，因此手動刪除是移除快照的唯一方法。

您可以使用 [記憶體資料庫] 主控台、或 [記憶體資料庫 API] 刪除快照集。AWS CLI

刪除快照（控制台）

下列程序會使用 MemoryDB 主控台刪除快照集。

刪除快照

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [快照]。

   會出現「快照」畫面，其中包含您的快照清單。
3. 選擇要刪除之快照名稱左側的圓鈕。
4. 選擇 Actions (動作)，然後選擇 Delete (刪除 VPC)。
5. 如果您要刪除此快照，請delete在文字方塊中輸入，然後選擇 [刪除]。若要取消刪除，請選擇「取消」。狀態會變更為「刪除中」。

刪除快照 (AWSCLI)

使用刪除快照AWS CLI作業搭配下列參數來刪除快照。

- `--snapshot-name`— 要刪除的快照名稱。

下列程式碼會刪除快照myBackup。

```
aws memorydb delete-snapshot --snapshot-name myBackup
```

如需詳細資訊，請參閱 AWS CLI 命令參考中的 delete-snapshot。

刪除快照 (記憶體資料庫 API)

使用 DeleteSnapshot API 作業搭配下列參數來刪除快照。

- SnapshotName— 要刪除的快照名稱。

下列程式碼會刪除快照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>
```

如需詳細資訊，請參閱DeleteSnapshot。

擴展

您的應用程式需要處理的資料量通常是動態的。它會隨著您的業務成長或遇到需求的一般波動而增加或減少。如果您自行管理應用程式，則需要佈建足夠的硬體來滿足您的需求峰值，這可能很昂貴。使用 MemoryDB to Redis，您可以擴展以滿足目前的需求，只需為您使用的部分付費。

下列各項能幫助您找到用於您要執行之擴展動作的正確主題。

擴展 MemoryDB

動作	memoryDB
向外擴展	MemoryDB 的線上重新分片和碎片重新平衡功能
變更節點類型	透過修改節點類型來進行線上垂直擴展
變更碎片數	擴展 MemoryDB 叢集

擴展 MemoryDB 叢集

隨著對叢集的需求變更，您可能會決定透過變更在 MemoryDB 叢集中碎片的數量，來改善效能或減少成本。我們建議使用線上水平擴展來執行此動作，因為它可允許叢集在擴展程序期間繼續提供請求的服務。

您用來決定重新擴展叢集的可能條件包括下列：

- 記憶體壓力：

  如果叢集中的節點遭受記憶體壓力，您可以決定向外擴展，使得您有更多資源能更妥善地存放資料和提供請求的服務。

  您可以監控下列指標，判斷節點是否處於記憶體壓力之下：FreeableMemorySwapUsage、和 BytesUsedForMemoryDB。
- CPU 或網路瓶頸：

  如果延遲/傳輸量問題正困擾著您的叢集，您可能需要向外擴展來解決問題。

  您可以監控下列指標來監控延遲和輸送量層級：CPUR ISER NetworkBytesInNetworkBytesOut、CurrConnections、、和NewConnections。
- 您的叢集過度擴展：

  對叢集的目前需求使得向內擴展不會傷害效能和減少成本。

  您可以監視叢集的使用情況，以判斷是否可以使用下列指標安全地調整規模：FreeableMemorySwapUsage、、BytesUsedForMemoryDB、CPUR urse、NetworkBytesIn、NetworkBytesOut、CurrConnections，和NewConnections。

擴展的效能影響

使用離線程序擴展時，您的叢集將有一大部分程序會離線，因此無法提供請求的服務。使用線上方法擴展時，因為擴展是運算密集的操作，效能會有一些下降，然後，您的叢集會繼續在整個擴展操作中提供請求的服務。您遭遇到的下降程度取決於您的一般 CPU 使用率和您的資料。

擴展 MemoryDB 叢集有兩種方式；水平和垂直擴展。

- 水平縮放允許您通過添加或刪除碎片來更改集群中的碎片數量。線上重新分片程序允許向內/向外擴展，同時間叢集仍可繼續服務傳入請求。

- 垂直擴展 - 變更節點類型以調整叢集大小。線上垂直擴展允許向上/向下擴展，同時間叢集仍可繼續服務傳入請求。

如果您要透過向內擴展或向下擴展來降低叢集的大小和記憶體容量，請確定新的組態有足夠的記憶體，可供您的資料和 Redis 額外負荷使用。

MemoryDB 的離線重新分片和碎片重新平衡功能

您從離線碎片重新配置中獲得的主要優點是，您不僅可以從叢集中添加或刪除碎片，還可以執行更多操作。當您離線重新分片時，除變更叢集中的碎片數目之外，您還可以執行下列動作：

- 變更叢集的節點類型。
- 升級至較新的引擎版本。

> **Note**
>
> 啟用資分層的叢集不支援離線重新分片。如需詳細資訊，請參閱 資料分層。

離線碎片重新組態的主要缺點是，您的叢集會離線開始進行程序的還原部分，並繼續直到您在應用程式中更新端點為止。叢集離線的時間長度會因叢集中資料量而不同。

若要離線重新設定您的碎片記憶體 DB 叢集

1. 建立現有 MemoryDB 叢集的手動快照。如需詳細資訊，請參閱 製作手動快照。
2. 透過從快照還原來建立新叢集。如需詳細資訊，請參閱 從快照還原。
3. 在應用程式中，將端點更新為新叢集的端點。如需詳細資訊，請參閱 尋找連線端點。

MemoryDB 的線上重新分片和碎片重新平衡功能

透過使用線上重新分片和碎片重新平衡搭配 MemoryDB，您可以動態擴展 MemoryDB 而不會發生停機。此方法表示叢集可以繼續提供請求的服務 (甚至是在擴展或重新平衡進行中時)。

您可以執行下列作業：

- 向外擴充 — 透過將碎片新增至 MemoryDB 叢集來增加讀取和寫入容量。

  如果您將一或多個碎片新增至叢集，則每個新碎片中的節點數目與現有碎片中最小的節點數目相同。

- 擴充規模-透過從 MemoryDB 叢集移除碎片來減少讀取和寫入容量，進而降低成本。

目前，下列限制適用於 MemoryDB 線上重新分片：

- 槽或金鑰空間和大型項目的限制為：

  如果碎片中的任何金鑰包含大型項目，在向外擴展或重新平衡時，不會將該金鑰遷移至新碎片。此功能可能造成不平衡的碎片。

  如果碎片中的任何金鑰包含大型項目 (序列化後項目大於 256 MB)，在向內擴展時，不會刪除該碎片。此功能可能造成一些碎片不會遭到刪除。
- 縮小時，任何新碎片中的節點數量等於現有碎片中的節點數量。

如需詳細資訊，請參閱 最佳實務：線上叢集大小調整。

您可以使用AWS Management Console、和 MemoryDB API，為 MemoryDB 叢集進行AWS CLI水平擴展或重新平衡。

使用線上重新分片功能新增碎片

您可以使用AWS Management Console、AWS CLI或 MemoryDB API 來將碎片新增至 MemoryDB 叢集。

新增碎片 (主控台)

您可以使用AWS Management Console來將一或多個碎片新增至 MemoryDB 叢集。下列程序描述該程序。

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從叢集清單中選擇您要新增碎片的目標叢集名稱。
3. 在「碎片和節點」選項卡下，選擇「添加/刪除碎片」
4. 在新的碎片數量中，輸入您想要的碎片數量。
5. 選擇「確認」以保留變更，或選擇「取消」 捨棄。

新增碎片 (AWS CLI)

下列程序說明如何透過使用新增碎片來重新設定 MemoryDB 叢集中的碎片AWS CLI。

使用下列參數搭配 update-cluster。

參數

- --cluster-name - 必要項目。指定要在哪個叢集 (叢集) 上執行碎片重新配置作業。
- --shard-configuration - 必要項目。允許您設定碎片數。
  - ShardCount-設定此屬性以指定您要的碎片數。

Example

下列範例會將叢集中的碎片數目修改my-cluster為 2。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --shard-configuration \
        ShardCount=2
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --shard-configuration ^
        ShardCount=2
```

它會傳回下列 JSON 回應：

```
{
    "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
    }
}
```

若要在已更新叢集的狀態從更新變更為可用時檢視其詳細資訊，請使用下列命令：

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

它將返回以下 JSON 響應：

```
{
    "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
        }
    ]
}
```

如需詳細資訊，請參閱AWS CLI命令參考中的更新叢集。

新增碎片 (MemoryDB API)

您可以使用 MemoryDB API，透過使用此UpdateCluster作業，在 MemoryDB 叢集中線上重新設定碎片。

使用下列參數搭配 UpdateCluster。

參數

- `ClusterName` - 必要項目。指定要在哪個叢集上執行碎片重新配置作業。
- `ShardConfiguration` - 必要項目。允許您設定碎片數。
  - `ShardCount`-設定此屬性以指定您要的碎片數。

如需詳細資訊，請參閱UpdateCluster。

使用線上重新分片移除碎片

您可以使用AWS Management Console、AWS CLI或 MemoryDB API 來移除 MemoryDB 叢集中的碎片。

移除碎片 (主控台)

下列程序說明如何透過使用移除碎片來重新設定 MemoryDB 叢集中的碎片AWS Management Console。

> **Important**
>
> 從叢集移除碎片之前，MemoryDB 會確定所有資料可納入其餘碎片中。如果數據適合，則會根據要求從叢集中刪除碎片。如果資料不適合剩餘的碎片，則會終止處理序，並且叢集會保留與發出要求之前相同的碎片配置。

您可以使用AWS Management Console來從 MemoryDB 叢集移除一或多個碎片。您無法移除叢集中的所有碎片。相反的，您必須刪除叢集。如需詳細資訊，請參閱 步驟 4：刪除叢集。下列程序說明移除一或多個碎片的程序。

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從叢集清單中選擇您要從中移除碎片的目標叢集名稱。
3. 在「碎片和節點」選項卡下，選擇「添加/刪除碎片」
4. 在新的碎片數量中，輸入您想要的碎片數量（至少為 1）。
5. 選擇「確認」以保留變更，或選擇「取消」 捨棄。

移除碎片 (AWS CLI)

下列程序說明如何透過使用移除碎片來重新設定 MemoryDB 叢集中的碎片AWS CLI。

> **Important**
>
> 從叢集移除碎片之前，MemoryDB 會確定所有資料可納入其餘碎片中。如果數據適合，碎片將根據要求從集群中刪除，並將其密鑰空間映射到剩餘的碎片中。如果數據將不適合在剩餘的碎片，該進程被終止，並保留與發出請求之前相同的碎片配置集群。

您可以使用AWS CLI來從 MemoryDB 叢集移除一或多個碎片。您無法移除叢集中的所有碎片。相反的，您必須刪除叢集。如需詳細資訊，請參閱 步驟 4：刪除叢集。

使用下列參數搭配 `update-cluster`。

參數

- `--cluster-name` - 必要項目。指定要在哪個叢集 (叢集) 上執行碎片重新配置作業。
- `--shard-configuration` - 必要項目。允許您使用`ShardCount`屬性設置碎片的數量：

  `ShardCount`-設定此屬性以指定您要的碎片數。

Example

下列範例會將叢集中的碎片數目修改`my-cluster`為 2。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --shard-configuration \
        ShardCount=2
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --shard-configuration ^
        ShardCount=2
```

它會傳回下列 JSON 回應：

```
{
    "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
    }
}
```

若要在已更新叢集的狀態從更新變更為可用時檢視其詳細資訊，請使用下列命令：

若為 Linux、macOS 或 Unix：

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

針對 Windows：

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

它將返回以下 JSON 響應：

```
{
    "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
        }
    ]
}
```

如需詳細資訊，請參閱AWS CLI命令參考中的更新叢集。

移除碎片 (MemoryDB API)

您可以使用 MemoryDB API，透過使用此UpdateCluster作業，在 MemoryDB 叢集中線上重新設定碎片。

下列程序說明如何透過使用 MemoryDB API 移除碎片來重新設定 MemoryDB 叢集中的碎片。

> **Important**
>
> 在移除碎片 rom 叢集之前，MemoryDB 會確定所有資料可納入其餘碎片中。如果數據適合，碎片將根據要求從集群中刪除，並將其密鑰空間映射到剩餘的碎片中。如果數據將不適合在剩餘的碎片，該進程被終止，並保留與發出請求之前相同的碎片配置集群。

您可以使用 MemoryDB API 來從 MemoryDB 叢集移除一或多個碎片。您無法移除叢集中的所有碎片。相反的，您必須刪除叢集。如需詳細資訊，請參閱 步驟 4：刪除叢集。

使用下列參數搭配 UpdateCluster。

參數

- ClusterName - 必要項目。指定要在哪個叢集 (叢集) 上執行碎片重新配置作業。
- ShardConfiguration - 必要項目。允許您使用ShardCount屬性設置碎片的數量：

  ShardCount-設定此屬性以指定您要的碎片數。

透過修改節點類型來進行線上垂直擴展

使用線上垂直擴展配 MemoryDB，您可以動態擴展叢集，只需最低限度的停機。這可讓您的叢集即使在擴展時也能提供要求。

> **Note**
>
> 不支援在資料分層叢集 (例如，使用 r6gd 節點類型的叢集) 與未使用資料分層叢集 (例如，使用 r6g 節點類型的叢集) 之間的擴展。如需詳細資訊，請參閱 資料分層。

您可以執行下列作業：

- 擴充規模-將 MemoryDB 叢集的節點類型調整為使用大型節點類型，增加讀取和寫入容量。

  MemoryDB 會動態調整叢集大小，同時保持線上狀態並服務請求。
- 縮減規模 - 將節點類型向下調整為使用較小的節點，減少讀取和寫入容量。同樣地，MemoryDB 會動態調整叢集大小，同時保持線上狀態並服務請求。在這種情況下，您透過縮減節點來降低成本。

> **Note**
>
> 向上擴展和向下擴展程序牽涉到使用新選取的節點類型來建立叢集，並將新節點與先前的節點進行同步。若要確保順暢的向上/向下擴展流程，請執行以下操作：
>
> - 雖然垂直擴展程序的目標是保持全面上線，但此程序也需要在舊節點和新節點之間同步資料。建議您在預期資料流量最小的時間內啟動向上/向下擴展。
> - 盡可能在預備環境中測試您的應用程式行為。

線上擴充規模

主題

- 為 MemoryDB 叢集擴充規模 (主控台)
- 擴展記憶體資料庫叢集 (AWSCLI)
- 擴展記憶體資料庫叢集 (記憶體資料庫 API)

為 MemoryDB 叢集擴充規模 (主控台)

下列程序說明如何使用來為 MemoryDB 叢集擴充規模AWS Management Console。在此過程中，您的 MemoryDB 叢集將以最少的停機時間繼續為請求提供服務。

為叢集擴充規模 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從叢集的清單中，選擇叢集。
3. 選擇 Actions (動作)，然後選擇 Modify (修改)。
4. 在「修改叢集」對話方塊中：
   - 從 Node type (節點類型) 清單選擇您要擴展的節點類型。若要向上擴展，請選取大於現有節點的節點類型。
5. 選擇 Save changes (儲存變更)。

   叢集的狀態會變更為「修改」。當狀態變更為 available (可用)，修改即已完成，並且您可以開始使用新叢集。

擴展記憶體資料庫叢集 (AWSCLI)

下列程序說明如何使用來為 MemoryDB 叢集擴充規模AWS CLI。在此過程中，您的 MemoryDB 叢集將以最少的停機時間繼續為請求提供服務。

為 MemoryDB 叢集擴充規模 (AWSCLI)

1. 判斷您可以向上擴展的節點類型，方法是執行 AWS CLI `list-allowed-node-type-updates` 命令搭配下列參數。

   若為 Linux、macOS 或 Unix：

   ```
   aws memorydb list-allowed-node-type-updates \
        --cluster-name my-cluster-name
   ```

   針對 Windows：

   ```
   aws memorydb list-allowed-node-type-updates ^
        --cluster-name my-cluster-name
   ```

   上述命令的輸出看起來會類似這個 (JSON 格式)。

   ```
   {
        "ScaleUpNodeTypes": [
   ```

```
        "db.r6g.2xlarge",
        "db.r6g.large"
    ],
    "ScaleDownNodeTypes": [
        "db.r6g.large"
    ],
}
```

如需詳細資訊，請參閱AWS CLI參考中的 list-allowed-node-type-updates。

2. 使用AWS CLIupdate-cluster命令和下列參數來修改叢集以擴充規模為較大的新節點類型。

   - `--cluster-name`-您要擴充規模的目標叢集名稱。
   - `--node-type`-您要擴展叢集規模的新節點類型。此值必須是步驟 1 中 `list-allowed-node-type-updates` 命令傳回的其中一個節點類型。

   若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster  \
 --cluster-name my-cluster \
 --node-type db.r6g.2xlarge
```

   針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --node-type db.r6g.2xlarge ^
```

   如需詳細資訊，請參閱更新叢集。

擴展記憶體資料庫叢集 (記憶體資料庫 API)

下列程序會使用 MemoryDB API，將叢集從其目前的節點類型擴展至較大的新節點類型。在此程序中，MemoryDB 會更新 DNS 項目，使它們指向新的節點。您可以擴展啟用自動容錯移轉的叢集，同時繼續保持線上狀態並回應傳入請求。

擴充至較大節點類型所需的時間會因節點類型和目前叢集中的資料量而有所不同。

為 MemoryDB 叢集擴充規模 (MemoryDB API)

1. 使用 MemoryDB APIListAllowedNodeTypeUpdates 動作搭配下列參數，判斷您可以擴充規模的節點類型。

   - ClusterName— 叢集的名稱。使用此參數可描述特定叢集而非所有叢集。

   ```
   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>
   ```

   如需詳細資訊，請參閱記憶體資料庫ListAllowedNodeTypeUpdates中的 Redis API 參考。

2. 使用UpdateCluster MemoryDB API 動作搭配下列參數，將您目前的叢集擴充規模為新的節點類型。

   - ClusterName— 叢集的名稱。
   - NodeType-此叢集中較大的新叢集節點類型。此值必須是步驟 1 中 ListAllowedNodeTypeUpdates 動作傳回的其中一個執行個體類型。

   ```
   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>
   ```

如需詳細資訊，請參閱UpdateCluster。

線上縮減規模

主題

- 為 MemoryDB 叢集縮減規模 (主控台)
- 縮減記憶體資料庫叢集 (AWSCLI)
- 縮減記憶體資料庫叢集 (記憶體資料庫 API)

為 MemoryDB 叢集縮減規模 (主控台)

下列程序說明如何使用縮減規模AWS Management Console。在此過程中，您的 MemoryDB 叢集將以最少的停機時間繼續為請求提供服務。

為 MemoryDB 叢集縮減規模 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 從叢集的清單中，選擇您偏好的叢集。
3. 選擇 Actions (動作)，然後選擇 Modify (修改)。
4. 在「修改叢集」對話方塊中：
   - 從 Node type (節點類型) 清單選擇您要擴展的節點類型。若要向下擴展，請選取小於現有節點的節點類型。請注意，並非所有節點類型都可縮減規模。
5. 選擇 Save changes (儲存變更)。

   叢集的狀態會變更為「修改」。當狀態變更為 available (可用)，修改即已完成，並且您可以開始使用新叢集。

縮減記憶體資料庫叢集 (AWSCLI)

下列程序說明如何使用縮減規模AWS CLI。在此過程中，您的 MemoryDB 叢集將以最少的停機時間繼續為請求提供服務。

為 MemoryDB 叢集縮減規模 (AWSCLI)

1. 判斷您可以向下擴展的節點類型，方法是執行 AWS CLI `list-allowed-node-type-updates` 命令搭配下列參數。

   若為 Linux、macOS 或 Unix：

   ```
   aws memorydb list-allowed-node-type-updates \
        --cluster-name my-cluster-name
   ```

   針對 Windows：

   ```
   aws memorydb list-allowed-node-type-updates ^
        --cluster-name my-cluster-name
   ```

   上述命令的輸出看起來會類似這個 (JSON 格式)。

   ```
   {
        "ScaleUpNodeTypes": [
            "db.r6g.2xlarge",
            "db.r6g.large"
        ],
        "ScaleDownNodeTypes": [
            "db.r6g.large"
        ],
   }
   ```

   有關更多信息，請參見 list-allowed-node-type-更新。

2. 使用`update-cluster`命令和下列參數，修改叢集以縮減為新的較小節點類型。

   - `--cluster-name`-您要縮減規模的目標叢集名稱。
   - `--node-type`-您要擴展叢集規模的新節點類型。此值必須是步驟 1 中 `list-allowed-node-type-updates` 命令傳回的其中一個節點類型。

   若為 Linux、macOS 或 Unix：

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

```
    --node-type db.r6g.large
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --node-type db.r6g.large
```

如需詳細資訊，請參閱更新叢集。

縮減記憶體資料庫叢集 (記憶體資料庫 API)

下列程序會使用 MemoryDB API，將叢集從其目前的節點類型擴展至較小的新節點類型。在此過程中，您的 MemoryDB 叢集將以最少的停機時間繼續為請求提供服務。

縮減到較小節點類型所需的時間會因節點類型和目前叢集中的資料量而有所不同。

縮減規模 (MemoryDB API)

1. 使用 ListAllowedNodeTypeUpdatesAPI 搭配下列參數，判斷您可以縮減規模的節點類型：

   - `ClusterName`— 叢集的名稱。使用此參數可描述特定叢集而非所有叢集。

```
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. 使用 UpdateClusterAPI 搭配下列參數，將您目前的叢集縮減規模為新節點類型。

   - `ClusterName`— 叢集的名稱。
   - `NodeType`-此叢集中較小的新叢集節點類型。此值必須是步驟 1 中 `ListAllowedNodeTypeUpdates` 動作傳回的其中一個執行個體類型。

```
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>
```

使用參數群組設定引擎參數

適用於 Redis 的 MemoryDB 使用參數來控制節點和叢集的執行階段屬性。一般而言，更新的引擎版本會包含額外參數，可支援更新的功能。如需參數表，請參閱 雷迪斯特定參數。

如您所預期的，有些參數值 (例如 maxmemory) 會由引擎與節點類型決定。如需根據節點類型的參數值表，請參閱 內存數據庫節點類型的特定參數。

主題

- 參數管理
- 參數群組層
- 建立參數群組
- 依名稱列出參數群組
- 列出參數群組的值
- 修改參數群組
- 刪除參數群組
- 雷迪斯特定參數

參數管理

參數會群組成具名參數群組，簡化參數管理。參數群組代表在啟動期間傳遞給引擎軟體的參數特定值組合。這些值會決定每個節點上的引擎程序在執行時間的行為。特定參數群組上的參數值會套用到所有與群組相關聯的節點，無論節點所屬的叢集為何。

若要調整您叢集的效能，您可以修改一部分的參數值，或是變更叢集的參數群組。

- 您無法修改或刪除預設參數群組。若您需要自訂參數值，您必須建立自訂參數群組。
- 參數群組系列及您指派該群組的對象叢集必須相容。例如，如果您的叢集執行 Redis 第 6 版，您只能使用 memorydb_redis6 系列中的參數群組，預設或自訂參數群組。
- 當您變更叢集的參數時，變更會立即套用至叢集。無論是變更叢集的參數群組本身或是叢集的參數群組內的參數值，均適用此情況。

參數群組層

適用於 Redis 參數群組層的記憶體資料庫

全域預設

適用於該地區中 Redis 客戶的所有 MemoryDB 的最上層根參數群組。

全域預設參數群組：

- 保留給記憶體資料庫，而且不提供給客戶使用。

客戶預設

為客戶使用而建立的「全域預設」參數群組副本。

「客戶預設值」參數群組：

- 由內存數據庫創建和擁有。
- 對於執行此參數群組支援之引擎版本的任何叢集，客戶均可用作參數群組。
- 客戶無法對其進行編輯。

客戶自有

客戶預設參數群組的副本。每當客戶建立參數群組時，都會建立「客戶擁有」參數群組。

「客戶擁有」參數群組：

- 由客戶建立及擁有。
- 可指派給任何客戶的相容叢集。
- 客戶可以修改以建立自訂參數群組。

  並非所有參數值皆可修改。如需詳細資訊，請參閱 雷迪斯特定參數。

建立參數群組

若您希望修改不同於預設值的一或多個參數值，便需要建立新的參數群組。您可以使用 MemoryDB 主控台、或 MemoryDB API 來建立參數群組。 AWS CLI

建立參數群組 (主控台)

下列程序顯示如何使用 MemoryDB 主控台建立參數群組。

若要使用 MemoryDB 主控台建立參數群組

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看所有可用參數群組的清單，請從左側的導覽窗格中，選擇 Parameter Groups (參數群組)。
3. 若要建立參數群組，請選擇「建立參數群組」。

   便會顯示「建立參數群組」頁面。
4. 在 Name (名稱) 方塊中，輸入此參數群組的唯一名稱。

   建立叢集或修改叢集的參數群組時，您便會根據其名稱選擇參數群組。因此，我們建議選擇附帶資訊且能以某種方式識別參數群組系列的名稱。

   參數群組命名限制條件如下：

   - 必須以 ASCII 字母開頭。
   - 它只能包含 ASCII 字母、數字和連字符號。
   - 長度必須介於 1 至 255 個字元之間。
   - 不能連續包含兩個連字號。
   - 結尾不能是連字號。
5. 在 Description (描述) 方塊中，輸入參數群組的描述。
6. 在 Redis 版本相容性方塊中，選擇此參數群組對應的引擎版本。
7. 在「標籤」中，選擇性地新增標籤以搜尋和篩選參數群組或追蹤 AWS 成本。
8. 若要建立參數群組，請選擇 Create (建立)。

   若要終止程序而不建立參數群組，請選擇 Cancel (取消)。
9. 建立參數群組時，它會擁有系列的預設值。若要變更預設值，您必須修改參數群組。如需詳細資訊，請參閱 修改參數群組。

建立參數群組 (AWS CLI)

若要使用建立參數群組 AWS CLI，請`create-parameter-group`搭配這些參數使用指令。

- `--parameter-group-name` - 參數群組的名稱。

  參數群組命名限制條件如下：

  - 必須以 ASCII 字母開頭。
  - 它只能包含 ASCII 字母、數字和連字符號。
  - 長度必須介於 1 至 255 個字元之間。
  - 不能連續包含兩個連字號。
  - 結尾不能是連字號。
- `--family` - 參數群組的引擎和版本系列。
- `--description` - 使用者提供的參數群組說明。

Example

下列範例會使用記憶庫 _redis6 族群做為樣板，建立名為 myRedis6x 的參數群組。

若為 Linux、macOS 或 Unix：

```
aws memorydb create-parameter-group \
    --parameter-group-name myRedis6x  \
    --family memorydb_redis6 \
    --description "My first parameter group"
```

針對 Windows：

```
aws memorydb create-parameter-group ^
    --parameter-group-name myRedis6x  ^
    --family memorydb_redis6 ^
    --description "My first parameter group"
```

此命令的輸出看起來會與以下內容相似。

```
{
    "ParameterGroup": {
        "Name": "myRedis6x",
        "Family": "memorydb_redis6",
        "Description": "My first parameter group",
        "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/myredis6x"
    }
```

```
}
```

建立參數群組時，它會擁有系列的預設值。若要變更預設值，您必須修改參數群組。如需詳細資訊，請參閱 修改參數群組。

如需詳細資訊，請參閱 create-parameter-group。

建立參數群組 (記憶體資料庫 API)

若要使用 MemoryDB API 建立參數群組，請搭配這些參數使用CreateParameterGroup動作。

- ParameterGroupName - 參數群組的名稱。

  參數群組命名限制條件如下：

  - 必須以 ASCII 字母開頭。
  - 它只能包含 ASCII 字母、數字和連字符號。
  - 長度必須介於 1 至 255 個字元之間。
  - 不能連續包含兩個連字號。
  - 結尾不能是連字號。
- Family - 參數群組的引擎和版本系列。例如 memorydb_redis6。
- Description - 使用者提供的參數群組說明。

Example

下列範例會使用記憶庫 _redis6 族群做為樣板，建立名為 myRedis6x 的參數群組。

```
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>
```

此動作的回應看起來會與以下內容相似。

```
<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>
```

建立參數群組時，它會擁有系列的預設值。若要變更預設值，您必須修改參數群組。如需詳細資訊，請參閱 修改參數群組。

如需詳細資訊，請參閱 CreateParameterGroup。

依名稱列出參數群組

您可以使用 MemoryDB 主控台、或 MemoryDB API 來列出參數群組。 AWS CLI

依名稱列出參數群組 (主控台)

下列程序顯示如何使用 MemoryDB 主控台檢視參數群組清單。

若要使用 MemoryDB 主控台列出參數群組

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看所有可用參數群組的清單，請從左側的導覽窗格中，選擇 Parameter Groups (參數群組)。

依名稱列出參數群組 (AWS CLI)

若要使用產生參數群組清單 AWS CLI，請使用指令`describe-parameter-groups`。若您提供參數群組的名稱，便只會列出該參數群組。若您沒有提供參數群組的名稱，最多會列出 `--max-results` 個參數群組。在任一種情況下，都會列出參數群組的名稱、系列和描述。

Example

下列範例程式碼會列出參數群組 myRe dis6x。

若為 Linux、macOS 或 Unix：

```
aws memorydb describe-parameter-groups \
    --parameter-group-name myRedis6x
```

針對 Windows：

```
aws memorydb describe-parameter-groups ^
    --parameter-group-name myRedis6x
```

此命令的輸出看起來會與以下內容相似，列出參數群組的名稱、系列和描述。

```
{
    "ParameterGroups": [
```

```
     {
         "Name": "myRedis6x",
         "Family": "memorydb_redis6",
         "Description": "My first parameter group",
                "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
     }
    ]
}
```

Example

下列範例程式碼會列出在 Redis 引擎 5.0.6 版之後執行之參數群組的參數群組 myRedis6x。

若為 Linux、macOS 或 Unix：

```
aws memorydb describe-parameter-groups \
    --parameter-group-name myRedis6x
```

針對 Windows：

```
aws memorydb describe-parameter-groups ^
    --parameter-group-name myRedis6x
```

此指令的輸出將如下所示，列示參數群組的名稱、族群和描述。

```
{
    "ParameterGroups": [
     {
         "Name": "myRedis6x",
         "Family": "memorydb_redis6",
         "Description": "My first parameter group",
                "ARN": "arn:aws:memorydb:us-east-1:012345678912:parametergroup/
myredis6x"
     }
    ]
}
```

Example

下列範例程式碼最多可列出 20 個參數群組。

```
aws memorydb describe-parameter-groups --max-results 20
```

此命令的 JSON 輸出看起來像這樣，列出了每個參數組的名稱，系列和描述。

```
{
    "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"
        },
        ...
    ]
}
```

如需詳細資訊，請參閱 describe-parameter-groups。

依名稱列出參數群組 (記憶體資料庫 API)

若要使用 MemoryDB API 產生參數群組清單，請使用此DescribeParameterGroups動作。若您提供參數群組的名稱，便只會列出該參數群組。若您沒有提供參數群組的名稱，最多會列出 MaxResults 個參數群組。在任一種情況下，都會列出參數群組的名稱、系列和描述。

Example

下列範例程式碼最多可列出 20 個參數群組。

```
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>
```

此操作的響應看起來像這樣，列出了 memorydb_redis6 情況下的名稱，家庭和描述，為每個參數組。

```
<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

下列範例程式碼會列出參數群組 myRe dis6x。

```
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>
```

此動作的回應看起來會與以下內容相似，列出名稱、系列和描述。

```
<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>
```

如需詳細資訊，請參閱 DescribeParameterGroups。

列出參數群組的值

您可以使用 MemoryDB 主控台、或 MemoryDB API 列出參數群組的 AWS CLI參數及其值。

列出參數群組的值 (主控台)

下列程序顯示如何使用 MemoryDB 主控台列出參數群組的參數及其值。

若要使用 MemoryDB 主控台列出參數群組的參數及其值

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看所有可用參數群組的清單，請從左側的導覽窗格中，選擇 Parameter Groups (參數群組)。
3. 透過選擇參數群組名稱的名稱 (而不是旁邊的方塊)，選擇要列出其參數和值的參數群組。

   參數及其值會在畫面底部列出。根據參數的數量，您可能需要向上或向下捲動來尋找您想要的參數。

列出參數群組的值 (AWS CLI)

若要使用列示參數群組的參數及其值 AWS CLI，請使用指令`describe-parameters`。

Example

下列範例程式碼會列出參數群組 myRe dis6x 的所有參數及其值。

若為 Linux、macOS 或 Unix：

```
aws memorydb describe-parameters \
    --parameter-group-name myRedis6x
```

針對 Windows：

```
aws memorydb describe-parameters ^
    --parameter-group-name myRedis6x
```

如需詳細資訊，請參閱 `describe-parameters`。

列出參數群組的值 (記憶體資料庫 API)

若要使用 MemoryDB API 列出參數群組的參數及其值，請使用此`DescribeParameters`動作。

如需詳細資訊，請參閱 `DescribeParameters`。

修改參數群組

> **Important**
>
> 您無法修改任何預設參數群組。

您可以修改參數群組中的某些參數值。這些參數值都會套用到與參數群組相關聯的叢集。如需參數值變更套用到參數群組時間的詳細資訊，請參閱 雷迪斯特定參數。

修改參數群組 (主控台)

下列程序顯示如何使用 MemoryDB 主控台變更參數的值。您會使用相同的程序來變更任何參數的值。

若要使用 MemoryDB 主控台變更參數的值

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看所有可用參數群組的清單，請從左側的導覽窗格中，選擇 Parameter Groups (參數群組)。
3. 選擇參數群組名稱左側的圓鈕，選擇要修改的參數群組。

   選擇「作業」，然後選擇「檢視明 或者，您也可以選擇參數群組名稱，以移至詳細資訊頁面。
4. 若要修改參數，請選擇 「編輯」。將啟用所有可編輯的參數以進行編輯。您可能必須在頁面之間移動才能找到要變更的參數。或者，您也可以在搜尋方塊中依名稱、值或類型來搜尋參數。
5. 進行任何必要的參數修改。
6. 若要儲存您所做的變更，請選擇 Save changes (儲存變更)。
7. 如果您修改了跨頁數的參數值，則可以選擇「預覽變更」來檢閱所有變更。若要確認變更，請選擇 [儲存變更]。若要進行更多修改，請選擇「返回」。
8. 「參數詳細資訊」頁面也提供重設為預設值的選項。若要重設為預設值，請選擇 [重設為預設值]。複選框將出現在所有參數的左側。您可以選擇要重置的內容，然後選擇繼續進行重置以確認。

選擇 [確認] 以確認對話方塊上的重設動作。

9. 參數詳細資訊頁面可讓您設定要在每個頁面上看到的參數數目。使用右側的齒輪進行這些更改。您還可以在詳細信息頁面上啟用/禁用所需的列。這些變更會在主控台的工作階段中持續執行。

   若要尋找您變更的參數名稱，請參閱 雷迪斯特定參數。

修改參數群組 (AWS CLI)

若要使用變更參數的值 AWS CLI，請使用指令`update-parameter-group`。

若要尋找您欲變更的參數名稱及允許值，請參閱 雷迪斯特定參數

如需詳細資訊，請參閱update-parameter-group。

修改參數群組 (記憶體資料庫 API)

若要使用 MemoryDB API 變更參數群組的參數值，請使用此`UpdateParameterGroup`動作。

若要尋找您欲變更的參數名稱及允許值，請參閱 雷迪斯特定參數

如需詳細資訊，請參閱 `UpdateParameterGroup`。

刪除參數群組

您可以使用 MemoryDB 主控台、或 MemoryDB API 來刪除自訂參數群組。 AWS CLI

若參數群組已和任何叢集相關聯，您便無法刪除參數群組。您也無法刪除任何預設參數群組。

刪除參數群組 (主控台)

下列程序顯示如何使用 MemoryDB 主控台刪除參數群組。

若要使用 MemoryDB 主控台刪除參數群組

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 若要查看所有可用參數群組的清單，請從左側的導覽窗格中，選擇 Parameter Groups (參數群組)。
3. 選擇參數群組名稱左側的圓鈕，選擇要刪除的參數群組。

   選擇 Actions (動作)，然後選擇 Delete (刪除 VPC)。
4. Delete Parameter Group (刪除參數群組) 確認畫面隨即出現。
5. 若要刪除參數群組，請在確認文字方塊中輸入 Delete。

   若要保留參數群組，請選擇 Cancel (取消)。

刪除參數群組 (AWS CLI)

若要使用刪除參數群組 AWS CLI，請使用指令`delete-parameter-group`。針對要刪除的參數群組，以 `--parameter-group-name` 指定的參數群組不能有任何與其相關聯的叢集，也不能是預設參數群組。

下列範例程式碼會刪除參數群組。

Example

若為 Linux、macOS 或 Unix：

```
aws memorydb delete-parameter-group \
    --parameter-group-name myRedis6x
```

針對 Windows：

```
aws memorydb delete-parameter-group ^
    --parameter-group-name myRedis6x
```

如需詳細資訊，請參閱delete-parameter-group。

刪除參數群組 (記憶體資料庫 API)

若要使用 MemoryDB API 刪除參數群組，請使用此DeleteParameterGroup動作。針對要刪除的參數群組，以 ParameterGroupName 指定的參數群組不能有任何與其相關聯的叢集，也不能是預設參數群組。

Example

下列範例程式碼會刪除參數群組。

```
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>
```

如需詳細資訊，請參閱 DeleteParameterGroup。

雷迪斯特定參數

若您沒有為 Redis 叢集指定參數群組，則會使用適合您引擎版本的預設參數群組。您無法變更預設參數群組中任何參數的值。但是，只要可條件式修改參數的值在兩個參數群組中都是相同的，您便可以建立自訂參數群組並隨時將其指派給您的叢集。如需詳細資訊，請參閱 建立參數群組。

主題

- Redis 7 版參數變更
- 雷迪斯 6 參數
- 內存數據庫節點類型的特定參數

Redis 7 版參數變更

> **Note**
>
> MemoryDB 已推出 Vector 搜尋的預覽版本，其中包含新的不可變參數群組。`default.memorydb-redis7.search.preview`此參數群組可在 MemoryDB 主控台中使用，以及使用建立叢集 CLI 命令建立新 vector-search-enabled 叢集時使用。下列 AWS 區域提供預覽版本：美國東部 (維吉尼亞北部)、美國東部 (俄亥俄)、美國西部 (奧勒岡)、亞太區域 (東京) 和歐洲 (愛爾蘭)。

參數群組族群：記憶體

Redis 7 版新增的參數如下。

名稱	詳細資訊	描述
`latency-tracking`	允許的值：`yes`、`no` 預設：`no` 類型：字串 可修改：是	設定為 yes 時，會追蹤每個命令的延遲情況，並可透過 `INFO` 延遲統計資料命令匯出百分位數分佈，同時透過 `LATENCY` 命令匯出累積延遲分佈 (長條圖)。

名稱	詳細資訊	描述
	變更生效：直接套用至叢集中所有節點。	
hash-max-listpack-entries	允許的值：0+ 預設：512 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	要壓縮資料集的雜湊項目數目上限。
hash-max-listpack-value	允許的值：0+ 預設：64 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	最大雜湊項目的臨界值，以便壓縮資料集。
zset-max-listpack-entries	允許的值：0+ 預設：128 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	要壓縮資料集的雜湊有序集項目數目上限。

名稱	詳細資訊	描述
zset-max-listpack-value	允許的值：0+ 預設：64 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	要壓縮資料集的雜湊有序集項目數目上限。

Redis 7 版變更的參數如下。

名稱	詳細資訊	描述
activerehashing	可修改：no。在 Redis 7 中，此參數是隱藏的且預設情況下為啟用。為了將其停用，您需要建立一個支援案例。	「可修改」先前為「是」。

Redis 7 版移除的參數如下。

名稱	詳細資訊	描述
hash-max-ziplist-entries	允許的值：0+ 預設：512 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	使用 listpack 而非 ziplist 來表示小雜湊編碼

名稱	詳細資訊	描述
hash-max-ziplist-value	允許的值：0+ 預設：64 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	使用 listpack 而非 ziplist 來表示小雜湊編碼
zset-max-ziplist-entries	允許的值：0+ 預設：128 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	使用 listpack 而非 ziplist 來表示小雜湊編碼。
zset-max-ziplist-value	允許的值：0+ 預設：64 類型：整數 可修改：是 變更生效：直接套用至叢集中所有節點。	使用 listpack 而非 ziplist 來表示小雜湊編碼。

雷迪斯 6 參數

> **Note**
>
> 在 Redis 引擎版本 6.2 中，當引入 r6gd 節點系列以搭配使用時資料分層，`volatile-lru`且 r6gd 節點類型支援`allkeys-lru`最大記憶體原則。`noeviction`

參數組族群：記憶庫

在 Redis 6 中添加的參數如下。

名稱	詳細資訊	描述
`maxmemory-policy`	類型：字符串 允許的值：揮發性-lru，所有鍵-lru，揮發性-LFU，揮發性隨機，所有鍵-隨機，揮發性TTL，延遲 預設值：沒有	到達記憶體用量上限時，針對鍵的移出政策。 如需詳細資訊，請參閱使用 Redis 做為 LRU 快取使用 Redis 做為 LRU 快取。
`list-compress-depth`	類型：整數 允許的值：0- 預設：0	壓縮深度是來自清單每一端的快速清單 (quicklist) 壓縮清單 (ziplist) 節點數量，這些節點會從壓縮中排除。清單的前端和尾端一律不會進行壓縮，以進行快速的推送及彈出操作。設定如下： • 0：停用所有壓縮。 • 1：啟動壓縮，從第一個節點進入前端和尾端。 [head]->node->node->...->node->[tail] 除了 [head] 和 [tail] 之外的所有節點都會進行壓縮。 • 2：啟動壓縮，從第二個節點進入前端和尾端。

名稱	詳細資訊	描述
		[head]->[next]->node->node->...->node->[prev]->[tail] [head]、[next]、[prev] 和 [tail] 不會進行壓縮。其他所有節點都會壓縮。 • 其他等服務...
hll-sparse-max-bytes	類型：整數 允許的值： 預設：3000	HyperLogLog 稀疏表示字節限制。限制包含 16 位元組的標頭。當 HyperLogLog 使用稀疏表現法超過此限制時，會將其轉換為密集表現法。 不建議使用大於 16000 的值，因為屆時密集表示可以更有效率的使用記憶體。 我們建議使用大約 3000 的值，以獲得空間效率編碼的好處，而不會減慢PFADD太多速度，這是稀疏編碼的 O（N）。當 CPU 不是一個問題時，該值可以提高到〜 10000，但是空間是，並且數據集由許多 HyperLogLogs 在 0-15000 範圍內的基數組成。
lfu-log-factor	類型：整數 允許的值：1- 預設：10	用於遞增 LFU 移出原則索引鍵計數器的記錄因子。
lfu-decay-time	類型：整數 允許的值：0- 預設：1	減少 LFU 驅逐政策金鑰計數器的時間 (以分鐘為單位)。

名稱	詳細資訊	描述
active-defrag-max-scan-fields	類型：整數 允許的值： 預設：1000	在作用中磁碟重組期間，將從主字典掃描處理的設定/雜湊 /zset /清單欄位數目上限。
active-defrag-threshold-upper	類型：整數 允許值：1 到 100 預設：100	進行最大程度投入量所需要的最高分散百分比。
client-output-buffer-limit-pubsub-hard-limit	類型：整數 允許的值：0- 預設：33554432	針對 Redis 發佈/訂閱用戶端：若用戶端的輸出緩衝區達到指定的位元組數，便會中斷用戶端連線。
client-output-buffer-limit-pubsub-soft-limit	類型：整數 允許的值：0- 預設：8388608	對於 Redis 發佈/訂閱用戶端：如果用戶端的輸出緩衝區達到指定的位元組數，用戶端將會中斷連線，但前提是這種情況仍然存在 client-output-buffer-limit-pubsub-soft-seconds.
client-output-buffer-limit-pubsub-soft-seconds	類型：整數 允許的值：0- 預設：60	針對 Redis 發佈/訂閱用戶端：若用戶端的輸出緩衝區維持在 client-output-buffer-limit-pubsub-soft-limit 位元組超過此秒數，便會中斷用戶端連線。

名稱	詳細資訊	描述
timeout	類型：整數 允許的值：0,20- 預設：0	節點在逾時前等待的秒數。數值為： • 0 — 永遠不要中斷閒置用戶端的連線。 • 1-19 — 無效的值。 • >=20 — 中斷閒置用戶端之前節點等待的秒數。
notify-keyspace-events	類型：字符串 允許的值：空值 預設值：空	Redis 通知發布/訂閱客戶端的密鑰空間事件。默認情況下，所有通知都被禁用。
maxmemory-samples	類型：整數 允許的值：1- 預設：3	對於 least-recently-used (LRU)和time-to-live （TTL） 計算，此參數代表要檢查的索引鍵的範例大小。根據預設，Redis 會選擇 3 個鍵，並使用最近使用時間距離現在最遠的項目。
slowlog-max-len	類型：整數 允許的值：0- 預設：128	Redis Slow Log 的最大長度。這個長度沒有限制。請注意，它會消耗內存。您可以回收慢日誌使用的內存 SLOWLOG RESET.
activerehashing	類型：字符串 允許的值：是，否 預設：是	主要雜湊表會每秒重新雜湊十次。每一次的重新雜湊操作都會使用 1 毫秒的 CPU 時間。 您可以在建立參數群組時設定此值。將新的參數群組指派給叢集時，此值在舊的及新的參數群組中都必須相同。

名稱	詳細資訊	描述
client-output-buffer-limit-normal-hard-limit	類型：整數 允許的值：0- 預設：0	若用戶端的輸出緩衝區達到指定的位元組數，便會中斷用戶端連線。預設為零 (無硬式限制)。
client-output-buffer-limit-normal-soft-limit	類型：整數 允許的值：0- 預設：0	若用戶端的輸出緩衝區達到指定的位元組數，便會中斷用戶端連線，但只有在此條件持續達 client-output-buffer-limit-normal-soft-seconds 時。預設為零 (無軟式限制)。
client-output-buffer-limit-normal-soft-seconds	類型：整數 允許的值：0- 預設：0	若用戶端的輸出緩衝區維持在 client-output-buffer-limit-normal-soft-limit 位元組超過此秒數，便會中斷用戶端連線。預設為零 (無時間限制)。
tcp-keepalive	類型：整數 允許的值：0- 預設：300	若將此設為非零值 (N)，節點用戶端便會每 N 秒輪詢一次，確保仍然持續連線。使用預設設定的 0，便步會發生任何輪詢。
active-defrag-cycle-min	類型：整數 允許值：1 到 75 預設：5	用於磁碟重組的最小投入量 (CPU 百分比)。

名稱	詳細資訊	描述
stream-node-max-bytes	類型：整數 允許的值：0- 預設：4096	串流資料結構是節點基數樹狀結構，它會在內部編碼多個項目。使用這個組態可指定基數樹狀結構中單一節點的最大大小 (以位元組為單位)。如果設為 0，則節點的大小沒有限制。
stream-node-max-entries	類型：整數 允許的值：0- 預設：100	串流資料結構是節點基數樹狀結構，它會在內部編碼多個項目。使用此組態來指定在附加新串流項目時，切換至新節點之前，單一節點可包含的最大項目數。如果設定為 0，則樹節點中的項目數不受限制。
lazyfree-lazy-eviction	類型：字符串 允許的值：是，否 預設：否	對驅逐執行非同步刪除。
active-defrag-ignore-bytes	類型：整數 允許的值： 預設：104857600	啟動主動磁碟重組所需要的最低分散廢棄物數量。
lazyfree-lazy-expire	類型：字符串 允許的值：是，否 預設：否	對過期的金鑰執行非同步刪除。
active-defrag-threshold-lower	類型：整數 允許值：1 到 100 預設：10	啟動主動磁碟重組所需要的最低分散百分比。

名稱	詳細資訊	描述
active-defrag-cycle-max	類型：整數 允許值：1 到 75 預設：75	用於磁碟重組的最大投入量 (CPU 百分比)。
lazyfree-lazy-server-del	類型：字符串 允許的值：是，否 預設：否	針對更新數值的命令執行非同步刪除。
slowlog-log-slower-than	類型：整數 允許的值：0- 預設：10000	Red Slow Log is 功能記錄命令的最大執行時間 (以微秒為單位)。請注意，負數會停用慢速記錄，而零值會強制記錄每個命令。
hash-max-ziplist-entries	類型：整數 允許的值：0- 預設：512	決定用於雜湊的記憶體數量。少於指定項目數的雜湊會使用特別的編碼存放，以節省空間。
hash-max-ziplist-value	類型：整數 允許的值：0- 預設：64	決定用於雜湊的記憶體數量。項目小於指定位元組數的雜湊會使用特別的編碼存放，以節省空間。
set-max-intset-entries	類型：整數 允許的值：0- 預設：512	決定要用於特定類型組 (基數為 10，介於 64 位元帶正負號整數範圍內整數的字串) 的記憶體數量。這類少於指定項目數的組會使用特別的編碼存放，以節省空間。

名稱	詳細資訊	描述
zset-max-ziplist-entries	類型：整數 允許的值：0- 預設：128	決定用於排序組的記憶體數量。少於指定元素數的排序組會使用特別的編碼存放，以節省空間。
zset-max-ziplist-value	類型：整數 允許的值：0- 預設：64	決定用於排序組的記憶體數量。項目小於指定位元組數的排序組會使用特別的編碼存放，以節省空間。
tracking-table-max-keys	類型：整數 允許的值： 預設值：	為了協助用戶端快取，Redis 支援追蹤哪些用戶端存取了哪些索引鍵。 追蹤的索引鍵有所修改時，會傳送失效訊息給所有用戶端，通知他們快取的值不再有效。此值可讓您指定此資料表的上限。
acllog-max-len	類型：整數 允許的值：1 萬 預設：128	ACL 記錄中的項目數目上限。

名稱	詳細資訊	描述
active-expire-effort	類型：整數 允許的值：1-10 預設：1	Redis 會透過兩種機制刪除已超過存留時間的索引鍵。一種機制是系統會存取一個索引鍵，並發現其過期。另一種機制則是定期任務對索引鍵進行取樣，而導致超過存留時間的索引鍵過期。此參數定義 Redis 用來在週期性任務中使項目過期的工作量。 預設值 1 會嘗試避免有超過 10% 的過期索引鍵仍存在於記憶體中。也會嘗試避免佔用總記憶體的 25% 以上以及為系統增加延遲。您最多可以將此值增加 10，以增加使索引鍵過期花費的工作量。缺點是 CPU 會較高，且延遲也可能更高。除非您發現記憶體使用量很高，且可以容忍 CPU 使用率的增加，否則建議您使用值 1。
lazyfree-lazy-user-del	類型：字符串 允許的值：是，否 預設：否	指定指DEL令的預設行為是否與作用相同UNLINK。
activedefrag	類型：字符串 允許的值：是，否 預設：否	啟用主動記憶體磁碟重組。
maxclients	類型：整數 允許的值： 預設：65000	一次可連線的用戶端數量上限。不可修改。

名稱	詳細資訊	描述
client-query-buffer-limit	類型：整數 允許的值：1048576-1073741824 預設：1073741824	單一用戶端查詢緩衝區的大小上限。更改立即發生。
proto-max-bulk-len	類型：整數 允許的值： 預設：536870912	單一元素請求的大小上限。更改立即發生。

內存數據庫節點類型的特定參數

雖然大多數的參數都只有單一值，有些參數則可能會根據所使用的節點類型而有不同的值。下表顯示每個節點類型maxmemory的預設值。maxmemory 的值為您在節點上可以用於資料及其他用途的位元組上限。

節點類型	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

節點類型	Maxmemory
db.r6gd.2xlarge	56711183565
db.r6gd.4xlarge	113609865216
db.r6gd.8xlarge	225000375228
db.r6g.large	14037181030
db.r6g.xlarge	28261849702
db.r6g.2xlarge	56711183565
db.r6g.4xlarge	113609865216
db.r6g.8xlarge	225000375228
db.r6g.12xlarge	341206346547
db.r6g.16xlarge	450000750456
db.t4g.small	1471026299
db.t4g.medium	3317862236

> **Note**
>
> 所有 MemoryDB 執行個體類型都必須在 Amazon Virtual Private Cloud VPC 中建立。

教學課程：設定 Lambda 函數以存取 Amazon VPC 中的記憶體資料庫

在本教學課程中，您可以學習如何：

- 在美國東部 -1 區域的預設 Amazon Virtual Private Cloud (Amazon VPC) 中建立一個 MemoryDB 叢集。

- 建立 Lambda 函數以存取叢集。在您建立 Lambda 函數時，可在 Amazon VPC 和 VPC 安全群組中提供子網路 ID，以允許 Lambda 函數存取 VPC 中的資源。在本教學課程中，Lambda 函數會產生 UUID，將其寫入叢集，然後從叢集擷取。
- 手動叫用 Lambda 函數，並確認它是否存取了 VPC 中的叢集。
- 清除針對本教學課程設定的 Lambda 函數、叢集和 IAM 角色。

主題

- 步驟 1：建立叢集
- 步驟 2：建立 Lambda 函數
- 步驟 3：測試 Lambda 函數
- 步驟 4：清理（可選）

步驟 1：建立叢集

若要建立叢集，請依照下列步驟執行。

主題

- 步驟 1.1：建立叢集
- 步驟 1.2：複製叢集端點
- 步驟 1.3：建立 IAM 角色
- 步驟 1.4：建立存取控制清單 (ACL)

步驟 1.1：建立叢集

在此步驟中，您可以使用 (CLI) 在帳戶的 us-east-1 區域中的預設 Amazon VPC 中建立叢集。 AWS Command Line Interface 如需使用 MemoryDB 主控台或 API 建立叢集的相關資訊，請參閱〈〉。步驟 1：建立叢集

```
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
```

請注意，「狀態」欄位的值會設定為 CREATING。MemoryDB 可能需要幾分鐘的時間才能完成叢集的建立。

步驟 1.2：複製叢集端點

驗證 MemoryDB 已使用命令完成建立叢集。 `describe-clusters`

```
aws memorydb describe-clusters \
--cluster-name cluster-01
```

複製輸出中顯示的叢集端點位址。您為 Lambda 函數建立部署套件時，會需要這個地址。

步驟 1.3：建立 IAM 角色

1. 為您的角色建立如下所示的 IAM 信任政策文件，讓您的帳戶擔任新角色。將政策儲存到名為 trust-policy.json 的檔案。請務必使用您的帳戶識別碼來取代本政策中的帳戶 ID 123456789012。

   ```
   {
   "Version": "2012-10-17",
       "Statement": [{
           "Effect": "Allow",
           "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
           "Action": "sts:AssumeRole"
       },
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "lambda.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }]
   }
   ```

2. 建立 IAM 政策文件，如下所示。將政策儲存到名為 policy.json 的檔案。請務必使用您的帳戶識別碼來取代本政策中的帳戶 ID 123456789012。

   ```
   {
   "Version": "2012-10-17",
     "Statement": [
       {
         "Effect" : "Allow",
         "Action" : [
           "memorydb:Connect"
         ],
   ```

```
      "Resource" : [
        "arn:aws:memorydb:us-east-1:123456789012:cluster/cluster-01",
        "arn:aws:memorydb:us-east-1:123456789012:user/iam-user-01"
      ]
    }
  ]
}
```

3. 建立 IAM 角色。

```
aws iam create-role \
--role-name "memorydb-iam-auth-app" \
--assume-role-policy-document file://trust-policy.json
```

4. 建立 IAM 政策。

```
aws iam create-policy \
  --policy-name "memorydb-allow-all" \
  --policy-document file://policy.json
```

5. 將 IAM 政策連接至角色。請務必使用您的帳戶 ID 替換此策略中的帳戶 ID 123456789012。

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

步驟 1.4：建立存取控制清單 (ACL)

1. 建立已啟用 IAM 的新使用者。

```
aws memorydb create-user \
  --user-name iam-user-01 \
--authentication-mode Type=iam \
--access-string "on ~* +@all"
```

2. 建立已啟用 IAM 的新使用者。

```
aws memorydb create-user \
  --user-name iam-user-01 \
--authentication-mode Type=iam \
--access-string "on ~* +@all"
```

3. 建立 ACL 並將其附加至叢集。

```
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
```

步驟 2：建立 Lambda 函數

若要建立 Lambda 函數，請執行下列步驟。

主題

- 步驟 2.1：建立部署套件
- 步驟 2.2：建立 IAM 角色 (執行角色)
- 步驟 2.3：上傳部署套件 (建立 Lambda 函數)

步驟 2.1：建立部署套件

在本教程中，我們為您的 Lambda 函數提供了 Python 中的示例代碼。

Python

下面的示例 Python 代碼讀取和寫入一個項目到您的內存數據庫集群。複製程式碼並將它儲存到名為 app.py 的檔案中。請務必將程式碼中的cluster_endpoint值取代為您在步驟 1.2 中複製的端點位址。

```
from typing import Tuple, Union
from urllib.parse import ParseResult, urlencode, urlunparse

import botocore.session
import redis
from botocore.model import ServiceId
from botocore.signers import RequestSigner
from cachetools import TTLCache, cached
import uuid

class MemoryDBIAMProvider(redis.CredentialProvider):
```

```
    def __init__(self, user, cluster_name, region="us-east-1"):
        self.user = user
        self.cluster_name = cluster_name
        self.region = region

        session = botocore.session.get_session()
        self.request_signer = RequestSigner(
            ServiceId("memorydb"),
            self.region,
            "memorydb",
            "v4",
            session.get_credentials(),
            session.get_component("event_emitter"),
        )

    # Generated IAM tokens are valid for 15 minutes
    @cached(cache=TTLCache(maxsize=128, ttl=900))
    def get_credentials(self) -> Union[Tuple[str], Tuple[str, str]]:
        query_params = {"Action": "connect", "User": self.user}

        url = urlunparse(
            ParseResult(
                scheme="https",
                netloc=self.cluster_name,
                path="/",
                query=urlencode(query_params),
                params="",
                fragment="",
            )
        )
        signed_url = self.request_signer.generate_presigned_url(
            {"method": "GET", "url": url, "body": {}, "headers": {}, "context": {}},
            operation_name="connect",
            expires_in=900,
            region_name=self.region,
        )
        # RequestSigner only seems to work if the URL has a protocol, but
        # MemoryDB only accepts the URL without a protocol
        # So strip it off the signed URL before returning
        return (self.user, signed_url.removeprefix("https://"))

def lambda_handler(event, context):
    username = "iam-user-01" # replace with your user id
    cluster_name = "cluster-01" # replace with your cache name
```

```
    cluster_endpoint = "clustercfg.cluster-01.xxxxxx.memorydb.us-east-1.amazonaws.com"
# replace with your cluster endpoint
    creds_provider = MemoryDBIAMProvider(user=username, cluster_name=cluster_name)
    redis_client = redis.Redis(host=cluster_endpoint, port=6379,
credential_provider=creds_provider, ssl=True, ssl_cert_reqs="none")

    key='uuid'
    # create a random UUID - this will be the sample element we add to the cluster
    uuid_in = uuid.uuid4().hex
    redis_client.set(key, uuid_in)
    result = redis_client.get(key)
    decoded_result = result.decode("utf-8")
    # check the retrieved item matches the item added to the cluster and print
    # the results
    if decoded_result == uuid_in:
        print(f"Success: Inserted {uuid_in}. Fetched {decoded_result} from MemoryDB.")
    else:
        raise Exception(f"Bad value retrieved. Expected {uuid_in}, got
{decoded_result}")

    return "Fetched value from MemoryDB"
```

此程式碼使用 Python 程式redis-py庫將項目放入叢集並擷取它們。此代碼用cachetools於緩存生成的 IAM 身份驗證令牌 15 分鐘。若要建立包含redis-py和的部署套件cachetools，請執行下列步驟。

在包含app.py原始程式碼檔案的專案目錄中，建立要將redis-py和程式cachetools庫安裝到其中的資料夾套件。

```
mkdir package
```

安裝redis-py和cachetools使用點子。

```
pip install --target ./package redis
pip install --target ./package cachetools
```

建立包含redis-py和cachetools程式庫的 .zip 檔案。在 Linux 和 MacOS 中，執行下列命令。在 Windows 中，使用您偏好的 zip 公用程式建立一個 .zip 檔案，其中包含redis-py和程式cachetools庫的根目錄。

```
cd package
```

```
zip -r ../my_deployment_package.zip
```

將您的函數程式碼新增至 .zip 檔案。在 Linux 和 macOS 中，執行下列命令。在 Windows 中，使用您偏好的壓縮公用程式，將 app.py 新增至 .zip 檔案的根目錄。

```
cd package
zip -r ../my_deployment_package.zip
```

步驟 2.2：建立 IAM 角色 (執行角色)

將指定的 AWS 受管理策略附加`AWSLambdaVPCAccessExecutionRole`到角色。

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole"
```

步驟 2.3：上傳部署套件 (建立 Lambda 函數)

在此步驟中，您可以使用建立函數命令建立 AWS CLI Lambda 函數 (AccessMemoryDB)。

從包含部署套件 .zip 檔案的專案目錄中，執行下列 Lambda CLI `create-function` 命令。

對於角色選項，請使用您在步驟 2.2 中建立的執行角色的 ARN。針對 vpc-config，輸入預設 VPC 子網路的逗號分隔清單，以及預設 VPC 的安全群組識別碼。您可以在 Amazon VPC 主控台中找到這些值。若要尋找預設 VPC 的子網路，請選擇 [您的 VPC]，然後選擇 AWS 帳戶的預設 VPC。若要尋找此 VPC 的安全性群組，請移至 [安全性]，然後選擇 [安全性群組]。確認已選取 us-east-1 區域。

```
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
```

步驟 3：測試 Lambda 函數

在此步驟中，您可以使用叫用命令手動叫用 Lambda 函數。執行 Lambda 函數時，會產生 UUID，並將其寫入您在 Lambda 程式碼中指定的 ElastiCache 快取。然後 Lambda 函數從快取中取回項目。

1. 使用叫用命令叫用 Lambda 函數 (AccessMemoryDB)。 AWS Lambda

```
aws lambda invoke \
--function-name AccessMemoryDB  \
--region us-east-1 \
output.txt
```

2. 確認 Lambda 函數是否成功執行，如下：

   - 檢視 output.txt 檔案。
   - 開啟 CloudWatch 主控台並選擇函數的 CloudWatch 記錄群組 (/aws/lambda/AccessRedis)，以驗證記錄檔中的結果。日誌串流應包含類似以下的輸出內容：

```
Success: Inserted 826e70c5f4d2478c8c18027125a3e01e. Fetched
 826e70c5f4d2478c8c18027125a3e01e from MemoryDB.
```

   - 在 AWS Lambda 主控台中檢閱結果。

步驟 4：清理（可選）

若要清理，請採取這些步驟。

主題

- 步驟 4.1：刪除 Lambda 函數
- 步驟 4.2：刪除記憶體資料庫叢集
- 步驟 4.3：移除 IAM 角色和政策

步驟 4.1：刪除 Lambda 函數

```
aws lambda delete-function \
 --function-name AccessMemoryDB
```

步驟 4.2：刪除記憶體資料庫叢集

刪除叢集。

```
aws memorydb delete-cluster \
 --cluster-name cluster-01
```

移除使用者和 ACL。

```
aws memorydb delete-user \
 --user-id iam-user-01

aws memorydb delete-acl \
 --acl-name iam-acl-01
```

步驟 4.3：移除 IAM 角色和政策

```
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"
```

向量搜尋

此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

內存數據庫矢量搜索擴展了內存數據庫的功能。矢量搜索可以與現有的 MemoryDB 功能結合使用。不使用向量搜尋的應用程式不會受到其存在的影響。MemoryDB 7.1 版以後版本提供向量搜尋預覽功能，在下列區域：美國東部 (維吉尼亞北部和俄亥俄州)、美國西部 (奧勒岡)、歐洲 (愛爾蘭) 和亞太區域 (東京)。

適用於 Redis 的 Amazon MemoryDB 向量搜尋可簡化您的應用程式架構，同時提供高速向量搜尋。MemoryDB 向量搜尋非常適合峰值效能和規模是最重要的選擇標準的使用案例。您可以使用現有的 MemoryDB 資料或 Redis API 來建置機器學習和生成式人工智慧使用案例，例如擷取擴增產生、異常偵測、文件擷取和即時建議。

主題

- 向量搜尋概觀
- 向量搜尋功能和限制
- 使用案例
- 使用 AWS Management Console
- 使用 AWS Command Line Interface
- 向量搜尋指令

向量搜尋概觀

此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

向量搜尋是建立在索引的建立、維護和使用之上。每個向量搜索操作指定一個索引和它的操作被局限於該索引，也就是說，對一個索引的操作不受任何其他索引的操作的影響。除了創建和銷毀索引的操作之外，任何數量的操作都可以在任何時間對任何索引發出，這意味著在集群級別，針對多個索引的多個操作可能正在同時進行。

單個索引是存在於唯一命名空間中的命名對象，它與其他 Redis 的命名空間分開：鍵，函數等。每個索引在概念上都與傳統的資料庫表格相似，因為它的結構有兩個維度：欄和資料列。在表中的每一行

對應於一個 Redis 鍵。索引中的每一欄對應於該索引鍵的一個成員或部分。在本文檔中，術語鍵，行和記錄是相同的，並可以互換使用。同樣的術語列，字段，路徑和成員基本上是相同的，也可以互換使用。

沒有特殊命令可以添加，刪除或修改索引數據。而是修改索引中索引鍵的現有HASH或JSON命令也會自動更新索引。

主題

- 索引和 Redis 的密鑰空間
- 索引欄位類型
- 向量索引演算法
- 向量搜尋查詢運算式
- 資訊指令
- 向量搜尋安全

索引和 Redis 的密鑰空間

索引被構造和維護在 Redis 的密鑰空間的子集。多個索引可以不受限制地選擇 Redis 密鑰空間的脫節或重疊的子集。每個索引的索引鍵空間由建立索引時提供的索引鍵前置詞清單定義。前綴列表是可選的，如果省略，整個 Redis 的密鑰空間將是該索引的一部分。索引也被鍵入，它們只涵蓋具有匹配類型的鍵。目前只支援 JSON 和雜湊索引。HASH 索引僅對其前綴列表涵蓋的 HASH 鍵進行索引，同樣地，JSON 索引僅對其前綴列表涵蓋的 JSON 密鑰進行索引。索引鍵空間前置詞清單中沒有指定類型的索引鍵會被忽略，並且不會影響搜尋作業。

當 HASH 或 JSON 命令修改索引鍵空間內的索引鍵時，索引會更新。這個過程涉及為每個索引提取聲明的字段，並使用新值更新索引。更新過程是在後台線程中完成的，這意味著索引僅最終與其密鑰空間內容一致。因此，密鑰的插入或更新將不會在短時間內在搜索結果中可見。在系統負載繁重和/或數據重度突變期間，可見性延遲可能會變得更長。

索引的創建是多步驟的過程。第一步是執行定義索引的 FT.CRE ATE 命令。成功執行創建會自動啟動第二個步驟-回填。回填處理序會在背景執行緒中執行，並掃描 Redis 金鑰空間，尋找新索引的前置詞清單中的金鑰。找到的每個索引鍵都會新增至索引。最終掃描整個密鑰空間，完成索引創建過程。請注意，當回填處理序正在執行時，會允許對索引鍵進行突變，沒有任何限制，而且索引回填程序將不會完成，直到所有索引鍵都已正確編製索引。不允許在索引進行回填時嘗試的查詢操作，並且會因錯誤而終止。回填過程的完成可以從該索引的FT.INFO命令的輸出中確定（「backfill_status」）。

索引欄位類型

索引的每個欄位 (欄) 都有一個特定的類型，該類型會在建立索引時宣告，以及索引鍵內的位置。對於 HASH 鍵，位置是 HASH 中的字段名稱。對於 JSON 密鑰，該位置是 JSON 路徑描述。修改索引鍵時，會擷取與宣告欄位相關聯的資料、轉換為宣告的類型並儲存在索引中。如果資料遺失或無法成功轉換為宣告的類型，則會從索引中省略該欄位。有四種類型的字段，如下所述：

- 數字欄位包含單一數字。對於 JSON 欄位，必須遵循 JSON 數字的數值規則。對於 HASH，該字段應包含以固定或浮點數標準格式編寫的數字的 ASCII 文本。無論索引鍵內的表示方式為何，此欄位都會轉換為 64 位元浮點數，以便儲存在索引內。數字欄位可與範圍搜尋運算子搭配使用。由於基礎數字存儲在具有精度限制的浮點數中，因此適用有關浮點數數字比較的通常規則。
- 標籤欄位包含編碼為單一 UTF-8 字串的零個或多個標籤值。使用分隔符號字元 (預設值為逗號，但可以覆寫)，並移除前置和尾端空格，將字串剖析為標籤值。單一標籤欄位中可包含任何數目的標籤值。標籤欄位可用於透過區分大小寫或不區分大小寫的比較來篩選標籤值對等的查詢。
- 文字欄位包含不需要符合 UTF-8 標準的位元組。文字欄位可用於使用應用程式有意義的值來裝飾查詢結果。例如，URL 或文檔的內容等
- 向量欄位包含數字向量，也稱為嵌入。向量欄位支援使用指定演算法和距離度量的固定大小向量的 K-最近鄰搜尋 (KNN)。對於哈希索引，該字段應包含以二進制格式（小端 IEEE 754）編碼的整個向量。對於 JSON 鍵，路徑應引用填充數字的正確大小的數組。請注意，當 JSON 陣列用作向量欄位時，JSON 金鑰內陣列的內部表示會轉換成所選演算法所需的格式，以減少記憶體消耗和精確度。使用 JSON 命令的後續讀取操作將產生降低的精確度值。

向量索引演算法

提供兩種向量索引演算法：

- 平面 (Flat) — Flat 演算法是對索引中每個向量進行的蠻力線性處理，在距離計算的精確度範圍內產生精確的答案。由於索引的線性處理，對於大型索引而言，此演算法的執行時間可能會非常高。
- HNSW（分層可導航小世界）— HNSW 算法是一種替代方法，可提供正確答案的近似值，以換取大幅降低執行時間。演算法由三個參數`MEF_CONSTRUCTION`和控制`EF_RUNTIME`。前兩個參數是在索引建立時指定的，無法變更。`EF_RUNTIME`參數具有在索引建立時指定的預設值，但之後可以在任何個別查詢作業上覆寫。這三個參數會互動以平衡擷取和查詢作業期間的記憶體和 CPU 耗用量，並控制精確 KNN 搜尋 (稱為回復率) 的近似品質。

向量搜尋演算法 (平面和 HNSW) 都支援選用的`INITIAL_CAP`參數。如果有指定此指定，此參數會預先配置索引的記憶體，進而降低記憶體管理負荷並提高向量擷取速率。

像 HNSW 這樣的向量搜尋演算法可能無法有效地處理先前插入向量的刪除或覆寫。使用這些作業可能會導致過多的索引記憶體耗用量和/或回復品質降低。重新索引是恢復最佳內存使用率和/或調用的一種方法。

向量搜尋查詢運算式

搜尋和 FT. AGG AL 命令需要查詢運算式。這個表達式是由一個或多個運算符組成的單個字符串參數。每個運算子都會使用索引中的一個欄位來識別索引中索引鍵的子集。多個運算符可以使用布爾組合器以及括號進一步增強或限制收集的鍵（或結果集）的集合進行組合。

萬用字元

萬用字元運算子星號 ('*') 符合索引中的所有索引鍵。

數值範圍

數值範圍運算子的語法如下：

```
<range-search> ::= '@' <numeric-field-name> ':' '[' <bound> <bound> ']'
<bound>  ::= <number> | '(' <number>
<number> ::= <integer> | <fixed-point> | <floating-point> | 'Inf' | '-Inf' | '+Inf'
```

< numeric-field-name > 必須是類型的宣告欄位NUMERIC。默認情況下，綁定是包含性的，但可以使用前導左括號 ['('] 來製作綁定排斥。 您可以使用範圍搜尋，或將範圍搜尋轉換為單一關聯式比較 (<、<=、> =)Inf，+Inf或者-Inf做為其中一個邊界。 無論指定的數值格式為何（整數、定點數、浮點數、無窮大），數字都會轉換為 64 位元浮點以執行比較，因此降低精度。

Example 範例

```
@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
```

標籤比較

標籤比較運算子的語法如下：

```
<tag-search> ::= '@' <tag-field-name> ':' '{' <tag> [ '|' <tag> ]* '}'
```

如果運算子中的任何標籤符合記錄之標籤欄位中的任何標籤，則記錄會包含在結果集中。由設計的欄位<tag-field-name>必須是以 type 宣告之索引的欄位TAG。標籤比較的範例如下：

```
@tag-field:{ atag }
@tag-field: { tag1 | tag2 }
```

布林組合

數值或標籤運算子的結果集可以使用布林邏輯:和/或結合。括號可用於將運算子分組和/或變更評估順序。布爾邏輯運算符的語法是：

```
<expression> ::= <phrase> | <phrase> '|' <expression> | '(' <expression> ')'
<phrase> ::= <term> | <term> <phrase>
<term> ::= <range-search> | <tag-search> | '*'
```

多個術語合併成一個短語是「和」-ed。與管道 ('|') 相結合的多個短語是「或」-ed。

向量搜尋

向量搜尋運算子會執行向量欄位索引的 K-最近鄰搜尋。矢量搜索的語法是：

```
<vector-search> ::= <expression> '=>[KNN' <k> '@' <vector-field-name> '$' <parameter-
name> <modifiers> ']'
<modifiers> ::= [ 'EF_RUNTIME' <integer> ] [ 'AS' <distance-field-name>]
```

向量搜索僅適用於滿足其可以是上面定義<expression>的運算符的任意組合的向量：通配符，範圍搜索，標籤搜索和/或其布爾組合。

- <k>是一個整數，指定要返回的最近鄰向量的數量。
- <vector-field-name>必須指定類型的宣告欄位VECTOR。
- <parameter-name>field 指定FT.SEARCH或FT.AGGREGATE命令PARAM表的其中一個項目。此參數是距離計算的參考向量值。向量的值以小端 IEEE 754 二進位格式編碼為PARAM值 (與雜湊向量欄位的編碼相同)
- 對於 HNSW 類型的向量索引，可以使用選擇性EF_RUNTIME子句來覆寫建立索引時所建立之EF_RUNTIME參數的預設值。

- 可選項為結果集<distance-field-name>提供欄位名稱，以包含參考向量與定位鍵之間計算出的距離。

資訊指令

向量搜尋可增加 Redis INFO 命令，其中包含數個額外的統計資料和計數器區段。擷取區段的請求SEARCH將擷取下列所有區段：

`search_memory` 區段

名稱	描述
搜索使用記憶體位元組	所有搜尋資料結構中使用的記憶體位元組數
搜索使用記憶體人	以上的人類可讀版本

`search_index_stats` 區段

名稱	描述
索引的搜尋編號	已建立索引的數目
搜索全文索引	所有索引中非向量欄位的數目
搜索向量索引	所有索引中的向量欄位數
搜索哈希索引	HASH 類型索引鍵上的索引數
搜索索引	JSON 類型金鑰上的索引數
搜尋總索引鍵	所有索引中的索引鍵總數
搜索總索引向量	所有索引中的向量總數
搜索總索引哈希鍵	所有索引中 HASH 類型的鍵總數
搜尋總索引鍵	所有索引中 JSON 鍵的總數
搜尋總索引大小	所有索引使用的字節

名稱	描述
搜索總數 _ 全文索引 _ 大小	非向量索引結構使用的字節
搜索總向量索引大小	向量索引結構使用的位元組
搜索 _ 最大指數 _ 週期 _ 毫秒	上次擷取批次更新期間的擷取延遲

`search_ingestion` 區段

名稱	描述
搜索背景索引 _ 狀態	擷取狀態。 `NO_ACTIVITY` 意味著閒置。其他值表示在攝取過程中有關鍵字。
搜尋 (_I) 已暫停	除了重新啟動時，這應該始終是「否」。

`search_backfill` 區段

> **Note**
>
> 只有當回填目前正在進行中時，才能看見本節中所述的某些欄位。

名稱	描述
搜尋作用中 _ 回填	目前回填活動的數目
搜索 _ 反向填充（已暫停）	除了內存不足時，這應該永遠是「否」。
搜索 _ 最高 _ 后填 _ 進度 _ 百分比	完成最多回填的完成百分比 (0-100)
搜索最低 _ 后填 _ 進度 _ 百分比	最少完成回填的百分比 (0-100)

`search_query` 區段

名稱	描述
搜尋作用中的查詢	目前正在進行的`FT.AGGREGATE` 指令數`FT.SEARCH` 和指令

向量搜尋安全

Redis ACL (存取控制清單) 命令和資料存取的安全性機制會延伸以控制搜尋功能。完全支援個別搜尋指令的 ACL 控制。會提供新 ACL 品類`@search`，並更新許多既有品類 (`@fast`、`@read`、`@write`、等等) 以包括新指令。搜尋指令不會修改關鍵資料，這表示會保留用於寫入存取的既有 ACL 機制。HASH 和 JSON 操作的訪問規則不會被索引的存在修改; 正常的密鑰級訪問控制仍然適用於這些命令。

具有索引的搜尋命令也可透過 Redis ACL 控制其存取權限。存取檢查是在整個索引層級執行，而不是在每個索引鍵層級執行。這表示只有當使用者有權存取該索引的 keyspace 前置詞清單中所有可能的金鑰時，才會將索引存取權授予使用者。換句話說，索引的實際內容不會控制存取。相反，它是由用於安全性檢查的前綴列表定義的索引的理論內容。建立使用者對金鑰具有讀取和/或寫入權限，但無法存取包含該金鑰的索引的情況很容易。請注意，只需要對密鑰空間的讀取訪問權限來創建或使用索引-不考慮是否存在寫入訪問。

如需將 ACL 與 MemoryDB 搭配使用的詳細資訊，請參閱使用存取控制清單 (ACL) 驗證使用者。

向量搜尋功能和限制

此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

向量搜尋可用性

R6g、R7g 和 T4G 節點類型支援向量搜尋啟用 MemoryDB 組態，並在下列 AWS 區域提供：美國東部 (維吉尼亞北部)、美國東部 (俄亥俄)、美國西部 (奧勒岡)、亞太區域 (東京) 和歐洲 (愛爾蘭)。

參數限制

下表顯示預覽中各種向量搜尋項目的限制：

項目	最大值
向量中的維度數	32768
可以創建的索引數	10
索引中的字段數	50
同時索引 FT.CREATE 回填作業的數目	1
FT。搜索和 FT。聚合超時子句（毫秒）	60000
在 FT.AGGRAM 命令中的管道階段數	32
FT. 彙總載入子句中的欄位數	1024
匯總子句中的欄位數	16
FT. 彙總排序子句中的欄位數	16
在 FT.聚合參數/參數的數量	32
HNSW M 參數	512
HNSW EF_ 建構參數	4096
HNSW EF_ 執行階段參數	4096

縮放限制

MemoryDB 的向量搜尋目前僅限於單一碎片，且不支援水平縮放。向量搜尋支援垂直和複本縮放。

操作限制

索引持續性和回填

向量搜尋預覽會保留索引的定義，但不會保留其內容。因此，任何造成節點啟動或重新啟動的作業要求或事件都需要從其定義和來源索引資料重建所有索引。一旦恢復所有資料，就會自動啟動重建程序，不需要使用者採取任何動作即可啟動此程序。一旦資料還原，就會立即執行重建作業做為回填作業。這在功能上等同於系統為每個定義的索引自動執行 FT.CRE ATE 指令。請注意，一旦資料還原，但很可能

在索引回填完成之前，節點就可用於應用程式作業，這表示應用程式會再次看到回填，例如，使用回填索引的搜尋命令可能會遭到拒絕。如需回填的詳細資訊，請參閱向量搜尋概觀。

索引回填的完成不會在主要和複本之間同步處理。這種缺少同步處理可能會意外地顯示給應用程式，因此建議應用程式在初始化搜尋作業之前，先確認主要複本和所有複本的回填完成情況。

快照匯入/匯出和實時移轉

RDB 檔案中的搜尋索引存在會限制該資料的相容可傳輸性。預覽版定義的索引格式只有另一個預覽版叢集才能理解。因此，具有搜索索引的 RDB 文件只能在之間傳輸或通過預覽啟用 MemoryDB 集群使用。

但是，不包含索引的 RDB 文件不會以這種方式進行限制。因此，可以在匯出前刪除索引，將預覽叢集中的資料匯出至非預覽叢集。

記憶體消耗

向量索引的目前實作所耗用的記憶體量約為「一般可用性」實作所消耗的兩倍。

回填期間記憶體不足

與 Redis 寫入作業類似，索引回填會受到限制。 out-of-memory 如果正在進行回填時 Redis 記憶體已滿，則會暫停所有回填。如果記憶體可用，則會繼續回填程序。當回填由於內存不足而暫停時，也可以刪除和索引。

交易

命令`FT.CREATE`，，`FT.DROPINDEX``FT.ALIASADD`和`FT.ALIASDEL`，和不`FT.ALIASUPDATE`能在事務上下文中執行，也就是說，不是一個 MULT/EXEC 塊內或 LUA 或函數腳本中執行。

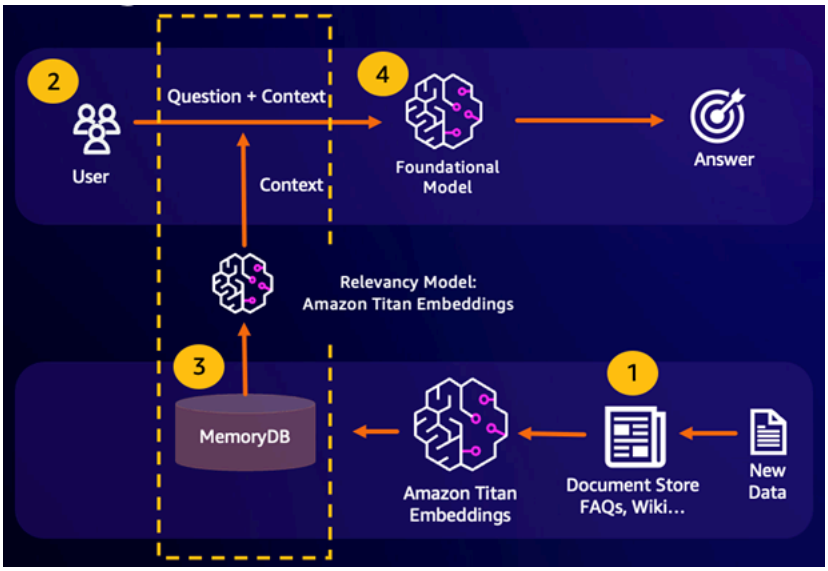
使用案例

> 此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

以下是矢量搜索的用例。

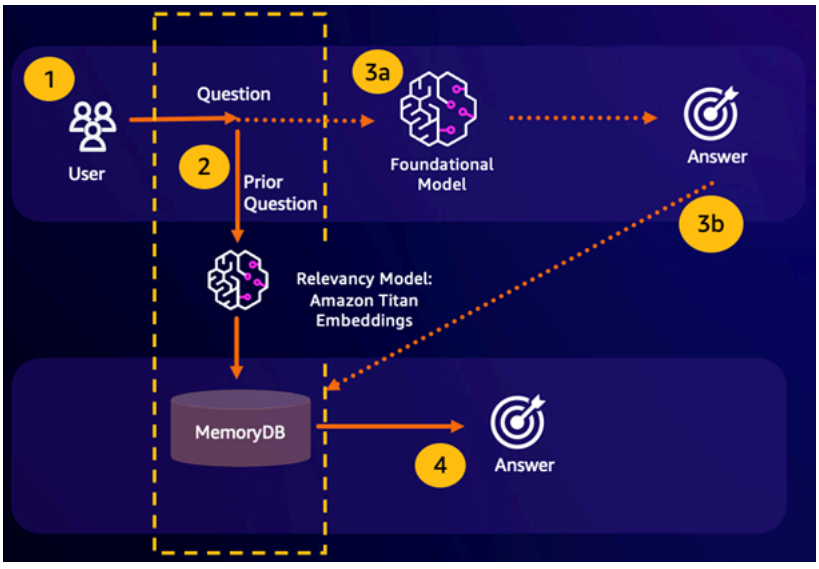
檢索增強生成 (RAG)

檢索增強生成（RAG）利用矢量搜索從大型數據庫中檢索相關段落，以增強大型語言模型（LLM）。具體來說，編碼器將輸入上下文和搜索查詢嵌入向量中，然後使用近似最近的鄰居搜索來查找語義上相似的段落。這些檢索到的段落與原始上下文連接起來，以向 LLM 提供其他相關信息，以向用戶返回更準確的響應。



基礎型號 (FM) 緩衝記憶體

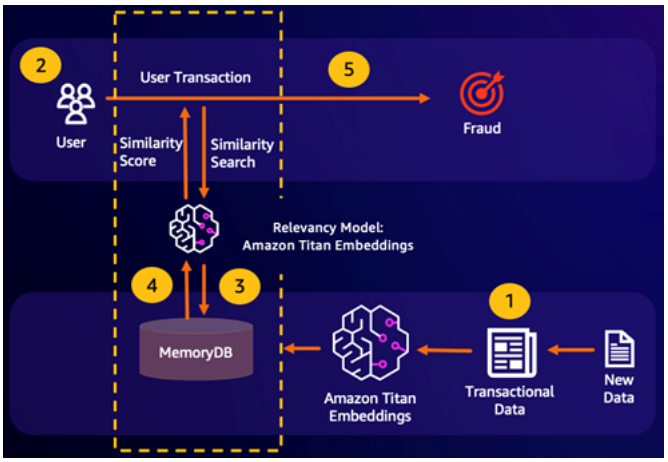
基礎模型（FM）緩衝存儲器是通過存儲從 FM 先前的結果降低計算成本的過程。FM 緩衝記憶體透過重複使用先前推論的先前結果，而不是重新計算這些結果，可減少透過 FM 推論期間所需的計算量。這款 FM 緩衝記憶體可讓大型語言模型因為 FM 的服務費用而以較低的成本加快回應速度。



- 語意搜尋命中 — 如果客戶的查詢與先前問題的定義相似度分數在語義上相似，則 FM 緩衝記憶體 (MemoryDB) 會在步驟 4 中傳回前一個問題的答案，而不會透過步驟 3 呼叫 FM。這將避免基礎模型（FM）延遲和產生的成本，從而為客戶提供更快的體驗。
- 語意搜尋未命中 — 如果客戶的查詢與先前查詢的定義相似度分數在語義上不相似，則客戶將呼叫 FM，在步驟 3a 中向客戶提供回應。然後，從 FM 生成的響應將作為向量存儲到 MemoryDB 中以供將 future 查詢（步驟 3b），以最大程度地減少語義類似問題的 FM 成本。在此流程中，不會叫用步驟 4，因為原始查詢沒有語義上類似的問題。

詐騙偵測

欺詐檢測是一種異常檢測形式，在比較淨新交易的矢量表示的同時，將有效交易以向量表示形式表示。當這些淨新交易與代表有效交易資料的向量具有較低的相似性時，就會偵測到詐騙。這允許通過建模正常行為來檢測欺詐行為，而不是試圖預測每個可能的欺詐實例。MemoryDB 允許組織在高輸送量期間執行此操作，誤報率最小且延遲為 10 毫秒。



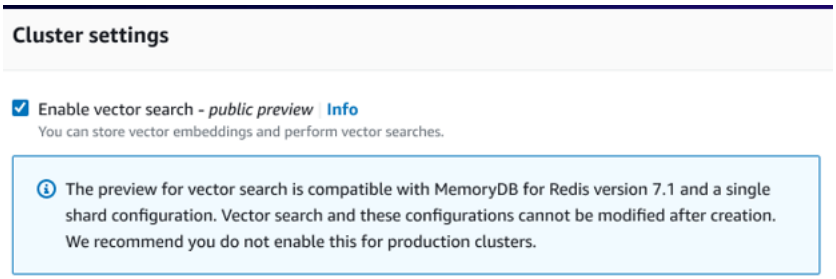
其他使用案例

- 推薦引擎可以通過將項目表示為矢量來找到用戶類似的產品或內容。向量是透過分析屬性和圖案來建立的。根據用戶模式和屬性，可以通過找到與用戶積極對齊的最相似的矢量來向用戶推薦新的看不見的項目。
- 文檔搜索引擎將文本文檔表示為數字的密集向量，捕獲語義意義。在搜尋時，引擎會將搜尋查詢轉換為向量，並使用近似鄰近搜尋尋尋找與查詢最相似向量的文件。這種向量相似性方法允許根據意義匹配文檔，而不僅僅是匹配關鍵字。

使用 AWS Management Console

此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

若要在主控台內建立啟用向量搜尋的叢集，您需要在叢集設定下啟用向量搜尋。向量搜尋可用於 Redis 7.1 版的 MemoryDB 和單一碎片組態。



如需搭配使用向量搜尋的詳細資訊 AWS Management Console，請參閱建立叢集 (主控台)。

使用 AWS Command Line Interface

此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

若要建立已啟用 MemoryDB 的向量搜尋叢集，您可以使用 MemoryDB 建立叢集命令，方法是傳遞不可變的參數群組`default.memorydb-redis7.search.preview`，以啟用向量搜尋功能的預覽模式。

```
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
```

向量搜尋指令

以下是用於矢量搜索支持的命令列表。

主題

- FT. 創建
- 英特搜索
- 英尺彙總
- 英尺下降指數
- 英尺資訊
- 英尺 _ 列表
- 英尺別名
- 英尺
- 英尺更新
- 英尺 _ 別名列表
- FT.CONFIG 獲取
- FT.CONFIG 說明
- 英特配置集
- 英尺. 輪廓
- 英尺解釋
- 英尺解釋

FT. 創建

創建一個索引，並啟動該索引的回填。如需詳細資訊，請參閱向量搜尋概觀以取得索引建構的詳細資訊。

語法

```
FT.CREATE <index-name>
ON HASH | JSON
[PREFIX <count> <prefix1> [<prefix2>...]]
SCHEMA
(<field-identifier> [AS <alias>]
```

```
  NUMERIC
| TAG [SEPARATOR <sep>] [CASESENSITIVE]
| TEXT
| VECTOR [HNSW|FLAT] <attr_count> [<attribute_name> <attribute_value>])

)+
```

結構描述

- 欄位識別碼：
  - 對於雜湊鍵，欄位識別碼是欄位名稱。
  - 對於 JSON 金鑰，欄位識別碼是 JSON 路徑。

  如需詳細資訊，請參閱 索引欄位類型。

- 欄位類型：
  - TAG：如需詳細資訊，請參閱標籤。
  - 數字：字段包含一個數字。
  - 文本：字段包含任何數據塊。
  - VECTOR：支援向量搜尋的向量欄位。
    - 算法-可以是 HNSW（分層可導航小世界）或 FLAT（蠻力）。
    - `attr_count`— 將作為算法配置傳遞的屬性數量，其中包括名稱和值。
    - `{attribute_name} {attribute_value}`— 定義索引組態的演算法特定鍵/值配對。

      對於 FLAT 算法，屬性是：

      必要：

      - DIM — 向量中的維度數。
      - 距離公制 — 可以是 [L2 | IP | 餘弦] 之一。
      - 類型 — 向量類型。唯一支援的類型為FLOAT32。

      選用：

      - INITIAL_CAP — 索引中的初始向量容量會影響索引的記憶體配置大小。

      對於 HNSW 演算法，屬性為：

      必要：

- 類型 — 向量類型。唯一支援的類型為FLOAT32。
- DIM — 向量維度，指定為正整數。最大值
- 距離公制 — 可以是 [L2 | IP | 餘弦] 之一。

選用：

- INITIAL_CAP — 索引中的初始向量容量會影響索引的記憶體配置大小。預設值為 1024。
- M-每層圖中每個節點允許的最大外出邊數。在第零層上，出出邊的最大數量為 2M。預設值為 16，最大值為 512。
- EF_CONSTRUCTION — 控制索引建構期間檢查的向量數目。此參數的值越高，會改善重新叫用率，但會犧牲較長的索引建立時間。預設值為 200。最大值為 4096。
- EF_RUNTIME — 控制查詢作業期間檢查的向量數目。這個參數的值越高，可以改善調用，但是查詢時間越長。這個參數的值可以在每個查詢的基礎上被覆蓋。預設值為 10。最大值為 4096。

傳回

返回一個簡單的字符串 OK 消息或錯誤回復。

範例

> **Note**
>
> 下列範例會在將資料傳送至 Redis 之前，使用 redis-cli 原生的引數，例如取消引用和取消逸出資料。要使用其他編程語言客戶端（Python，Ruby，C# 等），請遵循這些環境處理字符串和二進制數據的處理規則。如需有關支援用戶端的詳細資訊，請參閱建置的工具 AWS

Example 1：創建一些索引

為大小為 2 的矢量創建索引

```
FT.CREATE hash_idx1 ON HASH PREFIX 1 hash: SCHEMA vec AS VEC VECTOR HNSW 6 DIM 2 TYPE
 FLOAT32 DISTANCE_METRIC L2
OK
```

使用 HNSW 演算法建立六維 JSON 索引：

```
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 範例 2：填入一些資料

以下命令被格式化，以便它們可以作為參數執行到 redis-cli 終端程序。使用編程語言客戶端（例如 Python，Ruby，C# 等）的開發人員將需要遵循環境的處理規則來處理字符串和二進制數據。

創建一些哈希和 json 數據：

```
HSET hash:0 vec "\x00\x00\x00\x00\x00\x00\x00\x00"
HSET hash:1 vec "\x00\x00\x00\x00\x00\x00\x80\xbf"
JSON.SET json:0 . '{"vec":[1,2,3,4,5,6]}'
JSON.SET json:1 . '{"vec":[10,20,30,40,50,60]}'
JSON.SET json:2 . '{"vec":[1.1,1.2,1.3,1.4,1.5,1.6]}'
```

注意下列事項：

- 哈希和 JSON 數據的鍵具有其索引定義的前綴。
- 向量位於索引定義的適當路徑。
- 哈希向量輸入為十六進制數據，而 JSON 數據作為數字輸入。
- 向量是適當的長度，二維哈希向量條目有兩個浮點數值的十六進制數據，六維 json 向量條目有六個數字。

Example 範例 3：刪除並重新建立索引

```
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
```

請注意，新的 JSON 索引使用FLAT演算法而非HNSW演算法。另請注意，它將重新索引現有的 JSON 數據：

```
FT.SEARCH json_idx1 "*=>[KNN 100 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 3
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
6) "json:1"
7) 1) "__VEC_score"
   2) "9100"
   3) "$"
   4) "[{\"vec\":[10.0, 20.0, 30.0, 40.0, 50.0, 60.0]}]"
```

英特搜索

使用提供的查詢運算式來尋找索引內的索引鍵。找到後，可以返回這些鍵中索引字段的計數和/或內容。如需詳細資訊，請參閱 Vector 搜尋查詢運算式。

若要建立在這些範例中使用的資料，請參閱 FT.CRE ATE 指令。

語法

```
FT.SEARCH <index-name> <query>
[RETURN <token_count> (<field-identifier> [AS <alias>])+]
[TIMEOUT timeout]
[PARAMS <count> <name> <value> [<name> <value>]]
[LIMIT <offset> <count>]
[COUNT]
```

- RETURN：這個子句標識返回一個鍵的哪些字段。每個欄位上的選用 AS 子句會覆寫結果中欄位的名稱。只能指定已針對此索引宣告的欄位。
- LIMIT <offset><count>:: 該子句提供了分頁功能，因為只返回滿足偏移量和計數值的鍵。如果省略了這個子句，它默認為「LIMIT 0 10」，也就是說，只有最多 10 個鍵將被返回。

- PARAMS：鍵值對數量的兩倍。參數鍵/值對可以從查詢表達式中引用。如需詳細資訊，請參閱 Vector 搜尋查詢運算式。
- COUNT：該子句抑制返回鍵的內容，只返回鍵的數量。這是「限制 0 0 0」的別名。

傳回

返回一個數組或錯誤的答复。

- 如果操作成功完成，則返回一個數組。第一個元素是匹配查詢的鍵的總數。其餘元素是鍵名和字段列表的對。字段列表是另一個包含字段名稱和值對的數組。
- 如果索引正在回填，命令會立即傳回錯誤回覆。
- 如果達到逾時，命令會傳回錯誤回覆。

範例：執行一些搜尋

> **Note**
>
> 下列範例會在將資料傳送至 Redis 之前，使用 redis-cli 原生的引數，例如取消引用和取消逸出資料。要使用其他編程語言客戶端（Python，Ruby，C# 等），請遵循這些環境處理字符串和二進制數據的處理規則。如需有關支援用戶端的詳細資訊，請參閱建置的工具 AWS

哈希搜索

```
FT.SEARCH hash_idx1 "*=>[KNN 2 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00" DIALECT 2
1) (integer) 2
2) "hash:0"
3) 1) "__VEC_score"
   2) "0"
   3) "vec"
   4) "\x00\x00\x00\x00\x00\x00\x00\x00"
4) "hash:1"
5) 1) "__VEC_score"
   2) "1"
   3) "vec"
   4) "\x00\x00\x00\x00\x00\x00\x80\xbf"
```

這將產生兩個結果，按分數排序，即與查詢向量的距離（以十六進制輸入）。

搜尋結果

```
FT.SEARCH json_idx1 "*=>[KNN 2 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 2
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
```

這將產生兩個最接近的結果，按其分數排序，並注意 JSON 向量值被轉換為浮點數，並且查詢向量仍然是矢量數據。另請注意，由於KNN參數為 2，因此只有兩個結果。較大的值將返回更多結果：

```
FT.SEARCH json_idx1 "*=>[KNN 100 @VEC $query_vec]" PARAMS 2 query_vec
 "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
 DIALECT 2
1) (integer) 3
2) "json:2"
3) 1) "__VEC_score"
   2) "11.11"
   3) "$"
   4) "[{\"vec\":[1.1, 1.2, 1.3, 1.4, 1.5, 1.6]}]"
4) "json:0"
5) 1) "__VEC_score"
   2) "91"
   3) "$"
   4) "[{\"vec\":[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]}]"
6) "json:1"
7) 1) "__VEC_score"
   2) "9100"
   3) "$"
   4) "[{\"vec\":[10.0, 20.0, 30.0, 40.0, 50.0, 60.0]}]"
```

英尺彙總

FT.SEARCH 命令的超集，它允許對查詢表達式選擇的鍵進行大量額外的處理。

語法

```
FT.AGGREGATE index query
  [LOAD * | [count field [field ...]]]
  [TIMEOUT timeout]
  [PARAMS count name value [name value ...]]
  [FILTER expression]
  [LIMIT offset num]
  [GROUPBY count property [property ...] [REDUCE function count arg [arg ...] [AS name]
 [REDUCE function count arg [arg ...] [AS name] ...]] ...]]
  [SORTBY count [ property ASC | DESC [property ASC | DESC ...]] [MAX num]]
  [APPLY expression AS name]
```

- 過濾器，限制，GROUPBY，SORTBY 和 APPLY 子句可以以任何順序重複多次，並且可以自由混合。它們按照餵養下一個子句的輸入的輸出指定的順序施加。
- 在上述語法中，「屬性」可以是在 FT.CRE ATE 命令中宣告此索引的欄位，或先前的 APPLY 子句或 REDUCT 函式的輸出。
- LOAD 子句僅限於載入已在索引中宣告的欄位。「LOAD *」將加載索引中聲明的所有字段。
- 支持以下減速器功能：計數，計數 _ 不同，總和，最小值，最大值，平均，STDDEV，分位數，主列表，第一個值和隨機樣本。如需詳細資訊，請參閱彙總
- LIMIT <offset><count>：保留從開始<offset>並繼續的記錄<count>，最多會捨棄所有其他記錄。
- PARAMS：鍵值對數量的兩倍。參數鍵/值對可以從查詢表達式中引用。如需詳細資訊，請參閱 Vector 搜尋查詢運算式。

傳回

返回一個數組或錯誤的答复。

- 如果操作成功完成，則返回一個數組。第一個元素是沒有特定含義的整數（應忽略）。其餘的元素是最後一個階段輸出的結果。每個元素都是字段名稱和值對的數組。
- 如果索引正在回填，命令會立即傳回錯誤回覆。
- 如果達到逾時，命令會傳回錯誤回覆。

英尺下降指數

刪除索引。會刪除索引定義和相關聯的內容。Redis 的金鑰不受影響。

語法

```
FT.DROPINDEX <index-name>
```

傳回

返回一個簡單的字符串 OK 消息或錯誤回复。

英尺資訊

語法

```
FT.INFO <index-name>
```

FT.INFO 頁面的輸出是索引鍵值配對的陣列，如下表所述：

金鑰	值類型	描述
索引名稱	string	索引名稱
建立時間戳記	integer	Unix 風格的創建時間時間戳
鍵類型	string	雜湊或 JSON
密鑰前綴 (_S)	字串陣列	此索引的索引鍵前置詞
fields	欄位資訊陣列	此索引的欄位
空間使用	integer	此索引使用的記憶體位元組
滿足空間 _ 使用	integer	非向量欄位使用的記憶體位元組
向量空間使用	integer	向量欄位使用的記憶體位元組
文件	integer	索引中目前包含的索引鍵數

金鑰	值類型	描述
數字索引向量	integer	索引中目前包含的向量數
current_lag	integer	最近擷取延遲 (毫秒)
回填狀態	string	其中之一：已完成 InProgres、暫停或失敗

下表說明每個欄位的資訊：

金鑰	值類型	描述
identifier	string	欄位名稱
field_name	string	雜湊成員名稱或 JSON 路徑
type	string	其中一個：數字、標籤、文字或向量
option	string	ignore

如果欄位的類型為 Vector，則會根據演算法顯示其他資訊。

對於 HNSW 算法：

金鑰	值類型	描述
演算法	string	HNSW
data_type	string	浮點數 32
距離公制	string	其中之一：L2，IP 或餘弦
初始容量	integer	向量字段索引的初始大小
目前容量	integer	矢量字段索引的當前大小
最大邊緣	integer	建立時的 M 參數

金鑰	值類型	描述
工程建設	integer	建立時的 EF_ 建構參數
ef_Runtime	integer	建立時的 EF_ 執行階段參數

對於扁平算法：

金鑰	值類型	描述
演算法	string	平面
data_type	string	浮點數 32
距離公制	string	其中之一：L2，IP 或餘弦
初始容量	integer	向量字段索引的初始大小
目前容量	integer	矢量字段索引的當前大小

英尺 _ 列表

列出所有索引。

語法

```
FT._LIST
```

傳回

返回索引名稱的數組

英尺別名

新增索引的別名。新別名可以在需要索引名稱的任何地方使用。

語法

```
FT.ALIASADD <alias> <index-name>
```

傳回

返回一個簡單的字符串 OK 消息或錯誤回復。

英尺

刪除索引的現有別名。

語法

```
FT.ALIASDEL <alias>
```

傳回

返回一個簡單的字符串 OK 消息或錯誤回復。

英尺更新

更新現有別名以指向不同的實體索引。此指令只會影響 future 對別名的參考。目前進行中的作業 (FT.SEARCH、FT.AGGAL) 不會受到此指令的影響。

語法

```
FT.ALIASUPDATE <alias> <index>
```

傳回

返回一個簡單的字符串 OK 消息或錯誤回復。

英尺 _ 別名列表

列出索引別名。

語法

```
FT._ALIASLIST
```

傳回

返回一個數組當前別名的數量的大小。陣列的每個元素都是別名索引配對。

FT.CONFIG 獲取

返回超時參數的值。

語法

```
FT.CONFIG GET [* | <timeout>]
```

傳回

返回超時參數的值。

FT.CONFIG 說明

擷取有關逾時參數的資訊。

語法

```
FT.CONFIG HELP [* | <timeout>]
```

傳回

返回有關超時參數的信息

英特配置集

設定逾時參數。預設值為 10,000 毫秒。

> **Note**
>
> 可配置的參數名稱不區分大小寫。

語法

```
FT.CONFIG SET <timeout> <value>
```

傳回

返回超時設置的值。

英尺. 輪廓

執行查詢並傳回有關該查詢的設定檔資訊。

語法

```
FT.PROFILE

<index>
SEARCH | AGGREGATE
[LIMITED]
QUERY <query ....>
```

傳回

一個雙元素的數組。第一個元素是已進行效能分析的FT.SEARCH或FT.AGGREGATE指令的結果。第二個元素是效能和效能分析資訊的陣列。

英尺解釋

剖析查詢並傳回有關如何剖析該查詢的資訊。

語法

```
FT.EXPLAIN <index> <query>
```

傳回

包含解析結果的字符串。

英尺解釋

與 FT.EXPLAIN 命令相同，不同之處在於結果以不同的格式顯示在 redis-cli 中更有用。

語法

```
FT.EXPLAINCLI <index> <query>
```

傳回

包含解析結果的字符串。

MemoryDB for Redis is is is is is is is is is is

雲端安全是 AWS 最重視的一環。身為 AWS 客戶的您，將能從資料中心和網路架構的建置中獲益，以滿足組織最為敏感的安全要求。

安全是 AWS 與您共同肩負的責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端本身的安全 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。AWS 也提供您可安全使用的服務。第三方稽核人員會定期測試和驗證我們安全性的有效性，作為 AWS 合規計劃的一部分。若要了解 MemoryDB 適用的合規計劃，請參閱合規計劃的AWS服AWS務範圍合規計劃。
- 雲端內部的安全 – 您的責任取決於所使用的 AWS 服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規

本文件有助於您了解如何在使用 MemoryDB for Redis is is is is is is is is is is is is is is is is is is is is is is 它會示範如何設定 MemoryDB 以符合您的安全性和合規目標。您也將了解如何使用其他AWS服務來幫助您監控並保護 MemoryDB 資源。

內容

- 適用於 Redis 的記憶體資料庫中的資料保護
- 適用於 Redis 的記憶體資料庫中的身分識別和存取管理
- 日誌記錄和監控
- 適用於 Redis 的記憶體資料庫的合規性驗證
- Amazon MachyDB 中的基礎設施安全
- 網際網路流量隱私權
- Redis 內存數據庫中的服務更新

適用於 Redis 的記憶體資料庫中的資料保護

AWS共用責任模型適用於中的資料保護。如此模型所述，AWS 負責保護執行所有 AWS 雲端 的全球基礎設施。您負責維護在此基礎設施上託管內容的控制權。您也必須負責您所使用 AWS 服務 的安全組態和管理任務。如需有關資料隱私權的更多相關資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

基於資料保護目的，建議您使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 保護 AWS 帳戶 憑證，並設定個人使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 AWS CloudTrail 設定 API 和使用者活動日誌記錄。
- 使用 AWS 加密解決方案，以及 AWS 服務 內的所有預設安全控制項。
- 使用進階的受管安全服務（例如 Amazon Macie），協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在透過命令列介面或 API 存取 AWS 時，需要 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如 Name(名稱) 欄位。這包括當您使用主控台、API 或 AWS SDK 時AWS 服務使用或其他使用時。AWS CLI您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

適用於 Redis 的記憶體資料庫中的資料安全性

為了協助保護您的資料安全，Redis 和 Amazon EC2 適用的 MemoryDB 提供機制，以防止未經授權存取您在伺服器上的資料。

MemoryDB 還為叢集上的資料提供加密功能：

- 傳輸中加密會在您的資料從一處移動到另外一處時進行加密，例如在您叢集內的節點之間，或是在您的叢集與應用程式間。
- 靜態加密會在快照作業期間加密交易記錄和磁碟上資料。

您也可以使用使用存取控制清單 (ACL) 驗證使用者來控制使用者對叢集的存取。

主題

- 記憶體 DB 中的靜態加密
- 記憶體中的傳輸中加密 (TLS)
- 使用存取控制清單 (ACL) 驗證使用者

- 以 IAM 進行身分驗證

記憶體 DB 中的靜態加密

為了協助保護您的資料安全，Redis 和 Amazon S3 提供不同方式，可用於限制存取叢集中資料。如需詳細資訊，請參閱記憶數據庫和 Amazon VPC及適用於 Redis 的記憶體資料庫中的身分識別和存取管理。

MemoryDB 靜態加密始終啟用，可透過加密持續性資料來提高安全性。它加密了以下幾個方面：

- 交易記錄檔中的資料
- 同步、快照和交換作業期間的磁碟
- 存放在 Amazon S3 中的快照

MemoryDB 提供靜態的預設 (受管服務) 加密，也能讓您在 AWSKey Management Service (KMS) 中使用您自己的對稱式客戶受管客戶受管客戶受管客戶受管金鑰。

存放在 SSD (固態硬碟) 上已啟用資料分層的叢集中的資料預設為一律會加密。

如需傳輸中加密的詳細資訊，請參閱記憶體中的傳輸中加密 (TLS)

主題

- 使用來自AWS KMS 的客戶受管金鑰
- 另請參閱

使用來自AWS KMS 的客戶受管金鑰

MemoryDB 支援使用對稱式客戶受管根金鑰 (KMS 金鑰) 進行靜態加密。客戶受管 KMS 金鑰是您在 AWS 帳戶中建立、擁有和管理的加密金鑰。如需詳細資訊，請參閱 AWSKey Management Service Developer Guide 中的客戶根金鑰。必須先在AWS KMS 中建立金鑰，才能搭配 MemoryDB 使用。

若要了解如何建立 AWS KMS 根金鑰，請參閱 AWS Key Management Service 開發人員指南中的建立金鑰。

MemoryDB 能與AWS KMS 整合。如需詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的使用授權。無需客戶動作即可啟用 MemoryDB 與AWS KMS 的整合。

`kms:ViaService`條件索引鍵會將AWS KMS 金鑰的使用限制為來自指定AWS服務的請求。要`kms:ViaService`與 MemoryDB 一起使用，請在條件鍵值中包含兩個 ViaService 名稱：`memorydb.amazon_region.amazonaws.com`。如需詳細資訊，請參閱 kms:ViaService。

您可以使用AWS CloudTrail來追蹤 Redis 中的 MemoryDB 代表您傳送給AWS Key Management Service的請求。向與客戶受管金鑰AWS Key Management Service相關的發出的所有 API 呼叫都具有對應的 CloudTrail 日誌 您也可以透過呼叫 ListGrantsKMS API 呼叫來查看記憶體資料庫所建立的授權。

使用客戶受管金鑰對叢集進行加密後，叢集的所有快照都會依以下方式進行加密：

- 自動每日快照會使用與叢集關聯的客戶受管金鑰來進行加密。
- 叢集被刪除時所建立的最終快照，也使用與叢集關聯的客戶受管金鑰進行加密。
- 手動建立的快照在預設情況下，會使用與叢集關聯的金鑰來加密。您可以透過選擇其他客戶受管金鑰來覆寫此選項。
- 複製快照預設為使用與來源快照關聯的客戶受管金鑰。您可以透過選擇其他客戶受管金鑰來覆寫此選項。

> **Note**
>
> - 將快照匯出到所選 Amazon S3 儲存貯體時，無法使用客戶受管金鑰。不過，匯出至 Amazon S3 的所有快照都使用伺服器端加密來進行加密。 您可以選擇將快照檔案複製到新的 S3 物件並使用客戶受管 KMS 金鑰來加密、將檔案複製到使用 KMS 金鑰以預設加密設定的另一個 S3 儲存貯體，或是變更檔案本身的加密選項。
> - 您也可以使用客戶受管金鑰來加密手動建立的快照，這些快照不是使用客戶受管金鑰進行加密。使用此選項時，即使原始叢集上的資料未加密，Amazon S3 中存放的快照檔案仍會使用 KMS 金鑰來進行加密。
>
> 從快照還原可讓您從可用的加密選項中進行選擇，類似於建立新叢集時可用的加密選項。

- 如果您刪除金鑰或停用金鑰，並撤銷用來加密叢集之金鑰的授權，叢集就會變成無法復原。換句話說，在硬體故障後將無法修改或復原。AWSKMS 至少等待七天後，才會刪除根金鑰。金鑰刪除後，您可以使用其他客戶受管金鑰建立快照以用於封存。
- 自動金鑰輪換會保留AWS KMS 根金鑰的屬性，因此輪換並不影響您存取 MemoryDB 資料的能力。加密的 MemoryDB 叢集不支援手動金鑰輪換，其中包含建立新的根金鑰和更新任何舊金鑰的參照。如需詳細資訊，請參閱 AWSKey Management Service Developer Guide 中的輪換客戶根金鑰。
- 使用 KMS 金鑰加密 MemoryDB 叢集需要每個叢集各有一個授權。此授權會在叢集的整個生命週期內使用。此外，在建立快照期間，每個快照會使用一個授權。建立快照後，此授權就會被淘汰。

- 如需AWS KMS 授權和限制的詳細資訊，請參閱 AWSKey Management Service 開發人員指南中的配額。

另請參閱

- 記憶體中的傳輸中加密 (TLS)
- 記憶數據庫和 Amazon VPC
- 適用於 Redis 的記憶體資料庫中的身分識別和存取管理

記憶體中的傳輸中加密 (TLS)

為了協助保護您的資料安全，Redis 和 Amazon EC2 適用的 MemoryDB 提供機制，以防止未經授權存取您在伺服器上的資料。通過提供傳輸中加密功能，MemoryDB 為您提供了一種工具，可用於在數據從一個位置移動到另一個位置時幫助保護數據。例如，您可以將資料從主節點移至叢集內的僅供讀取複本節點，或在叢集和應用程式之間移動資料。

主題

- 傳輸中加密概觀
- 另請參閱

傳輸中加密概觀

適用於 Redis 的 MemoryDB 傳輸中加密是一項功能，可在最容易受到攻擊的位置 (當資料從一個位置傳輸到另一個位置時) 增加資料的安全性。

MemoryDB 傳輸中加密實現了以下功能：

- 加密的連線 — 伺服器和用戶端連線都經過傳輸層安全性 (TLS) 加密。
- 加密複寫 - 資料在主節點和複本節點之間移動時會加密。
- 伺服器身分驗證 - 用戶端可以驗證是否已連線至正確的伺服器。

從 2023 年 7 月 20 日起，TLS 1.2 是新叢集和現有叢集的最低支援版本。您可以使用此連結進一步瞭解 TLS 1.2，請參閱 AWS。

如需有關連線至 MemoryDB 叢集的詳細資訊，請參閱〈〉。使用雷迪斯 CLI 連接到內存數據庫節點

另請參閱

- 記憶體 DB 中的靜態加密
- 使用存取控制清單 (ACL) 驗證使用者
- 記憶數據庫和 Amazon VPC
- 適用於 Redis 的記憶體資料庫中的身分識別和存取管理

使用存取控制清單 (ACL) 驗證使用者

您可以使用存取控制清單 (ACL) 驗證使用者。

ACL 可讓您透過將使用者分組來控制叢集存取。這些存取控制清單的設計是組織叢集存取的一種方式。

使用 ACL 時，您可以使用存取字串來建立使用者，並將特定權限指派給他們，如下一節所述。您可以將使用者指派至與特定角色 (系統管理員、人力資源) 對齊的存取控制清單，然後部署到一或多個 MemoryDB 叢集。這樣，您可以使用相同的 MemoryDB 叢集或叢集在用戶端之間建立安全性界限，並防止用戶端存取彼此的資料。

ACL 的設計是為了支援在 Redis 6 中引入 Redis ACL。當您將 ACL 與您的 MemoryDB 叢集搭配使用時，會有一些限制：

- 您無法在存取字串中指定密碼。您可以使用CreateUser或通UpdateUser話設定密碼。
- 針對使用者權限，您需傳遞 on 和 `off` 作為存取字串的一部分。如果兩者都未在存取字串中指定，則會指派使用者，`off`且沒有叢集的存取權限。
- 您不能使用禁止的命令。如果您指定了禁止命令，則會拋出異常。如需這些指令的清單，請參閱受限制的 Redis 命令。
- 您無法使用 `reset` 命令作為存取字串的一部分。您可以使用 API 參數指定密碼，然後記憶體資料庫管理密碼。因此您無法使用 `reset`，因為它會刪除使用者的所有密碼。
- Redis 6 引入了 ACL LIST 命令。此命令會傳回使用者清單，以及套用至每個使用者的 ACL 規則。MemoryDB 支持該`ACL LIST`命令，但不像 Redis 那樣包括對密碼哈希的支持。使用 MemoryDB，您可以使用DescribeUsers操作來獲取類似的信息，包括訪問字符串中包含的規則。但是，DescribeUsers不會擷取使用者密碼。

  由記憶數據庫支持的其他只讀命令包括 ACL WHOAMI，ACL 用戶和 ACL 目錄。MemoryDB 不支援任何其他以寫入為基礎的 ACL 命令。

將 ACL 與記憶體資料庫搭配使用，在下面將有更詳細的說明。

主題

- 使用存取字串指定許可
- 向量搜尋功能
- 將 ACL 套用至記憶體資料庫的叢集

使用存取字串指定許可

若要指定 MemoryDB 叢集的權限，您可以使用或建立存取字串並將其指派給使用 AWS CLI 者。 AWS Management Console

存取字串的定義是套用至使用者的空格分隔規則清單。用於定義使用者可以執行哪些命令，以及使用者可以操作哪些索引鍵。為了執行命令，使用者必須能存取執行中的命令以及該命令存取的所有索引鍵。規則會從左到右累計套用，如果提供的字串中存在冗餘，則可以使用較簡單的字串來取代所提供的字串。

如需 ACL 規則語法的詳細資訊，請參閱 ACL。

在下列範例中，存取字串代表有權存取所有可用索引鍵和命令的活躍使用者。

```
on ~* &* +@all
```

存取字串語法可細分以下各項：

- on - 使用者是活躍使用者。
- ~* - 存取權限提供給所有可用的索引鍵。
- &*— 所有發布訂閱頻道都可以訪問。
- +@all - 存取權限提供給所有可用的命令。

先前的設定受到最低限度的限制。您可以修改這些設定，提高安全性。

在下面的範例中，存取字串代表對於以「app::」keyspace 開頭的索引鍵，存取權受限於讀取存取的使用者

```
on ~app::* -@all +@read
```

您可以列出使用者可存取的命令，進一步精簡這些許可：

+*command1* - 使用者對命令的存取權限受限於 *command1*。

+@category - 使用者的存取權限受限於某個命令類別。

如需將存取字串指派給使用者的相關資訊，請參閱「使用主控台和 CLI 建立使用者和存取控制清單」。

如果您要將現有工作負載移轉至 MemoryDB，則可以透過呼叫 ACL LIST (不包括使用者和任何密碼雜湊) 擷取存取字串。

向量搜尋功能

> **Note**
>
> 此功能正在針對 Redis 的 MemoryDB 的預覽版本中，並且可能會有所變更。

對於向量搜尋，所有搜尋指令均屬於@search品類和既有品類 @read@write，@fast並@slow會更新以包括搜尋指令。如果使用者沒有類別的存取權，則該使用者將無法存取該類別中的任何命令。例如，如果使用者沒有存取權限@search，則使用者無法執行任何與搜尋相關的指令。

下表指出搜尋指令與適當品類的對映。

VSS 指令	@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

VSS 指令	@read	@write	@fast	@slow
FT.PROFILE	Y			Y
FT.ALIASADD		Y	Y	
FT.ALIASDEL		Y	Y	
FT.ALIASUPDATE		Y	Y	
FT._ALIASLIST	Y			Y
FT.EXPLAIN	Y		Y	
FT.EXPLAINCLI	Y		Y	
FT.CONFIG	Y		Y	

將 ACL 套用至記憶體資料庫的叢集

若要使用記憶體資料庫 ACL，請執行下列步驟：

1. 建立一或多位使用者。
2. 建立 ACL 並將使用者新增至清單。
3. 將 ACL 指派給叢集。

以下詳細說明這些步驟。

主題

- 使用主控台和 CLI 建立使用者和存取控制清單

- 使用主控台和 CLI 管理存取控制清單
- 將存取控制清單指派給叢集

使用主控台和 CLI 建立使用者和存取控制清單

ACL 使用者的使用者資訊是使用者名稱，選擇性地輸入密碼和存取字串。存取字串提供索引鍵和命令的許可層級。該名稱對於用戶是唯一的，並且是傳遞給引擎的名稱。

請確定您提供的使用者權限符合 ACL 的預期用途。例如，如果您建立名為的 ACLAdministrators，則您新增至該群組的任何使用者都應將其存取字串設定為完整存取金鑰和指令。對於 e-commerce ACL 中的使用者，您可以將其存取字串設定為唯讀存取權。

MemoryDB 會自動使用用戶名為每個帳戶配置默認用戶。"default"除非將明確性新增至 ACL，否則它不會與任何叢集相關聯。您無法刪除或修改此使用者。此使用者的用意是要與先前 Redis 版本的預設行為相容，且具有允許它呼叫所有命令並存取所有索引鍵的存取字串。

每個包含預設使用者的帳戶都會建立不可變的「開放存取」ACL。這是預設使用者可以成為成員的唯一 ACL。建立叢集時，必須選取要與叢集關聯的 ACL。雖然您可以選擇將「開放存取」ACL 套用至預設使用者，但我們強烈建議您建立一個 ACL，使用者的權限僅限於其業務需求。

未啟用 TLS 的叢集必須使用「開放存取」ACL 來提供開放驗證。

您可以在沒有使用者的情況下建立 ACL。空 ACL 無法存取叢集，而且只能與啟用 TLS 的叢集相關聯。

建立使用者時，最多可以設定兩個密碼。當您修改密碼時，任何與叢集的現有連線都會保持不變。

特別是，在將 ACL 用於 MemoryDB 時，請注意以下使用者密碼限制：

- 密碼必須為 16 - 128 個可列印字元。
- 不允許使用以下非英數字元：, "" / @。

使用主控台和 CLI 管理使用者

建立使用者 (主控台)

在主控台上建立使用者

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。

2. 在左側導覽窗格中，選擇 [使用者]。
3. 選擇建立使用者
4. 在 [建立使用者] 頁面上，輸入 [名稱]。

   叢集命名限制條件如下：

   - 必須包含 1-40 個英數字元或連字號。
   - 必須以字母開頭。
   - 不能連續包含兩個連字號。
   - 結尾不能是連字號。
5. 在「密碼」下，您最多可以輸入兩個密碼。
6. 在「存取字串」下，輸入存取字串。存取字串會為允許使用者存取的索引鍵和命令設定許可層級。
7. 對於「標籤」，您可以選擇性地套用標籤來搜尋和篩選使用者，或追蹤您的 AWS 費用。
8. 選擇建立。

建立使用者 AWS CLI

若要使用 CLI 建立使用者

- 使用「建立使用者」指令建立使用者。

  若為 Linux、macOS 或 Unix：

```
aws memorydb create-user \
  --user-name user-name-1 \
  --access-string "~objects:* ~items:* ~public:*" \
  --authentication-mode \
        Passwords="abc",Type=password
```

  針對 Windows：

```
aws memorydb create-user ^
  --user-name user-name-1 ^
  --access-string "~objects:* ~items:* ~public:*" ^
  --authentication-mode \
        Passwords="abc",Type=password
```

修改使用者 (主控台)

在主控台上修改使用者

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [使用者]。
3. 選擇您要修改的使用者旁邊的選項按鈕，然後選擇 [動作]-> [修改]
4. 如果您要修改密碼，請選擇 [修改密碼] 圓鈕。請注意，如果您有兩個密碼，則在修改其中一個密碼時必須同時輸入兩個密碼。
5. 如果您要更新存取字串，請輸入新的存取字串。
6. 選擇 Modify (修改)。

修改使用者 AWS CLI

使用 CLI 修改使用者

1. 使用更新使用者命令來修改使用者。
2. 修改使用者時，會更新與該使用者相關聯的存取控制清單，以及與 ACL 相關聯的任何叢集。所有現有的連線都會保留。範例如下。

   若為 Linux、macOS 或 Unix：

   ```
   aws memorydb update-user \
     --user-name user-name-1 \
     --access-string "~objects:* ~items:* ~public:*"
   ```

   針對 Windows：

   ```
   aws memorydb update-user ^
     --user-name user-name-1 ^
     --access-string "~objects:* ~items:* ~public:*"
   ```

檢視使用者詳細資訊 (主控台

在主控台上檢視使用者詳細資訊

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [使用者]。
3. 在 [使用者名稱] 下選擇使用者，或使用搜尋方塊尋找使用者。
4. 在使用者設定下，您可以檢閱使用者的存取字串、密碼計數、狀態和 Amazon 資源名稱 (ARN)。
5. 在存取控制清單 (ACL) 下，您可以檢視使用者所屬的 ACL。
6. 在「標籤」下，您可以檢視與使用者相關聯的任何標籤。

使用檢視使用者詳細資訊 AWS CLI

使用描述-使用者命令來檢視使用者的詳細資訊。

```
aws memorydb describe-users \
  --user-name my-user-name
```

刪除使用者 (主控台)

刪除主控台上的使用者

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [使用者]。
3. 選擇您要修改的使用者旁邊的選項按鈕，然後選擇 [動作]-> [刪除]
4. 若要確認，請delete在確認文字方塊中輸入，然後選擇 [刪除]。
5. 若要取消，請選擇 Cancel (取消)。

刪除使用者 AWS CLI

使用 CLI 刪除使用者

- 使用 [刪除使用者] 指令刪除使用者。

  該帳戶將被刪除，並從其所屬的任何存取控制清單中移除。以下是範例。

若為 Linux、macOS 或 Unix：

```
aws memorydb delete-user \
  --user-name user-name-2
```

針對 Windows：

```
aws memorydb delete-user ^
  --user-name user-name-2
```

使用主控台和 CLI 管理存取控制清單

您可以建立存取控制清單來組織和控制使用者對一或多個叢集的存取，如下所示。

使用下列程序來使用主控台管理存取控制清單。

建立存取控制清單 (ACL) (主控台)

使用控制台建立存取控制清單

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [存取控制清單 (ACL)]。
3. 選擇「建立 ACL」。
4. 在 [建立存取控制清單 (ACL)] 頁面上，輸入 ACL 名稱。

   叢集命名限制條件如下：

   - 必須包含 1-40 個英數字元或連字號。
   - 必須以字母開頭。
   - 不能連續包含兩個連字號。
   - 結尾不能是連字號。
5. 在 [選取的使用者] 下，執行下列其中一項

   a. 選擇建立使用者以建立新使用者

   b. 選擇管理，然後從管理使用者對話方塊中選取使用者，然後選取選擇來新增使用者。
6. 對於標籤，您可以選擇性地套用標記來搜尋和篩選 ACL 或追蹤 AWS 成本。

7. 選擇建立。

使用建立存取控制清單 (ACL) AWS CLI

使用下列程序來使用 CLI 建立存取控制清單。

若要使用 CLI 建立新 ACL 並新增使用者

- 使用「建立 ACL」指令建立 ACL。

  若為 Linux、macOS 或 Unix：

```
aws memorydb create-acl \
  --acl-name "new-acl-1" \
  --user-names "user-name-1" "user-name-2"
```

  針對 Windows：

```
aws memorydb create-acl ^
  --acl-name "new-acl-1" ^
  --user-names "user-name-1" "user-name-2"
```

修改存取控制清單 (ACL) (主控台)

使用主控台修改存取控制清單

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [存取控制清單 (ACL)]。
3. 選擇您要修改的 ACL，然後選擇 [修改]
4. 在 [修改] 頁面的 [選取的使用者] 下，執行下列其中一項動作：
   a. 選擇建立要新增至 ACL 的使用者來建立新使用者。
   b. 選擇管理，然後從管理使用者對話方塊中選取或取消選取使用者，然後選取選擇，以新增或移除使用者。
5. 在 [建立存取控制清單 (ACL)] 頁面上，輸入 ACL 名稱。

   叢集命名限制條件如下：

- 必須包含 1-40 個英數字元或連字號。
- 必須以字母開頭。
- 不能連續包含兩個連字號。
- 結尾不能是連字號。

6. 在 [選取的使用者] 下，執行下列其中一項

   a. 選擇建立使用者以建立新使用者

   b. 選擇管理，然後從管理使用者對話方塊中選取使用者，然後選取選擇來新增使用者。

7. 選擇 [修改] 儲存變更，或選擇 [取消] 捨棄變更。

使用修改存取控制清單 (ACL) AWS CLI

透過新增使用者或使用 CLI 移除目前成員來修改 ACL

- 使用更新 ACL 指令來修改 ACL。

  若為 Linux、macOS 或 Unix：

```
aws memorydb update-acl --acl-name new-acl-1 \
--user-names-to-add user-name-3 \
--user-names-to-remove user-name-2
```

  針對 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**
>
> 任何屬於從 ACL 中移除之使用者的開啟連線都會以此指令結束。

檢視存取控制清單 (ACL) 詳細資訊 (主控台)

若要在主控台上檢視 ACL 詳細資訊

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [存取控制清單 (ACL)]。
3. 選擇 ACL 名稱下的 ACL，或使用搜尋方塊尋找 ACL。
4. 在「使用者」下，您可以檢視與 ACL 相關聯的使用者清單。
5. 在關聯的叢集下，您可以檢閱 ACL 所屬的叢集。
6. 在「標籤」下，您可以檢閱任何與 ACL 相關聯的標籤。

使用檢視存取控制清單 (ACL) AWS CLI

使用描述-acls 指令來檢視 ACL 的詳細資料。

```
aws memorydb describe-acls \
  --acl-name test-group
```

刪除存取控制清單 (ACL) (主控台)

使用主控台刪除存取控制清單

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [存取控制清單 (ACL)]。
3. 選擇您要修改的 ACL，然後選擇刪除
4. 在「刪除」頁面上，delete在確認方塊中輸入，然後選擇「刪除」或「取消」以避免刪除 ACL。

會刪除 ACL 本身，而不是屬於群組的使用者。

使用刪除存取控制清單 (ACL) AWS CLI

若要使用 CLI 刪除 ACL

- 使用「刪除 ACL」指令刪除 ACL。

  若為 Linux、macOS 或 Unix：

```
aws memorydb delete-acl /
   --acl-name
```

針對 Windows：

```
aws memorydb delete-acl ^
   --acl-name
```

上述範例會傳回下列回應。

```
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"
}
```

將存取控制清單指派給叢集

建立 ACL 並新增使用者之後，實作 ACL 的最後一個步驟就是將 ACL 指派給叢集。

使用主控台將存取控制清單指派給叢集

若要使用將 ACL 新增至叢集 AWS Management Console，請參閱建立記憶體資料庫叢集。

將存取控制清單指派給叢集使用 AWS CLI

下列 AWS CLI 作業會建立啟用傳輸中加密 (TLS) 的叢集，且具有值的acl-name參數*my-acl-name*。將子網路群組 subnet-group 取代為已存在的子網路群組。

重要參數

- **--engine-version**— 必須為 6.2 歲。
- **--tls-enabled**— 用於驗證和關聯 ACL。

- **`--acl-name`**— 此值提供由具有叢集指定存取權限之使用者組成的存取控制清單。

若為 Linux、macOS 或 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"
```

針對 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"
```

下列 AWS CLI 作業會修改啟用傳輸中加密 (TLS) 的叢集，以及具有值的acl-name參數`new-acl-2`。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name cluster-1 \
    --acl-name "new-acl-2"
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name cluster-1 ^
    --acl-name "new-acl-2"
```

以 IAM 進行身分驗證

主題

- 概要
- 限制
- 設定
- 連接

概要

當您的叢集設定為使用 Redis 第 7 版或更高版本時，透過 AWS IAM 身分驗證，您可以使用 IAM 身分驗證與 IAM 身分驗證連線。這可讓您強化安全模型，並簡化許多管理安全任務。透過 IAM 身分驗證，您可以為個別的 MemoryDB 叢集和使用者設定精細的存取控制，並遵循最低權限許可原則。MemoryDB Redis 的 IAM 身分驗證運作方式是，在 Redis 或命令中提供短期 IAM 身分驗證字符，而非長期 MemoryDB 使用者密碼。AUTH HELLO有關 IAM 身份驗證令牌的更多信息，請參閱《AWS一般參考指南》中的簽名版本 4 簽名過程和下面的代碼示例。

您可以使用 IAM 身分及其相關政策，進一步限制 Redis 存取。您也可以從使用者的聯合身分提供者，直接授予使用者 MemoryDB 叢集存取權。

若要搭配使用 AWS IAM 與 MemoryDB，您必須先建立身分驗證模式為 IAM 的 MemoryDB 使用者，才能建立或重複使用 IAM 身分。IAM 身分需有相關政策，才能將`memorydb:Connect`動作授予 MemoryDB 叢集和使用者的 IAM 身分。設定完成後，您便能使用 IAM 使用者或角色的 AWS 憑證，建立 IAM 身分驗證字符。最後，連線至 MemoryDB 叢集節點時，您需要在 Redis 用戶端提供短期 IAM 身分驗證字符做為密碼。支援憑證提供者的 Redis 用戶端能夠為每個新連線自動產生臨時憑證。MemoryDB 會針對已啟用 IAM 的記憶體資料庫使用者連線請求執行 IAM 身分驗證，並使用 IAM 驗證連線請求。

限制

使用 IAM 身分驗證，會套用以下限制：

- 使用 Redis 第 7.0 版或更高版本時，可使用 IAM 身分驗證。
- IAM 身分驗證字符的有效期限為 15 分鐘。針對長期連線，建議使用支援憑證提供者介面的 Redis 用戶端。
- 與 MemoryDB 的 IAM 驗證連線會在 12 小時後自動中斷。可以傳送包含新 IAM 身分驗證字符的 AUTH 或 HELLO 命令，將連線再延長 12 小時。

- MULTI EXEC 命令不支援 IAM 身分驗證。
- 目前，IAM 身分驗證不支援所有全域條件內容金鑰。如需有關全域條件內容金鑰的詳細資訊，請參閱《IAM 使用者指南》中的AWS全域條件內容金鑰。

設定

設定 IAM 身分驗證：

1. 建立 叢集

```
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. 為您的角色建立如下所示的 IAM 信任政策文件，讓您的帳戶擔任新角色。將政策儲存到名為 trust-policy.json 的檔案。

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
        "Action": "sts:AssumeRole"
    }
}
```

3. 建立 IAM 政策文件，如下所示。將政策儲存到名為 policy.json 的檔案。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
      "Action" : [
        "memorydb:connect"
      ],
      "Resource" : [
        "arn:aws:memorydb:us-east-1:123456789012:cluster/cluster-01",
```

```
        "arn:aws:memorydb:us-east-1:123456789012:user/iam-user-01"
      ]
    }
  ]
}
```

4. 建立 IAM 角色。

```
aws iam create-role \
  --role-name "memorydb-iam-auth-app" \
  --assume-role-policy-document file://trust-policy.json
```

5. 建立 IAM 政策。

```
aws iam create-policy \
  --policy-name "memorydb-allow-all" \
  --policy-document file://policy.json
```

6. 將 IAM 政策連接至角色。

```
aws iam attach-role-policy \
 --role-name "memorydb-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/memorydb-allow-all"
```

7. 建立已啟用 IAM 的新使用者。

```
aws memorydb create-user \
  --user-name iam-user-01 \
  --authentication-mode Type=iam \
  --access-string "on ~* +@all"
```

8. 建立 ACL 並連接使用者。

```
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
```

連接

以字符做為密碼進行連線

首先，您需要使用 AWS SigV4 預先簽章的請求，產生短期 IAM 身分驗證字符。接著，連線至 MemoryDB 叢集時，您要提供 IAM 身分驗證字符做為密碼，如以下範例所示。

```
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();
```

以下是 IAMAuthTokenRequest 的定義。

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

使用憑證提供者進行連線

下列程式碼顯示如何使用 IAM 身分驗證憑證提供者，向 MemoryDB 進行驗證。

```
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();
```

以下為 AUTH Redis 叢集用戶端範例，該用戶端將 IAM 包裝在憑證提供者AuthTokenRequest中，以在需要時自動產生臨時登入資料。

```
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());
    }
```

適用於 Redis 的記憶體資料庫中的身分識別和存取管理

AWS Identity and Access Management (IAM) 可協助管理員安全地控制 AWS 資源存取權。 AWS 服務 IAM 管理員控制哪些人可以進行身份驗證（登錄）和授權（具有權限）以使用 MemoryDB 資源。IAM 是您可以使用的 AWS 服務 ，無需額外付費。

主題

- 物件
- 使用身分驗證
- 使用政策管理存取權
- 適用於 Redis 的記憶體資料庫如何與 IAM 搭配使用
- Redis 的記憶體資料庫基於身分識別的原則範例
- 針對 Redis 身分和存取的記憶體資料庫疑難排解
- 存取控制
- 管理 MemoryDB 資源存取權限的概觀

物件

您如何使用 AWS Identity and Access Management (IAM) 會有所不同，這取決於您在 MemoryDB 中執行的工作。

服務使用者 — 如果您使用 MemoryDB 服務執行工作，則系統管理員會為您提供所需的認證和權限。當您使用更多 MemoryDB 功能來完成工作時，您可能需要額外的權限。了解存取許可的管理方式可協助您向管理員請求正確的許可。如果您無法存取 MemoryDB 中的功能，請參閱。針對 Redis 身分和存取的記憶體資料庫疑難排解

服務管理員 — 如果您負責公司的 MemoryDB 資源，您可能擁有對 MemoryDB 的完整存取權。確定您的服務使用者應該存取哪些 MemoryDB 功能和資源是您的工作。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步了解貴公司如何將 IAM 與 MemoryDB 搭配使用，請參閱。適用於 Redis 的記憶體資料庫如何與 IAM 搭配使用

IAM 管理員 — 如果您是 IAM 管理員，您可能想要瞭解如何撰寫政策來管理 MemoryDB 存取權的詳細資訊。若要檢視可在 IAM 中使用的記憶體資料庫身分型政策範例，請參閱。Redis 的記憶體資料庫基於身分識別的原則範例

使用身分驗證

驗證是您 AWS 使用身分認證登入的方式。您必須以 IAM 使用者身分或假設 IAM 角色進行驗證 (登入 AWS)。 AWS 帳戶根使用者

您可以使用透過 AWS 身分識別來源提供的認證，以聯合身分識別身分登入。 AWS IAM Identity Center (IAM 身分中心) 使用者、貴公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料都是聯合身分識別的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使 AWS 用同盟存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入的詳細資訊 AWS，請參閱《AWS 登入 使用指南》 AWS 帳戶中的如何登入您的。

如果您 AWS 以程式設計方式存取，請 AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以使用您的認證以加密方式簽署要求。如果您不使用 AWS 工具，則必須自行簽署要求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱 IAM 使用者指南中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重要素驗證 (MFA) 來增加帳戶的安全性。如需更多資訊，請參閱 AWS IAM Identity Center 使用者指南中的多重要素驗證和 IAM 使用者指南中的在 AWS中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取該帳戶中的所有資源 AWS 服務和資源。此身分稱為 AWS 帳戶 root 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱 IAM 使用者指南中的需要根使用者憑證的任務。

聯合身分

最佳作法是要求人類使用者 (包括需要系統管理員存取權的使用者) 使用與身分識別提供者的同盟，才能使用臨時認證 AWS 服務 來存取。

聯合身分識別是來自企業使用者目錄的使用者、Web 身分識別提供者、Identi ty Center 目錄，或使用透過身分識別來源提供的認證進行存取 AWS 服務 的任何使用者。 AWS Directory Service同盟身分存取時 AWS 帳戶，他們會假設角色，而角色則提供臨時認證。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，也可以連線並同步到自己身分識別來源中的一組使用者和群組，以便在所有應用

程式 AWS 帳戶 和應用程式中使用。如需 IAM Identity Center 的相關資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者是您內部的身分，具 AWS 帳戶 有單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱 IAM 使用者指南中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 內部具有特定許可的身分。它類似 IAM 使用者，但不與特定的人員相關聯。您可以切換角色，在中暫時擔任 IAM 角色。 AWS Management Console 您可以透過呼叫 AWS CLI 或 AWS API 作業或使用自訂 URL 來擔任角色。如需使用角色的方法更多相關資訊，請參閱 IAM 使用者指南中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 – 若要向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱 IAM 使用者指南中的為第三方身分提供者建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的委託人) 存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，對於某些策略 AWS 服務，您可以將策略直接附加到資源（而不是使用角色作為代理）。若要了解跨帳戶存取權角色和資源型政策間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源類型政策的差異。

- 跨服務訪問 — 有些 AWS 服務 使用其他 AWS 服務功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉寄存取工作階段 (FAS) — 當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
  - 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。
  - 服務連結角色 — 服務連結角色是連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
- 在 Amazon EC2 上執行的應用程式 — 您可以使用 IAM 角色來管理在 EC2 執行個體上執行的應用程式以及發出 AWS CLI 或 AWS API 請求的臨時登入資料。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並提供給其所有應用程式，請建立連接至執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需更多資訊，請參閱 IAM 使用者指南中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

若要了解是否要使用 IAM 角色或 IAM 使用者，請參閱 IAM 使用者指南中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透 AWS 過建立原則並將其附加至 AWS 身分識別或資源來控制中的存取。原則是一個物件 AWS ，當與身分識別或資源相關聯時，會定義其權限。 AWS 當主參與者 (使用者、root 使用者或角色工作階段) 提出要求時，評估這些原則。政策中的許可決定是否允許或拒絕請求。大多數原則會 AWS 以 JSON 文件的形式儲存在中。如需 JSON 政策文件結構和內容的更多相關資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該原則的使用者可以從 AWS Management Console AWS CLI、或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱 IAM 使用者指南中的建立 IAM 政策。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管理的策略是獨立策略，您可以將其附加到您的 AWS 帳戶. 受管政策包括 AWS 受管政策和客戶管理的策略。若要了解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

資源型政策是位於該服務中的內嵌政策。您無法在以資源為基礎的政策中使用 IAM 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些委託人 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 和 Amazon VPC 是支援 ACL 的服務範例。 AWS WAF若要進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的原則類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政

策中的明確拒絕都會覆寫該允許。如需許可範圍的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 實體許可範圍。

- 服務控制策略（SCP）— SCP 是 JSON 策略，用於指定中組織或組織單位（OU）的最大權限。AWS Organizations AWS Organizations 是一種用於分組和集中管理您企業擁 AWS 帳戶 有的多個服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 限制成員帳戶中實體的權限，包括每個 AWS 帳戶根使用者帳戶。如需組織和 SCP 的更多相關資訊，請參閱 AWS Organizations 使用者指南中的 SCP 運作方式。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需更多資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。要了解如何在涉及多個政策類型時 AWS 確定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

適用於 Redis 的記憶體資料庫如何與 IAM 搭配使用

在您使用 IAM 管理 MemoryDB 的存取權限之前，請先了解哪些 IAM 功能可用於 MemoryDB。

您可以搭配 Redis 的記憶體資料庫使用 IAM 功能

IAM 功能	內存數據庫支持
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵	否
ACL	是
ABAC (政策中的標籤)	是

IAM 功能	內存數據庫支持
臨時憑證	是
主體許可	是
服務角色	是
服務連結角色	是

若要深入瞭解 MemoryDB 和其他 AWS 服務如何搭配大多數 IAM 功能搭配使用，請參閱 IAM 使用者指南中的可與 IAM 搭配使用的AWS 服務。

記憶體資料庫的身分識別原則

支援身分型政策	是

身分型政策是可以連接到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

記憶體資料庫的身分識別原則範例

若要檢視以記憶體 DB 身分識別為基礎的原則範例，請參閱。Redis 的記憶體資料庫基於身分識別的原則範例

內存數據庫中以資源為基礎的策略

支援以資源基礎的政策	否

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源

的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

若要啟用跨帳戶存取，您可以指定在其他帳戶內的所有帳戶或 IAM 實體，作為資源型政策的主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當主體和資源位於不同時 AWS 帳戶，受信任帳戶中的 IAM 管理員也必須授與主體實體 (使用者或角色) 權限，才能存取資源。其透過將身分型政策連接到實體來授與許可。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 角色與資源型政策有何差異。

記憶體資料庫的原則動作

支援政策動作	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。原則動作通常與關聯的 AWS API 作業具有相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

若要查看記憶體資料庫動作清單，請參閱服務授權參考中的 Redis 的 MemoryDB 為 Redis 定義的動作。

MemoryDB 中的原則動作會在動作之前使用下列前置詞：

```
MemoryDB
```

如需在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
      "MemoryDB:action1",
      "MemoryDB:action2"
         ]
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 Describe 文字的所有動作，請包含以下動作：

```
"Action": "MemoryDB:Describe*"
```

若要檢視以記憶體 DB 身分識別為基礎的原則範例，請參閱。Redis 的記憶體資料庫基於身分識別的原則範例

記憶體資料庫的原則資源

支援政策資源	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。陳述式必須包含 Resource 或 NotResource 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 MemoryDB 資源類型及其 ARN 的清單，請參閱服務授權參考資料中的 MemoryDB 針對 Redis 定義的資源。若要瞭解您可以使用哪些動作指定每個資源的 ARN，請參閱針對 Redis 的 MemoryDB 定義的動作。

若要檢視以記憶體 DB 身分識別為基礎的原則範例，請參閱。Redis 的記憶體資料庫基於身分識別的原則範例

記憶體資料庫的原則條件索引鍵

支援服務特定政策條件金鑰	否

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值，請使用邏輯OR運算來 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定條件金鑰。若要查看所有 AWS 全域條件金鑰，請參閱《IAM 使用者指南》中的AWS 全域條件內容金鑰。

若要檢視以記憶體 DB 身分識別為基礎的原則範例，請參閱。Redis 的記憶體資料庫基於身分識別的原則範例

記憶體資料庫中的存取控制清單 (ACL)

支援 ACL	是

存取控制清單 (ACL) 可控制哪些委託人 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

基於屬性的訪問控制（ABAC）與內存數據庫

支援 ABAC (政策中的標籤)	是

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在中 AWS，這些屬性稱為標籤。您可以將標籤附加到 IAM 實體 (使用者或角色) 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

若要根據標籤控制存取，請使用 aws:ResourceTag/*key-name*、aws:RequestTag/*key-name* 或 aws:TagKeys 條件金鑰，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的什麼是 ABAC?。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的使用屬性型存取控制 (ABAC)。

使用臨時登入資料與記憶體資料庫

支援臨時憑證	是

當您使用臨時憑據登錄時，某些 AWS 服務 不起作用。如需其他資訊，包括哪些 AWS 服務 與臨時登入資料搭配AWS 服務 使用，請參閱 IAM 使用者指南中的 IAM。

如果您使用除了使用者名稱和密碼以外的任何方法登入，則您正在 AWS Management Console 使用臨時認證。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取時，該程序會自動建立暫時認證。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱《IAM 使用者指南》中的切換至角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而非使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

記憶體資料庫的跨服務主體權限

支援轉寄存取工作階段 (FAS)	是

當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。

記憶體資料庫的服務角色

支援服務角色	是

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。

> **Warning**
>
> 變更服務角色的權限可能會中斷 MemoryDB 功能。只有在 MemoryDB 提供指引時才編輯服務角色。

記憶體資料庫的服務連結角色

支援服務連結角色	是

服務連結角色是一種連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服務連結角色的詳細資訊，請參閱可搭配 IAM 運作的AWS 服務。在表格中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

Redis 的記憶體資料庫基於身分識別的原則範例

根據預設，使用者和角色沒有建立或修改 MemoryDB 資源的權限。他們也無法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 來執行工作。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

有關 MemoryDB 定義的動作和資源類型的詳細資訊，包括每個資源類型的 ARN 格式，請參閱服務授權參考中針對 Redis 的 MemoryDB 的動作、資源和條件索引鍵。

主題

- 政策最佳實務
- 使用記憶體資料庫主控台
- 允許使用者檢視他們自己的許可

政策最佳實務

以身分識別為基礎的原則會決定某人是否可以在您的帳戶中建立、存取或刪除 MemoryDB 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管原則並邁向最低權限權限 — 若要開始授與使用者和工作負載的權限，請使用可授與許多常見使用案例權限的AWS 受管理原則。它們在您的 AWS 帳戶. 建議您透過定義特定於您使用案例的 AWS 客戶管理政策，進一步降低使用權限。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低許可許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的權限。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。您也可以使用條件來授與服務動作的存取權 (如透過特 AWS 服務定的方式使用) AWS CloudFormation。如需更多資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。如需更多資訊，請參閱 IAM 使用者指南中的 IAM Access Analyzer 政策驗證。
- 需要多因素身份驗證 (MFA) — 如果您的案例需要 IAM 使用者或根使用者 AWS 帳戶，請開啟 MFA 以獲得額外的安全性。若要在呼叫 API 作業時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用記憶體資料庫主控台

若要存取 Redis 的主控台的 MemoryDB，您必須具有最低限度的權限集。這些權限必須允許您列出並檢視. AWS 帳戶如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

您不需要為僅對 AWS CLI 或 AWS API 進行呼叫的使用者允許最低主控台權限。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

若要確保使用者和角色仍然可以使用 MemoryDB 主控台，請將 MemoryDB `ConsoleAccess` 或`ReadOnly` AWS 受管理的原則附加至實體。如需詳細資訊，請參閱《IAM 使用者指南》中的新增許可到使用者。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此原則包含在主控台上或以程式設計方式使用 AWS CLI 或 AWS API 完成此動作的權限。

```
{
    "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": "*"
        }
    ]
}
```

針對 Redis 身分和存取的記憶體資料庫疑難排解

使用下列資訊可協助您診斷並修正使用 MemoryDB 和 IAM 時可能會遇到的常見問題。

主題

- 我沒有授權在內存數據庫中執行操作
- 我沒有授權執行 iam：PassRole
- 我想允許我的 AWS 帳戶以外的人訪問我的 MemoryDB 資源

我沒有授權在內存數據庫中執行操作

如果 AWS Management Console 告訴您您沒有執行動作的授權，則您必須聯絡管理員以尋求協助。您的管理員是提供您使用者名稱和密碼的人員。

下列範例錯誤會在 mateojackson 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 MemoryDB:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 MemoryDB:GetWidget on resource: my-example-widget
```

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 *my-example-widget* 動作存取 MemoryDB:*GetWidget* 資源。

我沒有授權執行 iam：PassRole

如果您收到未授權執行iam:PassRole動作的錯誤訊息，您的原則必須更新，以允許您將角色傳遞給 MemoryDB。

有些 AWS 服務 允許您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為的 IAM 使用者marymajor嘗試使用主控台在 MemoryDB 中執行動作時，就會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 iam:PassRole 動作。

如果您需要協助，請聯絡您的 AWS 系統管理員。您的管理員提供您的簽署憑證。

我想允許我的 AWS 帳戶以外的人訪問我的 MemoryDB 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要瞭解 MemoryDB 是否支援這些功能，請參閱。適用於 Redis 的記憶體資料庫如何與 IAM 搭配使用
- 若要了解如何提供對您所擁有資源 AWS 帳戶 的存取權，請參閱 IAM 使用者指南中您擁有的另一 AWS 帳戶 個 IAM 使用者提供存取權限。
- 若要了解如何將資源存取權提供給第三方 AWS 帳戶，請參閱 IAM 使用者指南中的提供第三方 AWS 帳戶 擁有的存取權。
- 若要了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源型政策的差異。

存取控制

您可以擁有有效的認證來驗證您的請求，但除非您有權限，否則您無法為 Redis 資源創建或訪問 MemoryDB。例如，您必須具有建立 MemoryDB 叢集的權限。

下列各節說明如何管理 Redis 的記憶體資料庫的權限。我們建議您先閱讀概觀。

- 管理 MemoryDB 資源存取權限的概觀
- 針對 Redis 的記憶體資料庫使用身分型政策 (IAM 政策)

管理 MemoryDB 資源存取權限的概觀

每個 AWS 資源都由一個 AWS 帳號擁有，建立或存取資源的權限由權限原則控制。帳戶管理員可以將許可政策連接到 IAM 身分 (即使用者、群組和角色)。此外，Redis 的 MemoryDB 也支援將權限原則附加至資源。

> **Note**
>
> 帳戶管理員 (或管理員使用者) 是具有管理員權限的使用者。如需詳細資訊，請參《IAM 使用者指南》中的 IAM 最佳實務。

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 使用者和群組位於 AWS IAM Identity Center：

  建立權限合集。請遵循AWS IAM Identity Center 使用者指南的建立許可集合中的指示。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。請遵循 IAM 使用者指南的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請遵循 IAM 使用者指南的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台)中的指示。

主題

- 適用於 Redis 的資源和作業的記憶體資料庫
- 了解資源所有權
- 管理資源存取
- 針對 Redis 的記憶體資料庫使用身分型政策 (IAM 政策)
- 資源層級許可
- 針對 Redis 的 Amazon 記憶體資料庫使用服務連結角色
- AWS適用於 Redis 的記憶體資料庫管理原則
- 記憶體資料庫 API 權限：動作、資源和條件參考

適用於 Redis 的資源和作業的記憶體資料庫

在 Redis 的內存數據庫中，主要資源是一個集群。

這些資源各與唯一的 Amazon 資源名稱 (ARN) 相關聯，如下表所示。

> **Note**
>
> 若要讓資源層級許可生效，ARN 字串上的資源名稱應為小寫。

資源類型	ARN 格式
使用者	*###:AW: ######:####*
存取控制清單 (ACL)	*##:AW: ###:####-1:123456789012## #/#*
叢集	*##AW########### 1#123456789012## #/####*
快照	*arn: AW: ######:####-1:1234 56789012: ##/####*
參數群組	*ARN#AW###########-1#1234567 89012####/* my-parameter-group
子網路群組	*##:AW: ######:#### 1:1234567 89012: ####/* my-subnet-group

記憶數據庫提供了一組操作與內存數據庫資源的工作。如需可用作業的清單，請參閱 Redis 動作的記憶體資料庫。

了解資源所有權

資源擁有者是建立資源的 AWS 帳號。也就是說，資源擁有者是驗證建立資源之要求的主體實體的 AWS 帳戶。主體實體可以是根帳戶、IAM 使用者或 IAM 角色。下列範例說明其如何運作：

- 假設您使用帳戶的根帳 AWS 戶認證來建立叢集。在這種情況下，您的 AWS 帳戶是資源的擁有者。在內存數據庫中，資源是集群。
- 假設您在 AWS 帳戶中建立 IAM 使用者，並授與建立叢集的權限給該使用者。在此情況下，使用者可以建立叢集。但是，使用者所屬的 AWS 帳戶擁有叢集資源。
- 假設您在具有建立叢集許可的 AWS 帳戶中建立 IAM 角色。在此情況下，任何可以擔任該角色的人都可以建立叢集。您的 AWS 帳號 (角色所屬) 擁有叢集資源。

管理資源存取

許可政策描述誰可以存取哪些資源。下一節說明可用來建立許可政策的選項。

> **Note**
>
> 本節討論在 Redis 的內存數據庫中使用 IAM。它不提供 IAM 服務的詳細資訊。如需完整的 IAM 文件，請參閱IAM 使用者指南中的什麼是 IAM。如需有關 IAM 政策語法和說明的資訊，請參閱IAM 使用者指南中的 AWS IAM 政策參考。

連接到 IAM 身分的政策稱為身分類型政策 (IAM 政策)。連接到資源的政策稱為資源型政策。

主題

- 身分類型政策 (IAM 政策)
- 指定政策元素：動作、效果、資源和主體
- 在政策中指定條件

身分類型政策 (IAM 政策)

您可以將政策連接到 IAM 身分。例如，您可以執行下列動作：

- 將許可政策連接至帳戶中的使用者或群組 - 帳戶管理員能夠透過與特定使用者相關聯的許可政策來授予許可。在此情況下，該使用者可以建立 MemoryDB 資源 (例如叢集、參數群組或安全性群組) 的權限。
- 將許可政策連接至角色 (授予跨帳戶許可)：您可以將身分識別型許可政策連接至 IAM 角色，藉此授予跨帳戶許可。例如，帳戶 A 中的系統管理員可以建立角色，將跨帳戶權限授與另一個 AWS 帳戶 (例如，帳戶 B) 或 AWS 服務，如下所示：

  1. 帳戶 A 管理員建立 IAM 角色，並將許可政策連接到可授與帳戶 A 中資源許可的角色。

2. 帳戶 A 管理員將信任政策連接至該角色，識別帳戶 B 做為可擔任該角的委託人。
3. 然後，帳戶 B 管理員可以將權限委派給帳戶 B 中的任何使用者擔任該角色的權限。這樣做可讓帳戶 B 中的使用者建立或存取帳戶 A 中的資源。在某些情況下，您可能想要授與 AWS 服務權限來擔任該角色。為了支援此方法，信任政策中的委託人也可以是 AWS 服務委託人。

如需使用 IAM 來委派許可的相關資訊，請參閱《IAM 使用者指南》中的存取管理。

下列範例原則可讓使用者針對您的 AWS 帳戶執行`DescribeClusters`動作。MemoryDB 還支持使用資源 ARN 進行 API 操作識別特定資源。(此方法也稱為資源層級許可)。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "DescribeClusters",
      "Effect": "Allow",
      "Action": [
         "memorydb:DescribeClusters"],
      "Resource": resource-arn
      }
   ]
}
```

如需有關將身分識別型原則與 MemoryDB 搭配使用的詳細資訊，請參閱。針對 Redis 的記憶體資料庫使用身分型政策 (IAM 政策)如需使用者、群組、角色和許可的詳細資訊，請參閱 IAM 使用者指南中的身分 (使用者、群組和角色)。

指定政策元素：動作、效果、資源和主體

對於 Redis 資源的每個 MemoryDB (請參閱適用於 Redis 的資源和作業的記憶體資料庫)，服務會定義一組 API 作業 (請參閱動作)。為了授予這些 API 操作的權限，MemoryDB 定義了一組您可以在策略中指定的操作。例如，對於 MemoryDB 叢集資源，會定義下列動作：`CreateClusterDeleteCluster`、和。`DescribeClusters`執行一項 API 操作可能需要多個動作的許可。

以下是最基本的政策元素：

- 資源 – 在政策中，您可以使用 Amazon Resource Name (ARN) 來識別要套用政策的資源。如需詳細資訊，請參閱 適用於 Redis 的資源和作業的記憶體資料庫。

- 動作：使用動作關鍵字識別您要允許或拒絕的資源操作。例如，根據指定的`Effect`，權限允`memorydb:CreateCluster`許或拒絕執行 Redis 操作的 MemoryDB 的用戶權限。`CreateCluster`
- 效果 - 您可以指定使用者要求特定動作時會有什麼效果；可為允許或拒絕。如果您未明確授予存取 (允許) 資源，則隱含地拒絕存取。您也可以明確拒絕存取資源。例如，您可以這樣做以確保使用者無法存取資源，即使不同的政策授與存取。
- 主體：在以身分為基礎的政策 (IAM 政策) 中，政策所連接的使用者就是隱含主體。對於資源型政策，您可以指定想要收到許可的使用者、帳戶、服務或其他實體 (僅適用於資源型政策)。

如需進一步了解有關 IAM 政策語法和說明的詳細資訊，請參閱《IAM 使用者指南》 中的 AWS IAM 政策參考。

如需顯示 Redis API 動作的所有記憶體資料庫的資料表，請參閱。記憶體資料庫 API 權限：動作、資源和條件參考

在政策中指定條件

當您授與許可時，您可以使用 IAM 政策語言指定政策生效時間的條件。例如，建議只在特定日期之後套用政策。如需使用政策語言指定條件的詳細資訊，請參閱IAM 使用者指南中的條件。

針對 Redis 的記憶體資料庫使用身分型政策 (IAM 政策)

這個主題提供以身分為基礎的政策範例，在該政策中帳戶管理員可以將許可政策連接至 IAM 身分 (即使用者、群組和角色)。

> **Important**
>
> 我們建議您先閱讀說明基本概念和選項的主題，以管理 Redis 資源的 MemoryDB 存取權。如需詳細資訊，請參閱 管理 MemoryDB 資源存取權限的概觀。

本主題中的各節涵蓋下列內容：

- 使用 Redis 主控台的記憶體資料庫所需的權限
- AWS適用於 Redis 的記憶體資料庫的受管 (預先定義) 政策
- 客戶受管政策範例

以下顯示許可政策範例。

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

此政策具有兩個陳述式：

- 第一個陳述式會授予權限 MemoryDB 的 Redis 動作 (`memorydb:CreateCluster`、`memorydb:DescribeClusters`、和`memorydb:UpdateCluster`) 上帳戶擁有的任何叢集。
- 第二個陳述式會對 `Resource` 值結尾指定的 IAM 角色名稱授予 IAM 動作 (`iam:PassRole`) 的許可。

此政策不指定 `Principal` 元素，因為您不會在以身分為基礎的政策中，指定取得許可的主體。當您將政策連接至使用者時，這名使用者即為隱含主體。當您將許可政策連接至 IAM 角色，該角色的信任政策中所識別的委託人即取得許可。

如需顯示適用於 Redis API 動作的所有 MemoryDB 及其套用至的資源的表格，請參閱。記憶體資料庫 API 權限：動作、資源和條件參考

使用 Redis 主控台的記憶體資料庫所需的權限

權限參考資料表會列出 Redis API 作業的 MemoryDB，並顯示每項作業所需的權限。如需有關記憶體資料庫 API 作業的詳細資訊，請參閱。記憶體資料庫 API 權限：動作、資源和條件參考

若要使用 Redis 的 MemoryDB 主控台，請先授予其他動作的權限，如下列權限原則所示。

```
{
    "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": "*"
        }
    ]
}
```

MemoryDB 主控台需要這些額外的權限，原因如下：

- MemoryDB 動作的權限可讓主控台在帳戶中顯示 MemoryDB 資源。
- 主控台需要 `ec2` 動作的許可才能來查詢 Amazon EC2，以便顯示可用區域、VPC、安全群組和帳戶屬性。
- `cloudwatch`動作權限可讓主控台擷取 Amazon CloudWatch 指標和警示，並在主控台中顯示這些指標和警示。
- `sns` 動作許可讓主控台可擷取 Amazon Simple Notification Service (Amazon SNS) 主題和訂閱，並在主控台中顯示。

客戶受管政策範例

如果您未使用預設政策並選擇使用自訂受管政策，請確保下列兩件事的其中一項。您應具有呼叫 `iam:createServiceLinkedRole` 的許可 (如需詳細資訊，請參閱範例 4：允許使用者呼叫 IAM CreateServiceLinkedRole API)。或者，您應該已經建立了 MemoryDB 服務連結的角色。

當與使用 Redis 的 MemoryDB 主控台所需的最低權限結合使用時，本節中的範例原則會授予其他權限。這些範例也與 AWS SDK 和. AWS CLI如需有關使用 MemoryDB 主控台需要哪些權限的詳細資訊，請參閱。使用 Redis 主控台的記憶體資料庫所需的權限

如需設定 IAM 使用者和群組的說明，請參閱 IAM 使用者指南中的建立您的第一個 IAM 使用者和管理員群組。

> **Important**
>
> 在生產環境中使用 IAM 政策之前，請一律先徹底進行測試。當您使用 MemoryDB 主控台時，某些看起來很簡單的 MemoryDB 動作可能需要其他動作來支援它們。例如，`memorydb:CreateCluster`授予建立 MemoryDB 叢集的權限。但是，若要執行此作業，MemoryDB 主控台會使用許多`Describe`和`List`動作來填入主控台清單。

範例

- 範例 1：允許使用者以唯讀方式存取 MemoryDB 資源
- 範例 2：允許使用者執行一般的 MemoryDB 系統管理員工作
- 範例 3：允許使用者存取所有記憶體資料庫 API 動作
- 範例 4：允許使用者呼叫 IAM CreateServiceLinkedRole API

範例 1：允許使用者以唯讀方式存取 MemoryDB 資源

下列原則會授與允許使用者列出資源的 MemoryDB 動作的權限。您通常會將此類型的許可政策連接到管理員群組。

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "MemDBUnrestricted",
      "Effect":"Allow",
      "Action": [
          "memorydb:Describe*",
          "memorydb:List*"],
      "Resource":"*"
      }
   ]
}
```

範例 2：允許使用者執行一般的 MemoryDB 系統管理員工作

常見的系統管理員工作包括修改叢集、參數和參數群組。系統管理員也可能想要取得有關 MemoryDB 事件的資訊。下列原則會授與使用者權限，以針對這些一般系統管理員工作執行 MemoryDB 動作。您通常會將此類型的許可政策連接到系統管理員群組。

```
{
   "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":"*"
      }
   ]
}
```

範例 3：允許使用者存取所有記憶體資料庫 API 動作

下列原則可讓使用者存取所有 MemoryDB 動作。建議您只將此類型的許可政策授予管理員使用者。

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "MDBAllowAll",
      "Effect":"Allow",
      "Action":[
          "memorydb:*" ],
      "Resource":"*"
      }
   ]
}
```

範例 4：允許使用者呼叫 IAM CreateServiceLinkedRole API

下列政策允許使用者呼叫 IAM `CreateServiceLinkedRole` API。我們建議您將這種類型的權限原則授與叫用突變 MemoryDB 作業的使用者。

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"CreateSLRAllows",
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"*",
      "Condition":{
        "StringLike":{
          "iam:AWS ServiceName":"memorydb.amazonaws.com"
        }
      }
    }
  ]
}
```

資源層級許可

您可以在 IAM 政策中指定資源來限制許可的範圍。許多 AWS CLI API 動作都支援視動作行為而有所不同的資源類型。每個 IAM 政策陳述式授予在資源上執行動作的許可。當動作沒有作用於具名資源，或是當您授予對所有資源執行動作的許可，政策中資源的值是萬用字元 (*)。對於許多 API 動作，您可以透過指定資源的 Amazon Resource Name (ARN) 或符合多個資源的 ARN 模式，來限制使用者可以修改的資源。若要依照資源限制許可，請依照 ARN 指定資源。

記憶體資源 ARN 格式

> **Note**
>
> 若要讓資源層級許可生效，ARN 字串上的資源名稱應為小寫。

- *### — ARN: AW: ###:####*
- *#### — ##:AW: ###:####-1:123456789012: ##/## ACL*
- *##-ARN#AW###########-1#123456789012###/####*
- *## — ##AW###########-1#123456789012###/####*
- *###-ARN#AW###########-1#123456789012####/*my-parameter-group
- *##### — ARN: AW: ######:####-1:123456789012: #####/*my-subnet-group

範例

- 範例 1：允許使用者完整存取特定 MemoryDB 資源類型
- 範例 2：拒絕使用者存取叢集。

範例 1：允許使用者完整存取特定 MemoryDB 資源類型

下列原則明確允許指定`account-id`完整存取子網路群組、安全群組和叢集類型的所有資源。

```
{
        "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/*"
```

```
        ]
}
```

範例 2：拒絕使用者存取叢集。

下列範例明確拒絕特定叢集的指定account-id存取權。

```
{
        "Sid": "Example2",
        "Effect": "Deny",
        "Action": "memorydb:*",
        "Resource": [
                "arn:aws:memorydb:us-east-1:account-id:cluster/name"
        ]
}
```

針對 Redis 的 Amazon 記憶體資料庫使用服務連結角色

適用於 Redis 的 Amazon 記憶體資料庫使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是一種獨特的 IAM 角色類型，可直接連結至 AWS 服務，例如適用於 Redis 的 Amazon MemoryDB。適用於 Redis 服務連結角色的 Amazon 記憶體資料庫是由 Amazon 記憶體資料庫為 Redis 預先定義的。此角色包含服務需要的所有許可，以代您來呼叫其他的 AWS 服務。

服務連結角色可讓您輕鬆設定適用於 Redis 的 Amazon MemoryDB，因為您不必手動新增必要的許可。這些角色已存在於您的 AWS 帳戶中，但會連結至適用於 Redis 使用案例的 Amazon MemoryDB，且具有預先定義的許可。只有適用於 Redis 的 Amazon MemoryDB 可以擔任這些角色，而且只有這些角色可以使用預先定義的許可政策。您必須先刪除角色的相關資源，才能刪除角色。這樣可以保護適用於 Redis 資源的 Amazon MemoryDB，因為您無法不小心移除存取資源的必要許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，並尋找 Service-Linked Role (服務連結角色) 欄顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

內容

- 適用於 Redis 的 Amazon 內存數據庫的服務鏈接角色許可
- 建立服務連結角色 (IAM)
  - 建立服務連結角色 (IAM 主控台)
  - 建立服務連結角色 (IAM CLI)
  - 建立服務連結角色 (IAM API)

- 編輯適用於 Redis 的 Amazon 記憶體資料庫服務連結角色說明
  - 編輯服務連結角色描述 (IAM 主控台)
  - 編輯服務連結角色描述 (IAM CLI)
  - 編輯服務連結角色描述 (IAM API)
- 刪除適用於 Redis 的 Amazon 內存數據庫的服務鏈接角色
  - 清除服務連結角色
  - 刪除服務連結角色 (IAM 主控台)
  - 刪除服務連結角色 (IAM CLI)
  - 刪除服務連結角色 (IAM API)

適用於 Redis 的 Amazon 內存數據庫的服務鏈接角色許可

適用於 Redis 的 Amazon MemoryDB 使用名為的服務連結角色 AWSServiceRoleForMemoryDB— 此政策允許 MemoryDB 在管理叢集的必要時代表您管理 AWS 資源。

AWSServiceRoleForMemoryDB 服務連結的角色許可政策允許適用於 Redis 的 Amazon MemoryDB 在指定的資源上完成下列動作：

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

如需詳細資訊，請參閱 AWS管理策略：內存數據庫ServiceRolePolicy。

允許 IAM 實體建立 AWSServiceRoleForMemoryDB 服務連結角色

將以下政策陳述式新增到該 IAM 實體的許可中：

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

允許 IAM 實體刪除 AWSServiceRoleForMemoryDB 服務連結角色

將以下政策陳述式新增到該 IAM 實體的許可中：

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

或者，您也可以使用受 AWS 管政策來為 Redis 提供對 Amazon MemoryDB 的完整存取權。

建立服務連結角色 (IAM)

您可以使用 IAM 主控台、CLI 或 API 建立服務連結角色。

建立服務連結角色 (IAM 主控台)

您可以使用 IAM 主控台建立服務連結角色。

建立服務連結角色 (主控台)

1. 登入 AWS Management Console 並開啟身分與存取權管理主控台，網址為 https://console.aws.amazon.com/iam/。
2. 在 IAM 主控台的左側導覽窗格中，選擇 [角色]。然後選擇 Create new role (建立新角色)。
3. 在 Select type of trusted entity (選擇可信任的實體類型) 下，選擇 AWS Service (AWS 服務)。
4. 在 「或」下選取要檢視其使用案例的服務，選擇 Memory DB。
5. 選擇下一步：許可。
6. 在 Policy name (政策名稱)，請注意 `MemoryDBServiceRolePolicy` 是此角色的必要項目。選擇 Next: Add Tags (下一步：新增標籤)。
7. 請注意，服務連結的角色不支援標籤。選擇 Next:Review (下一步：檢閱)。
8. (選擇性) 針對 Role description (角色描述)，編輯新服務連結角色的描述。
9. 檢閱角色，然後選擇 Create role (建立角色)。

建立服務連結角色 (IAM CLI)

您可以使用的 IAM 操作 AWS Command Line Interface 來建立服務連結角色。此角色可包含服務擔任該角色所需的信任政策與內嵌政策。

建立服務連結角色 (CLI)

使用以下操作：

```
$ aws iam create-service-linked-role --aws-service-name memorydb.amazonaws.com
```

建立服務連結角色 (IAM API)

您可以使用 IAM API 建立服務連結角色。此角色可包含服務擔任該角色所需的信任政策與內嵌政策。

建立服務連結角色 (API)

使用 CreateServiceLinkedRole API 呼叫。在請求中指定 `memorydb.amazonaws.com` 的服務名稱。

編輯適用於 Redis 的 Amazon 記憶體資料庫服務連結角色說明

適用於 Redis 的 Amazon 記憶體資料庫不允許您編輯服務連結的 AWSServiceRoleForMemoryDB 角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。

編輯服務連結角色描述 (IAM 主控台)

您可以使用 IAM 主控台來編輯服務連結角色描述。

編輯服務連結角色的說明 (主控台)

1. 在 IAM 主控台的左側導覽窗格中，選擇 [角色]。
2. 選擇要修改之角色的名稱。
3. 在 Role description (角色說明) 的最右邊，選擇 Edit (編輯)。
4. 在方塊中輸入新的描述，然後選擇 Save (儲存)。

編輯服務連結角色描述 (IAM CLI)

您可以使用的 IAM 操作 AWS Command Line Interface 來編輯服務連結角色描述。

變更服務連結角色的說明 (CLI)

1. (選擇性) 若要檢視角色的目前說明，請使 AWS CLI 用 IAM 操作`get-role`。

   Example

   ```
   $ aws iam get-role --role-name AWSServiceRoleForMemoryDB
   ```

   透過 CLI 操作，使用角色名稱 (而非 ARN) 來參照角色。例如，如果角色具有下列 ARN：`arn:aws:iam::123456789012:role/myrole`，請將角色參照為 **`myrole`**。
2. 若要更新服務連結角色的說明，請使 AWS CLI 用 IAM 操作`update-role-description`。

若為 Linux、macOS 或 Unix：

```
$ aws iam update-role-description \
    --role-name AWSServiceRoleForMemoryDB \
    --description "new description"
```

針對 Windows：

```
$ aws iam update-role-description ^
    --role-name AWSServiceRoleForMemoryDB ^
    --description "new description"
```

編輯服務連結角色描述 (IAM API)

您可以使用 IAM API 來編輯服務連結角色描述。

變更服務連結角色的說明 (API)

1. (選用) 若要檢視角色的目前描述，請使用 IAM API 作業 GetRole。

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=GetRole
      &RoleName=AWSServiceRoleForMemoryDB
      &Version=2010-05-08
      &AUTHPARAMS
   ```

2. 若要更新角色的描述，請使用 IAM API 操作 UpdateRoleDescription。

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=UpdateRoleDescription
      &RoleName=AWSServiceRoleForMemoryDB
      &Version=2010-05-08
      &Description="New description"
   ```

刪除適用於 Redis 的 Amazon 內存數據庫的服務鏈接角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能將其刪除。

適用於 Redis 的 Amazon 記憶體資料庫不會為您刪除服務連結角色。

清除服務連結角色

在您可以使用 IAM 刪除服務連結角色之前，請先確認該角色沒有與其相關聯的資源 (叢集)。

檢查服務連結角色是否於 IAM 主控台有作用中的工作階段

1. 登入 AWS Management Console 並開啟身分與存取權管理主控台，網址為 https://console.aws.amazon.com/iam/。
2. 在 IAM 主控台的左側導覽窗格中，選擇 [角色]。然後選擇 AWSServiceRoleForMemoryDB 角色的名稱（不是核取方塊）。
3. 在所選角色的 Summary (摘要) 頁面中，選擇 Access Advisor (存取 Advisor) 分頁。
4. 在 Access Advisor (存取 Advisor) 分頁中，檢閱服務連結角色的近期活動。

刪除 Amazon 內存數據庫的 Redis 資源需要 AWSServiceRoleForMemoryDB （控制台）

- 若要刪除叢集，請參閱下列指示：
  - 使用 AWS Management Console
  - 使用 AWS CLI
  - 使用記憶體資料庫 API

刪除服務連結角色 (IAM 主控台)

您可以使用 IAM 主控台刪除服務連結角色。

刪除服務連結角色 (主控台)

1. 登入 AWS Management Console 並開啟身分與存取權管理主控台，網址為 https://console.aws.amazon.com/iam/。
2. 在 IAM 主控台的左側導覽窗格中，選擇 [角色]。然後，選擇您要刪除的角色名稱旁的核取方塊，而非名稱或資料列本身。
3. 在頁面頂端的 Role (角色) 動作中選擇 Delete (刪除) 角色。

4. 在確認頁面中，複查上次存取的服務資料，其中顯示每個選取角色上次存取 AWS 服務的時間。這可協助您確認角色目前是否作用中。如果您想要繼續進行，請選擇 Yes, Delete (是，刪除) 來提交服務連結角色以進行刪除。
5. 查看 IAM 主控台通知，監視服務連結角色刪除的進度。因為 IAM 服務連結角色刪除不同步，所以在您提交角色進行刪除之後，刪除任務可能會成功或失敗。如果任務失敗，您可以從通知中選擇 View details (檢視詳細資訊) 或 View Resources (檢視資源)，以了解刪除失敗的原因。

刪除服務連結角色 (IAM CLI)

您可以從中使用 IAM 操作 AWS Command Line Interface 來刪除服務連結角色。

刪除服務連結角色 (CLI)

1. 如果您不知道想要刪除的服務連結角色名稱，請輸入以下命令。此命令會列出您的帳戶中的角色及其 Amazon 資源名稱 (ARN)。

   ```
   $ aws iam get-role --role-name role-name
   ```

   透過 CLI 操作，使用角色名稱 (而非 ARN) 來參照角色。例如，如果角色的 ARN 為 `arn:aws:iam::123456789012:role/myrole`，參考這個角色時就需使用 **`myrole`**。
2. 因為無法刪除正在使用或具有相關聯資源的服務連結角色，所以您必須提交刪除要求。如果不符合這些條件，則可以拒絕該請求。您必須從回應中擷取 `deletion-task-id`，以檢查刪除任務的狀態。輸入下列內容，提交服務連結角色刪除請求。

   ```
   $ aws iam delete-service-linked-role --role-name role-name
   ```
3. 輸入下列內容來檢查刪除任務的狀態。

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   刪除任務的狀態可以是 `NOT_STARTED`、`IN_PROGRESS`、`SUCCEEDED` 或 `FAILED`。如果刪除失敗，則呼叫會傳回失敗原因，以進行疑難排解。

刪除服務連結角色 (IAM API)

您可以使用 IAM API 刪除服務連結角色。

刪除服務連結角色 (API)

1. 若要提交服務連結名單的刪除請求，請呼叫 DeleteServiceLinkedRole。在請求中，指定角色名稱。

   因為無法刪除正在使用或具有相關聯資源的服務連結角色，所以您必須提交刪除要求。如果不符合這些條件，則可以拒絕該請求。您必須從回應中擷取 `DeletionTaskId`，以檢查刪除任務的狀態。
2. 若要檢查刪除的狀態，請呼叫 GetServiceLinkedRoleDeletionStatus。在請求中，指定 `DeletionTaskId`。

   刪除任務的狀態可以是 `NOT_STARTED`、`IN_PROGRESS`、`SUCCEEDED` 或 `FAILED`。如果刪除失敗，則呼叫會傳回失敗原因，以進行疑難排解。

AWS適用於 Redis 的記憶體資料庫管理原則

若要新增許可給使用者、群組和角色，使用 AWS 受管政策比自己撰寫政策更容易。建立 IAM 客戶受管政策需要時間和專業知識，而受管政策可為您的團隊提供其所需的許可。若要快速開始使用，您可以使用 AWS 受管政策。這些政策涵蓋常見的使用案例，並可在您的 AWS 帳戶中可用。如需 AWS 受管政策的詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

AWS 服務會維護和更新 AWS 受管政策。您無法更改 AWS 受管政策中的許可。服務偶爾會在 AWS 受管政策中新增其他許可以支援新功能。此類型的更新會影響已連接政策的所有身分識別 (使用者、群組和角色)。當新功能啟動或新操作可用時，服務很可能會更新 AWS 受管政策。服務不會從 AWS 受管政策中移除許可，因此政策更新不會破壞您現有的許可。

此外，AWS 支援跨越多項服務之任務職能的受管政策。例如，ReadOnlyAccessAWS受管政策提供針對所有AWS服務和資源的唯讀存取權限。當服務啟動新功能時，AWS 會為新的操作和資源新增唯讀許可。如需任務職能政策的清單和說明，請參閱《IAM 使用者指南》中有關任務職能的 AWS 受管政策。

AWS管理策略：內存數據庫ServiceRolePolicy

您無法將 MemoryDBServiceRolePolicy AWS 受管理的原則附加至帳戶中的身分識別。此政策是AWS MemoryDB 服務連結角色的一部分。此角色可讓服務管理您帳戶中的網路介面和安全性群組。

MemoryDB 使用此政策中的許可來管理 EC2 安全群組和網路界面。這是管理內存數據庫集群所必需的。

許可詳細資訊

此政策包含以下許可。

```
{
        "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適用於 Redis 的記憶體資料庫的受管 (預先定義) 政策

AWS 透過提供獨立的 IAM 政策來解決許多常用案例，這些政策由 所建立與管理。AWS受管政策授與常見使用案例中必要的許可，讓您免於查詢需要哪些許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

下列AWS受管政策為 MemoryDB 專用，可連接到您帳戶中的使用者：

AmazonMemory資料庫ReadOnlyAccess

您可將 AmazonMemoryDBReadOnlyAccess 政策連接到 IAM 身分。此政策授予允許唯讀存取 MemoryDB 所有資源的管理許可。

AmazonMemoryDBReadOnlyAccess-授予對 Redis 資源的唯讀存取權限。

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "memorydb:Describe*",
   "memorydb:List*"
  ],
  "Resource": "*"
 }]
}
```

AmazonMemory資料庫FullAccess

您可將 AmazonMemoryDBFullAccess 政策連接到 IAM 身分。此政策授予允許完整存取 MemoryDB 資源的管理許可。

AmazonMemoryDBFullAccess-授予對 Redis 資源的內存數據庫的完全訪問權限。

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

您也可以建立自己的自訂 IAM 政策，以允許 Redis API 動作的相關許可。您可以將這些自訂政策連接至需要這些許可的 IAM 使用者或群組。

AWS受管理原則的記憶體資料庫更新

檢視自 MemoryDB 開始追蹤AWS受管政策變更以來的更新詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 MemoryDB 文件歷史記錄頁面上的 RSS 摘要。

變更	描述	日期
AmazonMemory資料庫FullAccess— 新增政策	MemoryDB 已新增描述及列出支援資源的新許可。MemoryDB 需要這些許可，才能查詢帳戶中的所有支援資源。	10/07/2021
AmazonMemory資料庫ReadOnlyAccess— 新增政策	MemoryDB 已新增描述及列出支援資源的新許可。MemoryDB 需要這些許可，才能透過查詢帳戶中的所有支援資源來建立帳戶型應用程式。	10/07/2021
MemoryDB 開始追蹤變更	服務啟動	8/19/2021

記憶體資料庫 API 權限：動作、資源和條件參考

當您設定存取控制和寫入許可政策以附加至 IAM 政策 (以身分為基礎或以資源為基礎) 時，請使用下表作為參考。此表格會列出適用於 Redis API 作業的每個 MemoryDB，以及您可以授與執行動作之權限的對應動作。您需在政策的 `Action` 欄位中指定動作，然後在政策的 `Resource` 欄位中指定資源值。除非另有說明，否則資源為必要項目。某些欄位同時包含必要資源和選用資源。如果沒有資源 ARN，則政策中的資源為萬用字元 (*)。

> **Note**
>
> 若要指定動作，請使用後接 API 操作名稱的 `memorydb:` 字首 (例如，`memorydb:DescribeClusters`)。

日誌記錄和監控

監控是維護 Redis 和其他解決方案 MemoryDB 可靠性、可用性和效能的重要組成部分。 AWS AWS 提供以下監視工具來監視 MemoryDB，報告何時出現錯誤，並在適當時採取自動操作：

- Amazon 會即時 CloudWatch監控您的 AWS 資源和執行 AWS 的應用程式。您可以收集和追蹤指標、建立自訂儀板表，以及設定警示，在特定指標達到您指定的閾值時通知您或採取動作。例如，您可以 CloudWatch 追蹤 Amazon EC2 執行個體的 CPU 使用率或其他指標，並在需要時自動啟動新執行個體。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。
- Amazon CloudWatch 日誌可讓您從 Amazon EC2 執行個體和其他來源監控 CloudTrail、存放和存取日誌檔。 CloudWatch 記錄檔可以監控記錄檔中的資訊，並在符合特定臨界值時通知您。您也可以將日誌資料存檔在高耐用性的儲存空間。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南。
- AWS CloudTrail擷取您帳戶或代表您 AWS 帳戶發出的 API 呼叫和相關事件，並將日誌檔傳送到您指定的 Amazon S3 儲存貯體。您可以識別呼叫的使用者和帳戶 AWS、進行呼叫的來源 IP 位址，以及呼叫發生的時間。如需詳細資訊，請參閱 AWS CloudTrail 使用者指南。

使用 Amazon 監控 Redis 的內存數據庫 CloudWatch

您可以使用 Redis 監視 MemoryDB CloudWatch，該數據收集原始數據並將其處理為可讀的近實時指標。這些統計資料會保留 15 個月，以便您存取歷史資訊，並更清楚 Web 應用程式或服務的執行效能。您也可以設定留意特定閾值的警示，當滿足這些閾值時傳送通知或採取動作。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。

以下段落列出 MemoryDB 的測量結果和維度。

主題

- 主機層級指標
- 記憶體資料庫的度量
- 應監控哪些指標？
- 選擇指標統計資料與期間
- 監控 CloudWatch 指標

主機層級指標

AWS/MemoryDB命名空間包括個別節點的下列主機層級測量結果。

另請參閱

- 記憶體資料庫的度量

指標	描述	單位
CPUUtilization	整部主機的 CPU 使用率百分比。由於 Redis 是單執行緒，因此我們建議您監控具有 4 個或更多 vCPUs 的節點EngineCPUUtilization 指標。	百分比
FreeableMemory	主機上可用的記憶體數量。這是從 RAM，緩衝區派生的，並且操作系統報告為可用的。	位元組
NetworkBytesIn	主機已從網路讀取的位元組數。	位元組
NetworkBytesOut	執行個體在所有網路界面上送出的位元組數目。	位元組
NetworkPacketsIn	執行個體在所有網路界面上收到的封包數目。此指標識別單一執行個體上的傳入流量 (封包數目)。	計數

指標	描述	單位
NetworkPacketsOut	執行個體在所有網路界面上送出的封包數目。此指標識別單一執行個體上的傳出流量 (封包數目)。	計數
NetworkBandwidthInAllowanceExceeded	因傳入的彙總頻寬超過執行個體的上限而成形的封包數目。	計數
NetworkConntrackAllowanceExceeded	因為連線追蹤超過執行個體的上限且無法建立新的連線，而成形的封包數目。這可能會導致傳送或傳回執行個體流量的封包遺失。	計數
NetworkBandwidthOutAllowanceExceeded	因傳出的彙總頻寬超過執行個體的上限而成形的封包數目。	計數
NetworkPacketsPerSecondAllowanceExceeded	因雙向每秒封包數超過執行個體的上限而成形的封包數目。	計數
NetworkMaxBytesIn	每分鐘內接收到的最大位元組高載。	位元組
NetworkMaxBytesOut	每分鐘內傳輸的最大位元組高載。	位元組
NetworkMaxPacketsIn	每分鐘內接收到的最大封包高載。	計數
NetworkMaxPacketsOut	每分鐘內傳輸的最大封包高載。	計數
SwapUsage	主機已使用的交換空間的量。	位元組

記憶體資料庫的度量

AWS/MemoryDB 命名空間包含下列 Redis 指標。

但不包括 ReplicationLag 和 EngineCPUUtilization，這些指標產生自 Redis info 命令。每個量度都是在節點層級計算。

如需 Redis info 命令的完整文件，請參閱 http://redis.io/commands/info。

另請參閱

- 主機層級指標

指標	描述	單位
ActiveDefragHits	作用中重組程序每分鐘執行的值重新配置次數。這衍生自 Redis INFO 的 active_defrag_hits 統計數字。	Number
AuthenticationFailures	嘗試使用 AUTH 命令向 Redis 驗證失敗的總次數。如需個別身分驗證失敗的詳細資訊，請使用 ACL LOG 命令。建議對此設定警示，以偵測未經授權的存取嘗試。	計數
BytesUsedForMemoryDB	MemoryDB 為所有用途 (包括資料集、緩衝區等) 配置的位元組總數。	位元組
	Dimension: Tier=SSD 針對使用的叢集資料分層：SSD 使用的位元組總數。	位元組
	Dimension: Tier=Memory 針對使用的叢集資料分層：記憶體使用的位元組總數。這是 Redis INFO 的 used_memory 統計資料值。	位元組
BytesReadFromDisk	每分鐘從磁碟讀取的位元組總數。僅支援使用資料分層 的叢集。	位元組
BytesWrittenToDisk	每分鐘寫入磁碟的位元組總數。僅支援使用 資料分層 的叢集。	位元組
CommandAuthorizationFailures	使用者嘗試執行他們沒有呼叫許可的命令失敗總次數。如需個別身分驗證失敗的詳細資訊，請使用 ACL LOG 命令。建議對此設定警示，以偵測未經授權的存取嘗試。	計數
CurrConnections	用戶端連線數，不包含僅供讀取複本的連線。MemoryDB 使用兩到四個連接來監視在每種情況下的集群。這衍生自 Redis INFO 的 connected_clients 統計數字。	計數

指標	描述	單位
CurrItems	快取中的項目數。這透過加總整個金鑰空間中的所有金鑰而衍生自 Redis keyspace 統計資訊。	計數
	Dimension: Tier=Memory 適用於使用 資料分層 的叢集。記憶體中的項目數。	計數
	Dimension: Tier=SSD (固態硬碟) 適用於使用 資料分層 的叢集。SSD 中的項目數。	計數
DatabaseMemoryUsagePercentage	可供使用中叢集使用之記憶體的百分比。使用來自 Redis INFO 的 used_memory/maxmemory 計算得出。	百分比
DB0AverageTTL	從 Redis INFO 命令的 keyspace 統計數字中呈現 DBO 的 avg_ttl。	毫秒

指標	描述	單位
EngineCPUUtilization	提供 Redis 引擎執行緒的 CPU 使用率。因為 Redis 是以單一執行緒運作，所以您可使用此指標來分析 Redis 程序本身的負載。EngineCPUUtilization 指標可提供更精確的 Redis 程序可見性。您可以用來搭配 CPUUtilization 指標，CPUUtilization 會呈現整體伺服器執行個體的 CPU 使用率，包括其他作業系統與管理程序。對於具有 4 個或以上 vCPU 的大型節點類型，請使用 EngineCPUUtilization 指標來監控擴展並設定閾值。 **Note** 在 MemoryDB 主機上，背景處理序會監視主機，以提供受管理的資料庫體驗。這些背景處理程序可能會佔用大部分的 CPU 工作負載。在具有 2 個以上 vCPU 的大型主機上，這並不重要。但它可能會影響具有 2vCPU 或更少的較小主機。如果您只監視 EngineCPUUtilization 指標，則無法察覺主機超載的情況，Redis 的高 CPU 使用率，以及背景監視程序的高 CPU 使用率。因此，建議您針對具有 2 個 vCPU 或更少的主機監控 CPUUtilization 指標。	百分比
Evictions	因 maxmemory 限制而移出的金鑰數目。這衍生自 Redis INFO 的 evicted_keys 統計數字。	計數

指標	描述	單位
IsPrimary	指示節點是否為當前分片的主節點。指標可能是 0 (非主要) 或 1 (主要)。	計數
KeyAuthorizationFailures	使用者嘗試存取他們沒有存取許可的金鑰失敗總次數。如需個別身分驗證失敗的詳細資訊，請使用 ACL LOG 命令。建議對此設定警示，以偵測未經授權的存取嘗試。	計數
KeyspaceHits	主字典中的成功唯讀索引鍵查詢次數。這衍生自 Redis INFO 的 keyspace_hits 統計數字。	計數
KeyspaceMisses	主字典中的未成功唯讀索引鍵查詢次數。這衍生自 Redis INFO 的 keyspace_misses 統計數字。	計數
KeysTracked	Redis 金鑰所追蹤的金鑰數，以 tracking-table-max-keys 的百分比呈現。金鑰追蹤用來協助用戶端快取，並在金鑰修改時通知用戶端。	計數
MaxReplicationThroughput	最後一個測量週期中觀察到的最大複製輸送量。	每秒位元組數
MemoryFragmentationRatio	表示 Redis 引擎記憶體配置的效率。某些閾值表示不同的行為。建議的值是具有 1.0 以上的片段。從 Redis INFO 的 mem_fragmentation_ratio statistic 計算得出。	Number
NewConnections	在此期間內，伺服器已接受的連線總數。這衍生自 Redis INFO 的 total_connections_received 統計數字。	計數
NumItemsReadFromDisk	每分鐘從磁碟檢索的項目總數。僅支援使用 資料分層 的叢集。	計數

指標	描述	單位
NumItemsWrittenToDisk	每分鐘寫入磁碟的項目總數。僅支援使用 資料分層 的叢集。	計數
PrimaryLinkHealthStatus	此狀態有兩個值：0 或 1。值 0 表示記憶體資料庫主節點中的資料與 EC2 上的 Redis 不同步。值為 1 表示資料同步。	Boolean
Reclaimed	金鑰過期事件總數。這衍生自 Redis INFO 的 expired_keys 統計數字。	計數
ReplicationBytes	針對複寫組態中的節點，ReplicationBytes 會報告主節點傳送給其所有複本的位元組數。此測量結果代表叢集上的寫入負載。這衍生自 Redis INFO 的 master_repl_offset 統計數字。	位元組
ReplicationDelayedWriteCommands	因同步複寫而延遲的寫入命令數目。複寫可能會因為各種因素而延遲，例如網路壅塞或超過最大複寫輸送量。	計數
ReplicationLag	此指標僅適用於以讀取複本形式執行的節點。它代表複本要多久的時間 (秒) 才會套用主要節點變更。	秒鐘

這些是來自 info commandstats 的特定命令類型彙整。Commanstats 段落提供以命令類型為基礎的統計資料，包括呼叫次數。

如需可用命令的完整清單，請參閱 Redis 說明文件中的 redis 命令。

指標	描述	單位
EvalBasedCmds	以 eval 為基礎之命令的命令總數。這來自 Redis commandstats 統計資料。這衍生自 Redis 的 commandstats 統計數字，加總了 eval、evalsha。	計數

指標	描述	單位
GeoSpatialBasedCmds	以 geospatial- 為基礎的之命令的命令總數。這來自 Redis commandstats 統計資料。加總了下列 geo 類型的所有命令而得出：geoadd、geodist、geohash、geopos、georad 及 georadiusbymember。	計數
GetTypeCmds	read-only 類型命令的總數。這衍生自 Redis commandstats 統計數字，加總了 read-only 類型的所有命令 (get、hget、scard、lrange 等)	計數
HashBasedCmds	雜湊類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個雜湊的所有命令 (hget、hkeys、hvals、hdel 等)。	計數
HyperLogLogBasedCmds	以 HyperLogLog 為基礎的命令總數。這衍生自 Redis commandstats 統計數字，加總了 pf 類型的所有命令 (pfadd、pfcount、pfmerge 等)。	計數
JsonBasedCmds	JSON 類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個 JSON 文件物件的所有命令。	計數
KeyBasedCmds	金鑰類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於多個資料結構間一或多個索引鍵的所有命令 (del、expire、rename 等)。	計數
ListBasedCmds	清單類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個清單的所有命令 (lindex、lrange、lpush、ltrim 等)。	計數

指標	描述	單位
PubSubBasedCmds	pub/sub 功能的命令總數。這是由 Redis commandstats 統計資料衍生出來的，方法是總結用於 pub/sub 功能的所有命令：psubscribe、、、publishpubsub、punsubscribe和。subscribe unsubscribe	計數
SearchBasedCmds	次要索引和搜尋指令的總數，包括讀取和寫入指令。這是透過對次要索commandstats 引作用的所有搜尋命令求和衍生自 Redis 統計資料。	計數
SearchBasedGetCmds	次要索引和搜尋唯讀指令的總數。這是通過求和所有二級索引和搜索 get 命令衍生自 Redis 的commandstats 統計信息。	計數
SearchBasedSetCmds	次要索引和搜尋寫入指令的總數。這是透過求和所有次要索引和搜尋集命令衍生自 Redis commandstats 統計資料。	計數
SearchNumberOfIndexes	索引總數。	計數
SearchNumberOfIndexedKeys	已索引的 Redis 金鑰總數	計數
SearchTotalIndexSize	所有索引使用的內存（字節）。	位元組
SetBasedCmds	集合類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個組合的所有命令 (scard、sdiff、sadd、sunion 等)。	計數
SetTypeCmds	write 類型命令的總數。這衍生自 Redis commandstats 統計數字，加總了對資料執行的所有 mutative 類型命令 (set、hset、sadd、lpop 等)	計數

指標	描述	單位
SortedSetBasedCmds	有序集合類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個排序組的所有命令 (zcount、zrange、zrank、zadd 等)。	計數
StringBasedCmds	字串類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個字串的所有命令 (strlen、setex、setrange 等)。	計數
StreamBasedCmds	串流類型命令總數。這衍生自 Redis commandstats 統計數字，加總了作用於一或多個串流資料類型的所有命令 (xrange、xlen、xadd、xdel 等)。	計數

應監控哪些指標？

以下 CloudWatch 指標提供了對 MemoryDB 性能的良好洞察力。在大多數情況下，我們建議您為這些測量結果設定 CloudWatch 警示，以便在發生效能問題之前採取更正動作。

要監控的指標

- CPUUtilization
- EngineCPUUtilization
- SwapUsage
- 移出
- CurrConnections
- 記憶體
- 網路
- 複寫

CPUUtilization

此為主機層級指標，以百分比報告。如需詳細資訊，請參閱 主機層級指標。

對於具有 2 個或以下 vCPU 的小型節點類型，請使用 `CPUUtilization` 指標來監控工作負載。

一般而言，我們建議您將閾值設為您可用 CPU 的 90%。因為 Redis 為單執行緒，實際閾值應以節點總容量的分數計算。例如，假設您使用擁有二核心的節點類型。在此情況下，CPUUtilization 的閾值將為 90/2 或 45%。若要查看節點類型具有的核心 (vCPUs) 數量，請參閱 Memory DB 定價。

您將需要根據所使用節點中的核心數量來決定自己的臨界值。如果您超過此閾值，且主要工作負載來自讀取請求，請透過新增僅供讀取複本來向外擴展叢集。如果主要工作負載來自寫入請求，我們建議您新增更多碎片，以便將寫入工作負載分配到更多主要節點。

> **Tip**
>
> 您可能可以使用 Redis 量度`CPUUtilization`，而不是使用主機層級量度`EngineCPUUtilization`，該指標會報告 Redis 引擎核心上的使用百分比。若要查看節點上是否提供此測量結果，以及如需詳細資訊，請參閱 MemoryDB 的測量結果。

對於具有 4 個或以上 vCPU 的大型節點類型，您可使用 EngineCPUUtilization 指標來報告 Redis 引擎核心上的用量百分比。若要查看節點上是否提供此測量結果，以及如需詳細資訊，請參閱 MemoryDB 的測量結果。

EngineCPUUtilization

對於具有 4 個或以上 vCPU 的大型節點類型，您可使用 EngineCPUUtilization 指標來報告 Redis 引擎核心上的用量百分比。若要查看節點上是否提供此測量結果，以及如需詳細資訊，請參閱 MemoryDB 的測量結果。

SwapUsage

此為主機層級指標，以位元組報告。如需詳細資訊，請參閱 主機層級指標。

此指標不應超過 50 MB。

移出

這是引擎指標。建議您根據應用程式需求，親自判斷此指標的警示閾值。

CurrConnections

這是引擎指標。建議您根據應用程式需求，親自判斷此指標的警示閾值。

越來越多的CurrConnections可能表示您的應用程式發生問題；您必須調查應用程式行為以解決此問題。

記憶體

記憶體 Redis 的核心要素。為避免資料遺失以及因應資料集的未來成長而調整，了解叢集的記憶體使用率是必要的。節點的記憶體使用率統計數字可在 Redis INFO 命令的記憶體段中查看。

網路

叢集網路頻寬容量的決定因素之一，是您選取的節點類型。如需有關節點網路容量的詳細資訊，請參閱 Amazon MemoryDB 定價。

複寫

遭複寫的資料量可透過 ReplicationBytes 指標顯示。您可以MaxReplicationThroughput根據複寫容量輸送量進行監視。建議在達到最大複寫容量輸送量時新增更多碎片。

ReplicationDelayedWriteCommands也可以指出工作負載是否超過最大複寫容量輸送量。如需有關 MemoryDB 中複寫的詳細資訊，請參閱瞭解記憶體資料庫複寫

選擇指標統計資料與期間

雖然可 CloudWatch 讓您為每個指標選擇任何統計資料和期間，但並非所有組合都有用。例如，CPUUtilization 的平均值 (Average)、最小值 (Minimum)、最大值 (Maximum) 統計資料相當有用，但總和 (Sum) 統計資料則否。

所有 MemoryDB 樣本都會針對每個單獨的節點發佈 60 秒的持續時間。對於任何 60 秒期間，節點指標只會包含一個樣本。

監控 CloudWatch 指標

MemoryDB 和集成 CloudWatch ，因此您可以收集各種指標。您可以使用監視這些指標 CloudWatch。

> **Note**
>
> 下列範例需要命 CloudWatch 令列工具。如需開發人員工具的詳細資訊 CloudWatch 和下載，請參閱 CloudWatch 產品頁面。

下列程序說明如何使用 CloudWatch 收集過去一小時叢集的儲存空間統計資料。

> **Note**
>
> 以下範例中提供的`StartTime`和`EndTime`值僅供說明之用。請務必為節點取代適當的開始和結束時間值。

如需有關記憶體 DB 限制的資訊，請參閱 MemoryDB 的AWS 服務限制。

監視測 CloudWatch 量結果 (主控台)

收集叢集的 CPU 使用率統計資料

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 選取您要檢視其測量結果的節點。

   > **Note**
   >
   > 選取 20 個以上的節點時，會停用主控台上的指標檢視。

a. 在 AWS 管理主控台的 [叢集] 頁面上，按一下一或多個叢集的名稱。

   便會顯示叢集的詳細資訊頁面。

b. 按一下視窗頂端的 Nodes (節點) 標籤。

c. 在詳細資訊視窗的「節點」頁籤上，選取您要檢視其測量結果的節點。

   可用的 CloudWatch 測量結果清單會顯示在主控台視窗的底部。

d. 按一下 CPU Utilization (CPU 使用率) 指標。

   主 CloudWatch 控台隨即開啟，並顯示您選取的量度。若要變更顯示的指標，可以使用 Statistic (統計數字) 和 Period (期間) 下拉式清單方塊和 Time Range (時間範圍) 索引標籤。

使用 CloudWatch CLI CloudWatch 監控指標

收集叢集的 CPU 使用率統計資料

- 將 CloudWatch 指令aws cloudwatch get-metric-statistics與下列參數搭配使用 (請注意，開始時間和結束時間僅顯示為範例；您需要取代您自己適當的開始和結束時間)：

  若為 Linux、macOS 或 Unix：

```
aws cloudwatch get-metric-statistics CPUUtilization \
    --dimensions=ClusterName=mycluster,NodeId=0002" \
    --statistics=Average \
    --namespace="AWS/MemoryDB" \
    --start-time 2013-07-05T00:00:00 \
    --end-time 2013-07-06T00:00:00 \
    --period=60
```

  針對 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
```

使用 CloudWatch CloudWatch API 監控指標

收集叢集的 CPU 使用率統計資料

- GetMetricStatistics使用下列參數呼叫 CloudWatch API (請注意，開始時間和結束時間僅顯示為範例；您必須替換自己適當的開始和結束時間)：
  - 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>
```

監視記憶體資料庫的 Redis 事件

當叢集發生重大事件時，MemoryDB 會將通知傳送至特定的 Amazon SNS 主題。範例包含新增節點失敗、新增節點成功、安全群組修改和其他事件。藉由監控重要事件，您可以了解叢集目前的狀態，並根據事件採取正確的動作。

主題

- 管理記憶體資料庫 Amazon SNS 通知
- 檢視記憶體資料庫事件
- 事件通知和 Amazon SNS

管理記憶體資料庫 Amazon SNS 通知

您可以將 MemoryDB 設定為使用亞馬遜簡單通知服務 (Amazon SNS) 傳送重要叢集事件的通知。在這些範例中，您會使用 Amazon SNS 主題的 Amazon Resource Name (ARN) 設定叢集以接收通知。

> **Note**
>
> 本主題假設您已註冊 Amazon SNS，並已設定及訂閱 Amazon SNS 主題。如需操作方式的相關資訊，請參閱 Amazon Simple Notification Service 開發人員指南。

新增 Amazon SNS 主題

以下各節說明如何使用主 AWS 控台、或 MemoryDB API 新增 Amazon SNS 主題。 AWS CLI

新增 Amazon SNS 主題 (主控台)

下列程序示範如何為叢集新增 Amazon SNS 主題。

> **Note**
>
> 此程序也可用於修改 Amazon SNS 主題。

為叢集新增或修改 Amazon SNS 主題 (主控台)

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。

2. 在 Clusters (叢集) 中，選擇您要新增或修改 Amazon SNS 主題 ARN 的叢集。
3. 選擇 Modify (修改)。
4. 在 Modify Cluster (修改叢集) 的 Topic for SNS Notification (SNS 通知的主題) 下，選擇您要新增的 SNS 主題，或選擇 Manual ARN input (手動輸入 ARN)，並輸入 Amazon SNS 主題的 ARN。
5. 選擇 Modify (修改)。

新增 Amazon SNS 主題 (AWS CLI)

若要新增或修改叢集的 Amazon SNS 主題，請使用 AWS CLI 指令`update-cluster`。

下列程式碼範例會將 Amazon SNS 主題 ARN 新增至 my-cluster。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --sns-topic-arn arn:aws:sns:us-east-1:565419523791:memorydbNotifications
```

如需詳細資訊，請參閱UpdateCluster。

新增 Amazon SNS 主題 (記憶體資料庫 API)

若要新增或更新叢集的 Amazon SNS 主題，請使用下列參數呼叫`UpdateCluster`動作：

- `ClusterName=my-cluster`
- `SnsTopicArn=arn%3Aaws%3Asns%3Aus-east-1%3A565419523791%3AmemorydbNotifications`

若要新增或更新叢集的 Amazon SNS 主題，請呼叫`UpdateCluster`動作。

如需詳細資訊，請參閱UpdateCluster。

啟用和停用 Amazon SNS 通知

您可以為叢集開啟或關閉通知。下列程序示範如何停用 Amazon SNS 通知。

啟用和停用 Amazon SNS 通知 (主控台)

若要停用 Amazon SNS 通知，請使用 AWS Management Console

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 選擇您要修改通知的叢集左側的圓鈕。
3. 選擇 Modify (修改)。
4. 在 Modify Cluster (修改叢集) 的 Topic for SNS Notification (SNS 通知的主題) 下，選擇 Disable Notifications (停用通知)。
5. 選擇 Modify (修改)。

啟用和停用 Amazon SNS 通知 (AWS CLI)

若要停用 Amazon SNS 通知，請搭配下列參數使用 `update-cluster` 命令：

若為 Linux、macOS 或 Unix：

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --sns-topic-status inactive
```

針對 Windows：

```
aws memorydb update-cluster ^
    --cluster-name my-cluster ^
    --sns-topic-status inactive
```

啟用和停用 Amazon SNS 通知 (記憶體資料庫 API)

若要停用 Amazon SNS 通知，請搭配下列參數呼叫 `UpdateCluster` 動作：

- `ClusterName=my-cluster`
- `SnsTopicStatus=inactive`

此呼叫會傳回類似以下的輸出：

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>
```

檢視記憶體資料庫事件

MemoryDB 會記錄與叢集、安全群組和參數群組相關的事件。此資訊包含事件的日期和時間、事件的來源名稱和來源類型，以及事件的描述。您可以使用 MemoryDB 主控台、 AWS CLI `describe-events`命令或 MemoryDB API 動作，輕鬆擷取記錄檔中的事件。`DescribeEvents`

下列程序說明如何檢視過去 24 小時 (1440 分鐘) 的所有 MemoryDB 事件。

檢視記憶體資料庫事件 (主控台)

下列程序會顯示使用 MemoryDB 主控台的事件。

若要使用記憶體資料庫主控台檢視事件

1. 登入 AWS Management Console 並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇 [事件]。

   會出現「事件」畫面，列出所有可用事件。清單中的每一列代表一個事件，並顯示事件來源、事件類型 (例如叢集、參數群組、acl、安全性群組或子網路群組)、事件的 GMT 時間以及事件說明。

   您可以使用 Filter (篩選條件) 指定要查看事件清單中的所有事件，還是只查看特定類型的事件。

檢視記憶體資料庫事件 (CLI)AWS

若要使用產生 MemoryDB 事件的清單 AWS CLI，請使用指令。`describe-events`您可以使用選用參數來控制列出的事件類型、列出的事件時間範圍，要列出的最大事件數等等。

下列程式碼最多可列出 40 個叢集事件。

```
aws memorydb describe-events --source-type cluster --max-results 40
```

下列程式碼會列出過去 24 小時 (1440 分鐘) 內的所有事件。

```
aws memorydb describe-events --duration 1440
```

`describe-events` 命令的輸出會類似下列內容。

```
{
```

```
    "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"
        }
    ]
}
```

如需可用參數和允許參數值這類項目的詳細資訊，請參閱 describe-events。

檢視記憶體資料庫事件 (記憶體資料庫 API)

若要使用記憶體資料庫 API 產生記憶體資料庫事件清單，請使用動作。DescribeEvents您可以使用選用參數來控制列出的事件類型、列出的事件時間範圍，要列出的最大事件數等等。

下列程式碼會列出 40 個最新叢集事件。

```
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>
```

下列程式碼會列出過去 24 小時 (1440 分鐘) 的叢集事件。

```
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>
```

上述動作產生的輸出應該會類似下列內容。

```
<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>
```

如需可用參數和允許參數值這類項目的詳細資訊，請參閱 DescribeEvents。

事件通知和 Amazon SNS

當叢集上發生重要事件時，MemoryDB 可以使用 Amazon Simple Notification Service (SNS) 來發佈訊息。此功能可用於重新整理用户端機器上的伺服器清單，這些機器連線至叢集的個別節點端點端點。

> **Note**
>
> 如需 Amazon Simple Notification Service (SNS) 的詳細資訊，包含定價資訊和 Amazon SNS 說明文件的連結，請參閱 Amazon SNS 產品頁面。

通知會發佈至特定的 Amazon SNS 主題。下列是通知的需求：

- 用於內存數據庫通知，只能設定一個主題。
- 所以此AWS擁有 Amazon SNS 主題的帳户必須與擁有已啟用通知的叢集的帳户相同。

內存 DB 事件

以下內存數據庫事件觸發 Amazon SNS 通知：

事件名稱	Message	描述
內存 DB: 添加節點完成	`"Modified number of nodes from %d to %d"`	已將節點添加至叢集，且已就緒可供使用。
由於可用的 IP 地址不足，而導致的內存 DB：	`"Failed to modify number of nodes from %d to %d due to insufficient free IP addresses"`	由於可用的 IP 地址不足，而無法新增節點。
內存 DB: 羣集參數已更改	`"Updated parameter group for the cluster"` 在建立時，也傳送 `"Updated to use a Parameter Group %s"`	一或多個叢集參數已變更。
內存 DB: 羣集設置完成	`"Cluster created."`	叢集的設定已完成，且叢集中的節點已就緒可供使用。
內存 DB: 由於不相容的網路狀態，而導致:由於不相容的網路狀態,	`"Failed to create cluster due to incompatible network state. %s"`	嘗試將新叢集啟動至不存在的虛擬私有雲端 (VPC)。
內存 DB: 羣集存儲失敗	`"Restore from %s failed for node %s. %s"`	內存數據庫無法使用 Redis 快照數據填充集羣。這可能是由於 Amazon S3 中不存在快照文件，或者該文件的權限不正確。如果您描述叢集，其狀態

事件名稱	Message	描述
		將是`restore-failed` 。您將需要刪除叢集並重新開始。 如需詳細資訊，請參閱 使用外部建立的快照植入新叢集。
內存 DB: 羣集擴展完成	`"Succeeded applying modification to node type to %s."`	已順利完成叢集的向上擴展。
內存 DB: 羣集擴展失敗	`"Failed applying modification to node type to %s."`	叢集上的向上擴展操作失敗。
內存 DB: 羣集安全性組修改	`"Modified security group for cluster."`	已發生下列其中一項事件： • 已修改用於叢集的安全組清單。 • 一或多個新 EC2 安全組，已在與叢集相關聯的任何安全組上授權。 • 一或多個新 EC2 安全組，已從與叢集相關聯的任何安全組上撤銷。

事件名稱	Message	描述
內存 DB: 節點替換	`"Recovering node %s"`	MemoryDB 已檢測到執行快取節點的主機已降級或無法連線，且已開始取代快取節點。 **Note** 取代之快取節點的 DNS 項目並未變更。 在多數情況下，當發生此事件時，您不需要重新整理用戶端的伺服器清單。然而，某些用户端程式庫可能會在 MemeryDB 取代節點後停止使用該節點；在此情況下，應用程式應在此事件發生時重新整理伺服器清單。

事件名稱	Message	描述
內存 DB: 節點替換組合	`"Finished recovery for node %s"`	MemoryDB 已檢測到執行快取節點的主機已降級或無法連線，且已完成取代快取節點。 **Note** 取代之快取節點的 DNS 項目並未變更。 在多數情況下，當發生此事件時，您不需要重新整理用戶端的伺服器清單。然而，某些用户端程式庫可能會在 MemeryDB 取代節點後停止使用該節點；在此情況下，應用程式應在此事件發生時重新整理伺服器清單。
內存 DB: 創建羣集完成	`"Cluster created"`	已成功建立叢集。
內存 DB: 創建羣集失敗	`"Failed to create cluster due to unsuccessful creation of its node(s)."` 與 `"Deleting all nodes belonging to this cluster."`	未創建羣集。
內存 DB: 刪除羣集完成	`"Cluster deleted."`	已完成刪除叢集和所有與其相關聯的節點。
內存 DB: 故障切換完成	`"Failover to replica node %s completed"`	已成功容錯移轉至複本節點。

事件名稱	Message	描述
內存 DB: 節點替換已取消	"The replacement of node %s which was scheduled during the maintenance window from start time: %s, end time: %s has been canceled"	叢集中原先已排程替換的節點，已不再排程替換。
內存 DB: 節點替換已重新計劃	"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"	叢集中原先已排程替換的節點，已重新排程在通知中所述的新視窗期間進行替換。 如需您可以採取哪些動作的資訊，請參閱替換節點。
內存 DB: 節點替換計劃	"The node %s is scheduled for replacement during the maintenance window from start time: %s to end time: %s"	您叢集中的節點，已排程在通知中所述的視窗期間進行替換。 如需您可以採取哪些動作的資訊，請參閱替換節點。
內存 DB: 刪除已完成	"Removed node %s"	已將節點從叢集移除。
內存 DB: 快照完成	"Snapshot %s succeeded for node %s"	快照已成功完成。

事件名稱	Message	描述
內存 DB: 快照失敗	`"Snapshot %s failed for node %s"`	快照已失敗。請查看叢集的事件，以取得失敗原因的詳細資訊。 如果您描述快照，請查看DescribeSnapshots，則狀態將為`failed`。

記錄內存數據庫的 Redis API 調用AWS CloudTrail

MemoryDB inAWS 為 RedisAWS CloudTrail CloudTrail 將 Redis 的所有 API 呼叫擷取為事件，包括來自對 Redis 的 Redis 主控台的 MemoryDB is 如果您建立追蹤記錄，就可將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 Redis 的 MemoryDB (事件) 持續交付到 Amazon S3 儲存貯體。即使您未設定追蹤記錄，依然可以透過 CloudTrail 主控台中的 Event history (事件歷史記錄) 檢視最新事件。您可以利用所 CloudTrail收集的資訊來判斷對 Redis 的 Redis 發出的請求，以及發出請求的 IP 地址、人員、時間和其他詳細資訊。

若要進一步了解 CloudTrail，請參閱使AWS CloudTrail用者指南。

CloudTrail

CloudTrail 當您建立AWS帳戶時，系統會在您的帳戶中啟用。當活動發生於 Redis 的 MemoryDB in is 時，將該活動與 CloudTrail Event history (事件歷史記錄) 中的其他AWS服務事件一起記錄在事件中。您可以檢視、搜尋和下載 AWS 帳戶的最新事件。如需詳細資訊，請參閱使用 CloudTrail 事件歷史記錄檢視事件。

如需AWS帳戶中正在進行事件的記錄 (包含 追蹤能 CloudTrail 將日誌檔交付至 Amazon S3 儲存貯體。根據預設，當您在主控台建立權杖時，權杖會套用到所有區域。線索會記錄來自 AWS 分割區中所有區域的事件，然後將所有日誌檔案交付到您指定的 Amazon S3 儲存貯體。此外，您還能設定其他AWS服務，以進一步分析和處理 CloudTrail 日誌中收集的事件資料。如需詳細資訊，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務與整合
- 設定的 Amazon SNS 通知 CloudTrail
- 接收多個區域的 CloudTrail 日誌檔案及接收多個帳戶的 CloudTrail 日誌檔案

Redis 的動作的所有記憶體資料庫都會記錄 CloudTrail。例如，呼叫`DescribeClusters`和`UpdateCluster`動作會`CreateCluster`在 CloudTrail 記錄檔中產生項目。

每一筆事件或日誌項目都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是否使用根或 IAM 使用者憑證提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail 使用者身分元素。

了解 Redis 日誌文件條目的內存數據庫

追蹤是一種組態，能讓事件以日誌檔案的形式交付至您指定的 Amazon S3 儲存貯體。 CloudTrail 日誌檔案包含一個或多個日誌項目。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。 CloudTrail 日誌檔案並非依公有 API 呼叫追蹤記錄的堆疊排序，因此不會以任何特定順序出現。

以下範例顯示的是展示`CreateCluster`動作的 CloudTrail 日誌項目。

```
{
    "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"
}
```

以下範例顯示的是展示`DescribeClusters`動作的 CloudTrail 日誌項目。請注意，對於 Redis 的描述和列表調用（`Describe*`和`List*`）的所有 MemoryDB，該`responseElements`部分被刪除並顯示為`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"
}
```

以下範例顯示的是記錄UpdateCluster動作的 CloudTrail 日誌項目。

```
{
    "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"
}
```

以下範例顯示的是展示CreateUser動作的 CloudTrail 日誌項目。請注意，對於包含敏感數據 Redis 的 MemoryDB 調用，該數據將在相應的 CloudTrail 事件中進行編輯，如下面的requestParameters部分。

```
{
    "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"
}
```


適用於 Redis 的記憶體資料庫的合規性驗證

協力廠商稽核人員會評估 Redis 適用於 MemoryDB 的安全性和合規性，作為多個合規性方案的一部分。AWS其中包括：

- 支付卡產業資料安全標準 (PCI DSS)。如需詳細資訊，請參閱 PCI DSS 。
- 健康保險流通與責任法案商業夥伴協議 (HIPAA BAA)。如需詳細資訊，請參閱 HIPAA 合規。
- 系統和組織控制 (SOC) 1、2、3。如需詳細資訊，請參閱 SOC。
- 聯邦風險和授權管理計劃（FedRAMP）中等。如需詳細資訊，請參閱 FedRAMP。
- ISO/IEC 27001:2013, 27017:2015, 27018:2019, 和 IEC 9001:2015. 如需詳細資訊，請參閱 AWSISO 和 CSA STAR 認證與服務。

如需特定合規計劃範圍內的 AWS 服務清單，請參閱合規計劃內的 AWS 服務。

您可使用 AWS Artifact 下載第三方稽核報告。如需詳細資訊，請參閱 AWS Artifact 中的下載報告。

使用 MemoryDB 時，您的合規責任取決於資料的敏感性、公司的合規目標以及適用的法律和法規。AWS提供下列資源以協助遵循法規：

- 安全與合規 Quick Start 指南：這些部署指南就在 AWS 上部署以安全及合規為重心之基準環境，討論架構考量並提供相關步驟。
- AWS 合規資源 – 這組手冊和指南可能適用於您的產業和位置。
- 使用AWS Config開發人員指南中的規則評估資源 — 評AWS Config估您的資源配置如何符合內部實踐，業界準則和法規。

- AWS Security Hub：此 AWS 服務可供您檢視 AWS 中的安全狀態，可助您檢查是否符合安全產業標準和最佳實務。
- AWS Audit Manager – 此 AWS 服務可協助您持續稽核 AWS 使用情況，以簡化管理風險與法規與業界標準的合規事宜。

Amazon MachyDB 中的基礎設施安全

MemoryDB 為受管服務，受到 Amazon Web Services：安AWS全程序概觀白皮書所述的全球網路安全程序所保護。

您可使用AWS發佈的 API 呼叫，透過網路存取。用戶端必須支援 Transport Layer Security (TLS) 1.2 或更新版本。建議使用 TLS 或更新版本。用戶端也必須支援具備完美轉送私密 (PFS) 的密碼套件，例如臨時 Diffie-Hellman (DHE) 或橢圓曲線臨時 Diffie-Hellman (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以使用 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

網際網路流量隱私權

MemoryDB for Redis 使用下列技術保護您的資料並保護資料免受未經授權的存取：

- 記憶數據庫和 Amazon VPC 說明安裝所需的安全群組類型。
- Redis API 和界面 VPC 端點 (AWS PrivateLink)可讓您在 VPC is API 端點之間建立私有連線，以符合 Redis API 端 is。
- 適用於 Redis 的記憶體資料庫中的身分識別和存取管理 用於授予和限制使用者、群組和角色的動作。

記憶數據庫和 Amazon VPC

Virtual Private Cloud (Amazon VPC) 服務會定義虛擬網路，與傳統資料中心幾乎一模一樣。使用 Amazon VPC 設定虛擬私有雲端 (VPC) 時，您可以選擇其 IP 地址範圍、建立子網路和設定路由表、網路閘道和安全設定。您也可以將叢集新增至虛擬網路，然後使用 Amazon VPC 安全群組來控制對該叢集的存取。

本節說明如何在 VPC 中手動設定 MemoryDB。本資訊的預期對象是想要更深入了解 MemoryDB 和 Amazon VPC 如何搭配運作的使用者。

主題

- 了解記憶體資料庫和 VPC
- 用於存取 Amazon VPC 中存取記憶資料庫叢集的存取模式
- 建立 Virtual Private Cloud (VPC)

了解記憶體資料庫和 VPC

MemoryDB 已與 Amazon VPC 完全整整整整整整整整整整整整整整整整整整整整 對 MemoryDB 使用者，這表示下列：

- MemoryDB 一律會在 VPC 中啟動您的叢集。
- 如果您是初次使用AWS，系統將為您建立預設的。
- 如果您有預設的 VPC，且在啟動叢集時未指定子網路，叢集會於預設 Amazon VPC 啟動。

如需詳細資訊，請參閱偵測支援的平台以及您是否有預設 VPC。

使用 Amazon VPC，您可以在中建立虛擬網路。AWS與傳統資料中心幾乎一模一樣。您可以設定您的 VPC，包括選取其 IP 地址範圍、建立子網路，以及設定路由表、網路閘道與安全設定。

MemoryDB 會管理軟體升級、修補、故障偵測與復原。

VPC 中的記憶體資料庫概觀

1	VPC 是一個隔離的部分AWS雲端獲指派自己的 IP 地址區塊。
2	網際網路閘道會將您的 VPC 直接連接至網際網路，並提供對其他網路的存取AWSAmazon Simple Storage Service (Amazon S3)) 在您的 VPC 外部執行。
3	Amazon VPC 子網路是 VPC IP 地址範圍的區段，您可以在其中隔離AWS根據您的安全和操作需求。
4	VPC 中的路由表會在子網路與網際網路之間導向網路流量。亞馬遜 VPC 有一個隱含的路由器。
5	Amazon VPC 安全群組會控制 MemoryDB 叢集和 Amazon EC2 執行個體的傳入和傳出流量。
6	您可以在子網路中啟動 MemoryDB。節點具有來自子網路之地址範圍的私有 IP 地址。
7	

	您也可以在子網路中啟動 Amazon EC2 執行個體。每個 Amazon EC2 執行個體具有來自子網路之地址範圍的私有 IP 地址。Amazon EC2 執行個體可以連線至相同子網路中的任何節點。
8	若要讓 VPC 中的 Amazon EC2 執行個體可透過網際網路連接，您必須為執行個體指派一個靜態的公有地址 (稱為彈性 IP 地址)。

先決條件

若要在 VPC 中建立 MemoryDB 叢集，您的 VPC 必須符合下列要求：

- 您的 VPC 必須允許非專用的 Amazon EC2 執行個體。您無法在設定為使用專用執行個體租用戶的 VPC 中使用 MemoryDB。
- 必須為您的 VPC 定義子網路。MemoryDB 會使用此子網路群組選取子網路和該子網路中的 IP 地址，以與您的節點建立關聯。
- 必須為您的 VPC 定義安全群組，或是可以使用預設提供的安全群組。
- 每個子網路的 CIDR 區塊必須夠大，才能為 MemoryDB 提供備用 IP 地址，以在維護活動期間使用。

路由和安全性

您可以在 VPC 中設定路由，以控制流量流動的位置 (例如，通往網際網路閘道或虛擬私有閘道)。透過網際網路閘道，您的 VPC 可以直接存取其他網路閘道AWS未在 VPC 中執行的資源。如果您選擇只有一個虛擬私有閘道具有您的組織區域網路的連線，您可以透過 VPN 路由網際網路入站流量，並使用本機安全政策和防火牆來控制傳入。在該情況下，您會在存取時衍生額外的頻寬費用AWS透過網際網路的資源。

您可以使用 Amazon VPC 安全群組來協助保護 Amazon VPC 中的記憶體資料庫和 Amazon EC2 執行個體。安全群組會在執行個體層級 (而非子網路層級) 以防火牆形式運作。

> **Note**
>
> 我們強烈建議您使用 DNS 名稱來連線至您的節點，因為基本的 IP 地址可能會隨時間變動。

Amazon VPC 文件

Amazon VPC 有專屬的一套文件，說明如何建立和使用 Amazon VPC。下表顯示 Amazon VPC 指南中可找到資訊的位置。

描述	文件
如何開始使用 Amazon VPC	Amazon VPC 入門
透過 AWS Management Console 使用 Amazon VPC	《Amazon VPC 使用者指南》
所有 Amazon VPC 命令的完整描述	Amazon EC2 命令列參考 (Amazon VPC 命令均列在 Amazon EC2 參考中)
Amazon VPC API 作業、資料類型和錯誤的完整描述	Amazon EC2 API 參考 (Amazon VPC API 作業均列在 Amazon EC2 參考中)
需要在選用的 IPsec VPN 連線中的您那一端設定閘道之網路管理員的資訊	什麼是 AWS Site-to-Site VPN？

如需更多 Amazon Virtual Private Cloud 的詳細資訊，請參閱 Amazon Virtual Private Cloud。

用於存取 Amazon VPC 中存取記憶資料庫叢集的存取模式

Redis 的 MemoryDB 支援下列在 Amazon VPC 中存取叢集的使用案例：

內容

- 存取與 Amazon EC2 執行個體位於相同 Amazon VPC 中的 MemoryDB 叢集
- 存取與 Amazon EC2 執行個體位於不同 Amazon VPC 中的 MemoryDB 叢集
  - 存取與 Amazon EC2 執行個體位於相同區域內不同 Amazon VPC 中的 MemoryDB 叢集
    - 使用 Transit Gateway
  - 存取與 Amazon EC2 執行個體位於不同區域內不同 Amazon VPC 中的 MemoryDB 叢集
    - 使用傳輸 VPC
- 從在客戶資料中執行的應用程式存取 MemoryDB 叢集
  - 使用 VPN 連線，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集
  - 使用 Direct Connect，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集

存取與 Amazon EC2 執行個體位於相同 Amazon VPC 中的 MemoryDB 叢集

最常見的使用案例是部署於 EC2 執行個體的應用程式時，需要連線至相同 VPC 中的叢集。

若要管理相同 VPC 中 EC2 執行個體與叢集之間的存取權限，最簡單的方式如下：

1. 為您的叢集建立 VPC 安全群組。此安全群組可用來限制叢集的存取權限。舉例來說，您可以為此安全群建置立自訂規則，允許使用您在建立自訂規則時指派給叢集的連接埠存取 TCP，並可建立您將用於存取叢集的 IP 位址。

   MemoryDB 叢集的預設連接埠是6379。
2. 為您的 EC2 執行個體建立 VPC 安全群組 (Web 和應用程式伺服器)。若有需要，此安全群組可允許透過 VPC 路由表存取網際網路上的 EC2 執行個體。舉例來說，您可以在此安全群組上設定規則，允許 TCP 透過連接埠 22 存取 EC2 執行個體。
3. 在您叢集的安全群組中建立自訂規則，允許來自您為 EC2 執行個體所建立之安全群組的連線要求。這樣做會允許安全群組的所有成員存取叢集。

在允許來自其他安全群組連線的 VPC 安全群組中建立規則

1. 登入 AWS 管理主控台，並前往 https://console.aws.amazon.com/vpc 開啟 Amazon VPC 主控台。

2. 在左導覽窗格中，選擇 Security Groups (安全群組)。
3. 選取或建立您將用於叢集的安全群組。在 Inbound Rules (傳入規則) 下方，選取 Edit Inbound Rules (編輯傳入規則)，然後選取 Add Rule (新增規則)。此安全群組將允許其他安全群組成員存取。
4. 從 Type (類型) 選擇 Custom TCP Rule (自訂 TCP 規則)。

   a. 針對 Port Range (連接埠範圍)，指定您在建立叢集時所使用的連接埠。

      MemoryDB 叢集的預設連接埠是6379。

   b. 在 Source (來源) 方塊中輸入安全群組的 ID。從清單中選取您將用於 Amazon EC2 執行個體的安全群組。
5. 完成後，請選擇 Save (儲存)。

存取與 Amazon EC2 執行個體位於不同 Amazon VPC 中的 MemoryDB 叢集

當您的叢集與您用來存取該叢集的 EC2 執行個體位於不同 VPC 中時，有幾種方式可存取叢集。如果叢集與 EC2 執行個體位於不同 VPC，但位於相同區域，您可以使用 VPC 對等連線。如果叢集與 EC2 執行個體位於不同區域，您可以在區域之間建立 VPN 連線。

主題

- 存取與 Amazon EC2 執行個體位於相同區域內不同 Amazon VPC 中的 MemoryDB 叢集
- 存取與 Amazon EC2 執行個體位於不同區域內不同 Amazon VPC 中的 MemoryDB 叢集

存取與 Amazon EC2 執行個體位於相同區域內不同 Amazon VPC 中的 MemoryDB 叢集

由相同區域不同 Amazon VPC 中的 Amazon EC2 執行個體存取的叢集 - VPC 對等互連連線

VPC 對等連線是指兩個 VPC 之間的網路連線，透過此機制，您就可以使用私有 IP 地址在兩者之間路由流量。這兩個 VPC 中的執行個體能彼此通訊，有如位於相同網路中一樣。您可以在自己的 Amazon VPC 之間建立 VPC 對等互連連線，或者與單一區域內其他 AWS 帳戶中的 Amazon VPC 建立連線。若要進一步了解 Amazon VPC 對等互連，請參閱 VPC 說明文件。

透過對等互連存取不同 Amazon VPC 中的叢集

1. 請確保兩個 VPC 沒有重疊的 IP 範圍，否則您將無法為其建立互連連線。
2. 為兩個 VPC 建立互連連線。如需詳細資訊，請參閱建立和接受 Amazon VPC 對等互連連線。

3. 更新您的路由表。如需詳細資訊，請參閱更新 VPC 互連連線的路由表
4. 修改 MemoryDB 叢集的安全群組，以允許來自互連 VPC 中應用程式安全群組的傳入連線。如需詳細資訊，請參閱參考對等 VPC 安全群組。

透過互連連線存取叢集，將產生額外的資料傳輸費用。

使用 Transit Gateway

Transit Gateway 可讓您在相同 AWS 區域內連接 VPC 和 VPN 連線，並在它們之間路由流量。Transit Gateway 可跨 AWS 帳戶運作，您可以使用 AWS Resource Access Manager 與其他帳戶共享您的 Transit Gateway。與另一個 AWS 帳戶共享傳輸閘道後，帳戶擁有者可以將其 VPC 連接至您的傳輸閘道。這些帳戶的使用者均可隨時刪除連接。

您可以在傳輸閘道上啟用多點傳送，然後建立傳輸閘道多點傳送網域，讓多點傳送流量可透過與網域相關聯的 VPC 連接，從多點傳送來源傳送至多點傳送群組成員。

您也可以在不同 AWS 區域的傳輸閘道間建立互連連線連接。這可讓您跨不同區域在傳輸閘道附件之間路由流量。

如需詳細資訊，請參閱傳輸閘道。

存取與 Amazon EC2 執行個體位於不同區域內不同 Amazon VPC 中的 MemoryDB 叢集

使用傳輸 VPC

對於將多個在地理上分散的 VPC 和遠端網路進行連線，取代 VPC 對等互連的替代常用策略是建立做為全球網路傳輸中心的傳輸 VPC。傳輸 VPC 會簡化網路管理，並最大程度減少連線多個 VPC 和遠端網路所需的連線數。此設計可以節省時間和精力並降低費用，因為實際上不具有在託管傳輸中樞建立實體存在或部署實體網路設備的傳統支出。

位於不同區域不同 VPC 之間的連線

建立傳輸 Amazon VPC 後，在某區域中「軸輻式」VPC 中部署的應用程式，可以連線至另一個區域中「軸輻式」VPC 中的 MemoryDB 叢集。

存取不同 AWS 區域不同 VPC 中的叢集

1. 部署傳輸 VPC 解決方案。如需詳細資訊，請參閱 AWS Transit Gateway。

2. 更新應用程式和 VPC 中的 VPC 路由表，以透過 VGW (虛擬私有閘道) 和 VPN 設備來路由流量。在使用邊界閘道協定 (BGP) 的動態路由情況下，您的路由可能會自動傳播。
3. 修改 MemoryDB 叢集的安全群組，以允許來自應用程式執行個體 IP 範圍的傳入連線。請注意，在此情況下，您將無法參考應用程式伺服器安全群組。

跨區域存取叢集將造成聯網延遲，並產生額外跨區域數據傳輸費。

從在客戶資料中執行的應用程式存取 MemoryDB 叢集

另一種可能的情況是混合架構，其中客戶資料中中客戶資料中用戶資料中的用戶端或應用程式可能需要存取 VPC 中的 MemoryDB 叢集。如果客戶 VPC 與資料中心之間具有透過 VPN 或 Direct Connect 的連線，則此情況也受支援。

主題

- 使用 VPN 連線，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集
- 使用 Direct Connect，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集

使用 VPN 連線，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集

從您的資料中心透過 VPN 連線至記憶資料庫

透過 VPN 連接從現場部署應用程式存取 VPC 中的叢集

1. 透過將硬體虛擬私有閘道新增至您的 VPC 來建立 VPN 連線。如需詳細資訊，請參閱將硬體虛擬私有閘道新增至您的 VPC。
2. 更新已部署 MemoryDB 叢集之子網路的 VPC 路由表，以允許來自現場部署應用程式伺服器的流量。在使用 BGP 的動態路由情況下，您的路由可能會自動傳播。
3. 修改 MemoryDB 叢集的安全群組，以允許來自現場部署應用程式伺服器的傳入連線。

透過 VPN 連接存取叢集將造成聯網延遲，並產生額外跨區域數據傳輸費。

使用 Direct Connect，從在客戶資料中心執行的應用程式存取 MemoryDB 叢集

透過 Direct Connect 從您的資料中心連線至 MemoryDB

使用 Direct Connect，從您網路執行的應用程式存取 MemoryDB 叢集

1. 建立 Direct Connect 連線。如需詳細資訊，請參閱 AWS Direct Connect 入門。
2. 修改 MemoryDB 叢集的安全群組，以允許來自現場部署應用程式伺服器的傳入連線。

透過 DX 連線存取叢集可能造成聯網延遲，並產生額外跨區域數據傳輸費。

建立 Virtual Private Cloud (VPC)

在本範例中，您會根據 Amazon VPC 服務建立虛擬私有雲端 (VPC)，並為每個可用區域具有私有子網路。

建立 VPC (主控台)

在 Amazon Virtual Private Cloud 內建立 MemoryDB 叢集

1. 登入 AWS 管理主控台，開啟位於 https://console.aws.amazon.com/vpc/ 的 Amazon VPC 主控台。
2. 在 VPC 儀表板中，選擇建立 VPC。
3. 在 Resources to create (建立資源) 下，選擇 VPC and more (VPC 等)。
4. UNTER可用區域的數量 (AZ)」下，選擇可讓您選擇可用區域數，以在其中啟動子網路。
5. UNTER公有子網路的數量」下方，選擇您要新增至 VPC 的公用子網路數目。
6. UNTER私有子網路的數量」下方，選擇您要新增至 VPC 的私人子網路數目。

   > **Tip**
   >
   > 記下您的子網路識別符，它是公有和私有。之後啟動您的叢集和將 Amazon EC2 執行個體新增到 Amazon VPC 時，您會需要此資訊。

7. 建立 Amazon VPC 安全群組。您將對您的叢集和 Amazon EC2 執行個體使用。

   a. 在的左側導覽窗格中AWS Management Console，選擇安全群組。

   b. 選擇 Create Security Group (建立安全群組)。

   c. 在對應的方塊中，為安全群組輸入名稱和描述。適用於VPC」，選擇您的 VPC 的識別符。

   d. 當您滿意設定後，請選擇 Yes, Create (是，建立)。

8. 為您的安全群組定義網路傳入規則。此規則將允許您使用 Secure Shell (SSH) 連接至 Amazon EC2 執行個體。

   a. 在左導覽窗格中，選擇 Security Groups (安全群組)。

   b. 在清單中找到您的安全群組，然後選擇它。

   c. 在 Security Group (安全群組) 下，選擇 Inbound (入站) 標籤。在 Create a new rule (建立新規則) 方塊中，選擇 SSH，然後選擇 Add Rule (新增規則)。

      針對您的新傳入規則設定下列值，允許透過 HTTP 存取：

- 類型：HTTP
- 來源：0.0.0.0/0

d. 針對您的新傳入規則設定下列值，允許透過 HTTP 存取：

- 類型：HTTP
- 來源：0.0.0.0/0

選擇 Apply Rule Changes (套用規則變更)。

現在您已準備好可建立子網路群組和建立叢集在您的 VPC 中。

子網路和子網路群組

子網路群組是子網路的集合 (一般是私有)，您可以為在 Amazon Virtual Private Cloud (VPC) 環境中執行的叢集指定這些子網路。

在 Amazon VPC 中建立叢集時，您可以指定子網路群組或使用提供的預設子網路群組。MemoryDB 會使用此子網路群組選擇子網路及該子網路中的 IP 地址，以與您的節點建立關聯。

本節說明如何建立和利用子網路和子網路和子網路群組來管理 MemoryDB 資源的存取權。

如需 Amazon VPC 環境中子網路群組使用方式的詳細資訊，請參閱「步驟 2：授權對叢集的存取」。

支援的記憶體資料庫 AZ 識別碼

區域名稱/區域	支援的 AZ ID		
美國東部 (俄亥俄) 區域 us-east-2	use2-az1, use2-az2, use2-az3		
美國東部 (維吉尼亞北部) 區域 us-east-1	use1-az2, use1-az4, use1-az6		
美國西部 (加利佛尼亞北部) 區域 us-west-1	usw1-az1, usw1-az2, usw1-az3		
美國西部 (奧勒岡) 區域 us-west-2	usw2-az1, usw2-az2, usw2-az3		
加拿大 (中部) 區域 ca-central-1	cac1-az1, cac1-az2, cac1-az4		
亞太區域 (香港) 區域	ape1-az1, ape1-az2, ape1-az3		

區域名稱/區域	支援的 AZ ID		
ap-east-1			
亞太區域 (孟買) 區域 ap-south-1	aps1-az1, aps1-az2, aps1-az3		
亞太區域 (東京) 區域 ap-northeast-1	apne1-az1, apne1-az2, apne1-az4		
亞太區域 (首爾) 區域 ap-northeast-2	apne2-az1, apne2-az2, apne2-az3		
亞太區域 (新加坡) 區域 ap-southeast-1	apse1-az1, apse1-az2, apse1-az3		
亞太區域 (雪梨) 區域 ap-southeast-2	apse2-az1, apse2-az2, apse2-az3		
歐洲 (法蘭克福) 區域 eu-central-1	euc1-az1, euc1-az2, euc1-az3		
Europe (Ireland) Region eu-west-1	euw1-az1, euw1-az2, euw1-az3		
歐洲 (倫敦) 區域 eu-west-2	euw2-az1, euw2-az2, euw2-az3		

區域名稱/區域	支援的 AZ ID		
歐洲 (斯德哥爾摩) 區域 eu-north-1	eun1-az1, eun1-az2, eun1-az3		
南美洲 (聖保羅) 區域 sa-east-1	sae1-az1、sae1-az2、sae1-az3		
中國 (北京) 區域 cn-north-1	cnn1-az1、cnn1-az2		
中國 (寧夏) 區域 cn-northwest-1	cnw1-az1、cnw1-az3		

主題

- 建立子網路群組
- 更新子網路群組
- 檢視子網路群組明細
- 刪除子網路群組

建立子網路群組

建立新子網路群組時，請記下可用 IP 地址的數量。如果子網路有很少可用的 IP 地址，對於您還可以新增至叢集的節點數量，您可能受到限制。若要解決此問題，您可以對子網路群組指定一或多個子網路，使得您在叢集的可用區域中有足夠數量的 IP 地址。在那之後，您便可以將更多節點新增至您的叢集。

下列程序示範如何使用主控台、和 MemoryDB API 來建立名為mysubnetgroup (主控台) 的子網路群組。AWS CLI

建立子網路群組 (主控台)

下列程序顯示如何建立子網路群組 (主控台)。

建立子網路群組 (主控台)

1. 登入AWS管理主控台，並前往 https://console.aws.amazon.com/memorydb/ 開啟 MemoryDB 主控台。
2. 在左側導覽窗格中，選擇子網路群組。
3. 選擇 Create Subnet Group (建立子網路群組)。
4. 在建立子網路群組頁面，執行下列動作：
   a. 在 Name (名稱) 方塊中，輸入子網路群組的名稱。

      叢集命名限制條件如下：

      - 必須包含 1-40 個英數字元或連字號。
      - 必須以字母開頭。
      - 不能連續包含兩個連字號。
      - 結尾不能是連字號。

   b. 在 Description (描述) 方塊中，輸入子網路群組的描述。
   c. 在 VPC ID 方塊中，選擇您建立的 Amazon VPC。如果您尚未建立 VPC，請選擇 [建立 VPC] 按鈕，然後按照步驟建立 VPC。
   d. 在 [選取的子網路] 中，選擇 [可用區域] 和 [私人子網路的識別碼]，然後選擇 [選擇]。
5. 對於「標籤」，您可以選擇性地套用標籤來搜尋和篩選子網路，或追蹤您的AWS成本。
6. 當您滿意所有設定後，選擇建立。
7. 在出現的確認訊息中，選擇 Close (關閉)。

您新的子網路群組會顯示在 MemoryDB 主控台的子網路群組清單中。您可以在視窗底部選擇要查看詳細資訊的子網路群組，例如與此群組相關聯的所有子網路。

建立子網路群組 (AWSCLI)

在命令提示字元中，使用命令 `create-subnet-group` 來建立子網路群組。

若為 Linux、macOS 或 Unix：

```
aws memorydb create-subnet-group \
    --subnet-group-name mysubnetgroup \
    --description "Testing" \
    --subnet-ids subnet-53df9c3a
```

針對 Windows：

```
aws memorydb create-subnet-group ^
    --subnet-group-name mysubnetgroup ^
    --description "Testing" ^
    --subnet-ids subnet-53df9c3a
```

此命令應該產生類似下列的輸出：

```
    {
        "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"
        }
    }
```

如需詳細資訊，請參閱 AWS CLI 主題 create-subnet-group。

建立子網路群組 (MemoryDB API)

使用 MemoryDB API 呼叫並`CreateSubnetGroup`搭配下列參數：

- `SubnetGroupName=`*`mysubnetgroup`*
- `Description=`*`Testing`*
- `SubnetIds.member.1=`*`subnet-53df9c3a`*

更新子網路群組

您可以更新子網路群組的說明，或修改與子網路群組相關聯的子網路 ID 清單。如果叢集目前正使用子網路，則無法從子網路群組刪除該子網路 ID。

下列程序顯示如何更新子網路群組。

更新子網路群組 (主控台)

更新子網路群組

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇子網路群組。
3. 在子網路群組的清單中，選擇您要修改的子網路群組。
4. 「名稱」、「vPCID」和「說明」欄位無法修改。
5. 在「選取的子網路」段落中，按一下管理，對子網路所需的可用區域進行任何變更。選擇 Save (儲存) 以儲存變更。

更新子網路群組 (AWSCLI)

在命令提示中，使用命令`update-subnet-group`來更新子網路群組。

若為 Linux、macOS 或 Unix：

```
aws memorydb update-subnet-group \
    --subnet-group-name mysubnetgroup \
    --description "New description" \
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

針對 Windows：

```
aws memorydb update-subnet-group ^
    --subnet-group-name mysubnetgroup ^
    --description "New description" ^
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

此命令應該產生類似下列的輸出：

```
{
```

```
    "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",
    }
}
```

如需詳細資訊，請參閱 AWS CLI 主題 update-subnet-group。

更新子網路群組 (MemoryDB API)

使用 MemoryDB API 呼叫並UpdateSubnetGroup搭配下列參數：

- SubnetGroupName=*mysubnetgroup*
- 您想要變更其值的任何其他參數。此範例使用 Description=*New%20description* 來變更子網路群組的描述。

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**
>
> 建立新子網路群組時，請記下可用 IP 地址的數量。如果子網路有很少可用的 IP 地址，對於您還可以新增至叢集的節點數量，您可能受到限制。若要解決此問題，您可以對子網路群組指定一或多個子網路，使得您在叢集的可用區域中有足夠數量的 IP 地址。在那之後，您便可以將更多節點新增至您的叢集。

檢視子網路群組明細

下列程序顯示如何檢視子網路群組的詳細資料。

檢視子網路群組 (主控台) 的詳細資訊

檢視子網路群組的詳細資訊 (主控台)

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇子網路群組。
3. 在 [子網路群組] 頁面上，選擇 [名稱] 底下的子網路群組，或在搜尋列中輸入子網路群組的名稱。
4. 在 [子網路群組] 頁面上，選擇 [名稱] 底下的子網路群組，或在搜尋列中輸入子網路群組的名稱。
5. 在子網路群組設定下，您可以檢視子網路群組的名稱、說明、VPC ID 和 Amazon 資源名稱 (ARN)。
6. 在子網路下，您可以檢視子網路群組的可用區域、子網路 ID 和 CIDR 區塊
7. 在「標記」下，您可以檢視與子網路群組相關聯的任何標記。

檢視子網路群組詳細資訊 (AWSCLI)

在命令提示字元中，使用命令`describe-subnet-groups`來檢視指定子網路群組的詳細資料。

若為 Linux、macOS 或 Unix：

```
aws memorydb describe-subnet-groups \
    --subnet-group-name mysubnetgroup
```

針對 Windows：

```
aws memorydb describe-subnet-groups ^
    --subnet-group-name mysubnetgroup
```

此命令應該產生類似下列的輸出：

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

若要檢視所有子網路群組的詳細資料，請使用相同的指令，但不指定子網路群組名稱。

```
aws memorydb describe-subnet-groups
```

如需詳細資訊，請參閱 AWS CLI 主題 describe-subnet-groups。

檢視子網路群組 (MemoryDB API)

使用 MemoryDB API 呼叫並DescribeSubnetGroups搭配下列參數：

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
```

刪除子網路群組

如果您決定不再需要使用您的子網路群組，您可以將它刪除。某個叢集正在使用子網路群組，則無法將它刪除。如果啟用多個可用區的叢集使該叢集的子網路少於兩個，您也無法刪除該叢集上的子網路群組。您必須先取消勾選異地同步備份，然後刪除子網路。

下列程序顯示如何刪除子網路群組。

刪除子網路群組 (主控台)

刪除子網路群組

1. 登入AWS Management Console並開啟 Redis 的記憶體資料庫主控台，網址為 https://console.aws.amazon.com/memorydb/。
2. 在左側導覽窗格中，選擇子網路群組。
3. 在子網路群組清單中，選擇您要刪除的子網路群組，選擇 [動作]，然後選擇 [刪除]。

   > **Note**
   >
   > 您無法刪除預設子網路群組或與任何叢集相關聯的子網路群組。

4. 刪除子網路群組確認畫面隨即出現。
5. 若要刪除子網路群組，請`delete`在確認文字方塊中輸入。若要保留子網路群組，請選擇 Cancel (取消)。

刪除子網路群組 (AWSCLI)

使用 AWS CLI，搭配下列參數呼叫命令 delete-subnet-group：

- `--subnet-group-name` *`mysubnetgroup`*

若為 Linux、macOS 或 Unix：

```
aws memorydb delete-subnet-group \
    --subnet-group-name mysubnetgroup
```

針對 Windows：

```
aws memorydb delete-subnet-group ^
```

```
    --subnet-group-name mysubnetgroup
```

如需詳細資訊，請參閱 AWS CLI 主題 delete-subnet-group。

刪除子網路群組 (記憶體 DbAPI)

使用 MemoryDB API 呼叫並DeleteSubnetGroup搭配下列參數：

- 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
```

此命令不會產生輸出。

如需詳細資訊，請參閱記憶體資料庫 API 主題DeleteSubnetGroup。

Redis API 和界面 VPC 端點 (AWS PrivateLink)

您可以在 VPC 和亞馬遜 Redis API 終端節點的內存數據庫通過創建界面 VPC 端點。界面端點提供的 AWS PrivateLink。AWS PrivateLink 允許您可以私有存取 Redis API 操作的 MemoryDB，無需透過網際網際網路閘道、NAT 裝置、VPN 連接或AWSDirect Connect 連線。

您的 VPC 中的實例不需要公有 IP 地址，即能與 Redis API 端點的 MemoryDB 通訊。您的實例也不需要公有 IP 地址，即能使用任何可用的 MemoryDB API 操作。您的 VPC 與 Redis MemoryDB 之間的流量，都會在 Amazon 網路的範圍內。每個界面端點都是由您子網路中的一或多個彈性網路界面表示。如需彈性網路界面的詳細資訊，請參閱 Amazon EC2 使用者指南中的彈性網路界面。

- 如需 VPC 端點的詳細資訊，請參界面 VPC 端點 (AWS PrivateLink)中的Amazon VPC User Guide。
- 如需 MemoryDB API 操作的詳細資訊，請參MemoryDB API 操作。

在建立界面 VPC 端點之後，如果您啟用私有 DNS終端節點的主機名，默認的內存數據庫終端節點（ https://memorydb.*Region (##)*.amazonaws.com) 可以解析成您的 VPC 端點。如果您尚未啟用私有 DNS 主機名稱，Amazon VPC 會透過以下格式提供一個 DNS 端點名稱，供您使用：

```
VPC_Endpoint_ID.memorydb.Region.vpce.amazonaws.com
```

如需詳細資訊，請參閱「」界面 VPC 端點 (AWS PrivateLink)中的Amazon VPC User Guide。MemoryDB 支援呼叫其所有API 動作在您的 VPC 中。

> **Note**
>
> 只能為 VPC 中的一個 VPC 終端節點啟用私有 DNS 主機名。如果您想創建額外的 VPC 終端節點，則應該為其禁用私有 DNS 主機名。

VPC 端點的考量事項

在設定 Redis API 端點的 MemoryDB 的界面 VPC 端點前，請務必檢界面端點屬性和限制中的Amazon VPC User Guide。所有 MemoryDB API 操作那些是與管理 Redis 資源的 MemoryDB 相關，都從使用AWS PrivateLink。內存 DB API 端點支援 VPC 端點政策。根據預設，允許透過端點完整存取 MemoryDB API 操作。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 端點控制對服務的存取。

為 建立介面 VPC 端點該MemoryDB API

您可使用 Amazon VPC 主控台或AWS CLI。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的建立介面端點。

在建立界面 VPC 端點之後，您可以啟用此端點的私有 DNS 主機名稱。執行此操作時，Redis 終端節點的默認內存數據庫（ https://memorydb.*Region (##)*.amazonaws.com) 可以解析成您的 VPC 端點。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 透過介面端點存取服務。

為 建立 VPC 端點政策該Amazon MemoryDB API

您可以將端點政策連接至控制 MemoryDB API 存取權限的 VPC 端點。此政策會指定以下項目：

- 可執行動作的主體。
- 可執行的動作。
- 可供執行動作的資源。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 端點控制對服務的存取。

Example 內存 DB API 動作的 VPC 端點政策

以下是 MemoryDB API 端點政策的範例。連接至端點後，此政策會針對所有資源上的所有委託人，授予列出的 MemoryDB API 動作的存取權限。

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
  "Action": [
   "memorydb:CreateCluster",
   "memorydb:UpdateCluster",
   "memorydb:CreateSnapshot"
  ],
  "Resource": "*"
 }]
}
```

Example 拒絕所有來自指定AWS帳戶

下列 VPC 端點政策拒絕 AWS 帳戶 *123456789012* 對使用該端點之資源的任何存取。此政策允許來自其他帳戶的所有動作。

```
{
 "Statement": [{
   "Action": "*",
   "Effect": "Allow",
   "Resource": "*",
   "Principal": "*"
  },
  {
   "Action": "*",
   "Effect": "Deny",
   "Resource": "*",
   "Principal": {
```

```
    "AWS": [
     "123456789012"
    ]
   }
  }
 ]
}
```

Redis 內存數據庫中的服務更新

Redis MemyDB (Redis) 會自動監控叢集和節點的機羣，以在他們可用的時候套用服務更新。通常，您可以設定預先定義的維護窗口，以便 MemoryDB 可以套用這些更新。不過，在某些情況下，您可能會發現此方法過於嚴格，而且可能會限制您的業務流程。

使用服務更新，您就可以控制更新的套用時間和套用的時間。您也可以即時監控這些更新套用到您所選 MemoryDB 叢集的進度。

管理服務更新

內存 DB 服務更新為定期發佈。如果您有一或多個符合那些服務更新資格的叢集，您會透過電子郵件、SNS、Palth Health Dashboard (PHD) 和 Amazon CloudWatch 事件收到通知。這些更新也會顯示在服務更新頁面。透過使用此儀表板，您就可以檢視 MemoryDB 機羣的所有服務更新及其狀態。

您可以控制在開始自動更新之前套用更新的時間。我們強烈建議您套用類型為安全性更新來確保透過最新的安全性修正讓 MemyDB 隨時保持在最新的狀態。

以下區段更會詳細探討這些選項。

主題

- 應用服務更新

應用服務更新

您可以在更新包含可用狀態。服務更新是累積性的。換句話說，您尚未套用的任何更新都會包含在最新的更新中。

如果服務更新啟用了自動更新，則可以選擇在服務可用時不執行任何操作。MemeryDB 將計劃在羣集的維護窗口後應用更新自動更新開始日期。您將收到更新的每個階段的相關通知。

> **Note**
>
> 您只能應用具有可用或者計劃狀態。

如需檢視和將任何服務特定更新套用到適用 MemoryDB 叢集的詳細資訊，請參閱使用主控台套用服務更新。

當新服務更新適用於一或多個 MemoryDB 叢集時，您就可以使用 MemonyDB 控台、API 或AWS CLI 來套用更新。以下區段會說明您可以用來套用更新的選項。

使用主控台套用服務更新

若要檢視可用服務更新的清單以及其他資訊，請轉到服務更新頁面。

1. 登入AWS Management Console，然後打開 Redis 控制台的內存數據庫https://console.aws.amazon.com/memorydb/。
2. 在導覽窗格中，選擇服務更新。

下服務更新詳細資訊您可以檢視下列項目：

- 服務更新名稱：服務更新的唯一名稱
- 更新說明：服務更新的詳細資訊
- 自動更新開始日期：如果設置了此屬性，MemeryDB 將開始計劃您的集羣，以便在此日期之後在相應的維護窗口中自動更新。您將在確切的計劃維護窗口中提前收到通知，這可能不是在自動更新開始日期。您仍然可以隨時將更新應用到您的集羣。如果未設置屬性，則服務更新不會啟用自動更新，MemeryDB 不會自動更新您的集羣。

在 中叢集更新狀態部分中，您可以查看尚未應用或最近剛應用服務更新的羣集列表。您可以檢視每個叢集：

- 叢集名稱：叢集的名稱
- 節點已更新：特定叢集中的單個節點比率，這些節點已更新或仍可用於特定服務更新。
- 更新類型：服務更新的類型，可以是安全性更新或者引擎更新
- 狀態：服務更新的狀態，可以是下列狀態之一：
  - 可用：更新可用於必要的叢集。
  - 進步：正在將更新套用到此叢集。

- 計劃：已排程更新日期。
- 完成：已成功套用更新。具有完整狀態的集羣將在完成後顯示 7 天。

如果您選擇了任何或所有具有可用或者計劃狀態，然後選擇立即申請，則更新將開始應用於這些羣集。

使用 AWS CLI 套用服務更新

在收到服務更新可供使用的通知後，您就可以使用 AWS CLI 來檢查和套用這些更新：

- 若要擷取可用服務更新的說明，請執行下列命令：

  ```
  aws memorydb describe-service-updates --status available
  ```

  如需詳細資訊，請參閱「」描述服務更新。
- 若要在叢集清單上套用服務更新，請執行下列命令：

  ```
  aws memorydb batch-update-cluster --service-update
  ServiceUpdateNameToApply=sample-service-update --cluster-names cluster-1
  cluster2
  ```

  如需詳細資訊，請參閱「」批處理更新羣集。

參考

本節中的主題涵蓋使用內存的內存，以及AWS CLI。本節也包含常見錯誤訊息及服務通知。

- 使用內存 DB API
- 內存數據庫 API 參考
- 的內存 DB 區段AWS CLI參考

使用內存 DB API

本節提供如何使用和實作 MemoryDB 操作的任務導向的說明。如需這些操作的完整說明，請參閱 。內存 DB API 參考。

主題

- 使用查詢 API
- 可用程式庫
- 對應用程式進行疑難排解

使用查詢 API

查詢參數

HTTP 查詢型請求是使用 HTTP 動詞 GET 或 POST，以及名為 `Action` 之查詢參數的 HTTP 請求。

每一個查詢請求都須包括一些常用參數，來處理動作的身份驗證和選取。

部分操作有數個參數清單。這些清單是使用 `param.`*n* 表示法來指定的。*n* 的值是從 1 開始的整數。

查詢請求身份驗證

您只能透過 HTTPS 傳送查詢請求，而且必須在每一個查詢請求中包括一個簽章。本節說明如何建立簽章。下列程序所述的方法也稱為「簽章第 4 版」。

下列為用來對 AWS 進行驗證要求的基本步驟。此處假設您已向 AWS 註冊，並具有存取金鑰 ID 與私密存取金鑰。

查詢身分驗證程序

1. 寄件者可編寫傳給 AWS 的請求。
2. 寄件者會計算請求簽章，其為使用 SHA-1 雜湊函數的雜湊型訊息身份驗證程式碼 (HMAC) 的金鑰式雜湊，如本主題下一節所述。
3. 請求寄件者會將請求資料、簽章和存取金鑰 ID (所用之私密存取金鑰的金鑰識別碼) 傳送給 AWS。
4. AWS 會使用存取金鑰 ID 來查詢私密存取金鑰。
5. AWS 使用的演算法與計算請求中簽章的演算法相同，可從請求資料及私密存取金鑰產生簽章。

6. 如果簽章相符，則會將請求視為真實請求。若比較失敗，則會捨棄該要求，且 AWS 會傳回錯誤回應。

> **Note**
>
> 如果請求包含 `Timestamp` 參數，針對請求計算出的簽章就會在其值的 15 分鐘後過期。
> 如果請求包含 `Expires` 參數，簽章就會於 `Expires` 參數指定的時間過期。

計算請求簽章

1. 建立標準化查詢字串，以在本程序稍後使用：

   a. 依據參數名稱與自然位元組順序，排序 UTF-8 查詢字串元件。參數可以來自 GET URI 或 POST 內文 (當 Content-Type 為 application/x-www-form-urlencoded 時)。

   b. 根據下列規則，為每個參數名稱和值執行 URL 編碼：

      i. 請不要針對 RFC 3986 所定義的任何未預留字元執行 URL 編碼。這些未預留字元包括 A-Z、a-z、0-9、連字號 (-)、底線 (_)、句點 (.) 及波狀符號 (~)。

      ii. 對所有其他含有 %XY 的字元執行百分比編碼，其中 X 和 Y 是十六進位字元 0-9 和大寫 A-F。

      iii. 對 %XY%ZA... 格式的延伸 UTF-8 字元執行百分比編碼。

      iv. 將空白字元百分比編碼為 %20 (而非常見編碼機制使用的 +)。

   c. 使用等號 (=) (ASCII 字元 61) 分隔編碼過的參數名稱與其編碼值，即使該參數值為空白也是如此。

   d. 使用 & 符號 (ASCII 字碼 38) 分隔名稱值對。

2. 依據下列虛擬語法 ("\n" 代表 ASCII 換行符號)，建立要簽署的字串。

```
StringToSign = HTTPVerb + "\n" +
ValueOfHostHeaderInLowercase + "\n" +
HTTPRequestURI + "\n" +
CanonicalizedQueryString <from the preceding step>
```

   HTTPRequestURI 元件是 URI 的 HTTP 絕對路徑元件，一直到查詢字串為止 (但不包括查詢字串)。如果 HTTPRequestURI 空白，請使用斜線 (/)。

3. 使用您剛建立的字串、私密存取金鑰做為金鑰，以及 SHA256 或 SHA1 做為雜湊演算法，計算 RFC 2104 相容的 HMAC。

   如需詳細資訊，請參閱 https://www.ietf.org/rfc/rfc2104.txt。

4. 將結果值轉換成 base64。
5. 將該值包含為請求中的 `Signature` 參數值。

例如，以下是範例請求 (為了清楚起見，已加入分行符號)。

```
https://memory-db.us-east-1.amazonaws.com/
    ?Action=DescribeClusters
    &ClusterName=myCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2021-01-01
```

針對上述的查詢字串，您可透過下列字串來計算 HMAC 簽章。

```
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:
```

結果為下列簽署的請求。

```
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
```

如需簽章程序和計算請求簽章的詳細資訊，請參Signature 第 4 版簽署程序及其分專題.

可用程式庫

針對偏好使用特定語言 API (而非查詢 API) 建置應用程式的開發人員，AWS 提供軟體開發套件 (SDK)。這些軟體開發套件提供 API 中不包含的基本功能，例如請求身份驗證、請求重試與錯誤處理，以便輕鬆上手。我們提供以下程式設計語言的軟體開發套件和其他資源：

- Java
- Windows 與 .NET
- PHP
- Python
- Ruby

如需其他語言的相關資訊，請參示例代碼和程式庫。

對應用程式進行疑難排解

MemoryDB 提供特定和描述性錯誤，以協助您在與 MemoryDB API 互動時進行故障診斷。

擷取錯誤

通常，您想要應用程式在您花費任何時間處理結果之前，先檢查請求是否已產生錯誤。若要了解系統是否發生錯誤，最簡單的方式即為尋找`Error`節點。

XPath 語法提供簡單的方式，讓您可搜尋 `Error` 節點是否存在，並輕鬆擷取錯誤碼和訊息。下列程式碼片段使用 Perl 和 XML::XPath 模組，來判斷請求期間是否發生錯誤。如果發生錯誤，程式碼會列印回應中的第一個錯誤碼和訊息。

```
use XML::XPath;
my $xp = XML::XPath->new(xml =>$response);
if ( $xp->find("//Error") )
{print "There was an error processing your request:\n", " Error code: ",
$xp->findvalue("//Error[1]/Code"), "\n", " ",
$xp->findvalue("//Error[1]/Message"), "\n\n"; }
```

對秘訣進行故障診斷

我們建議以下列程序來診斷並解決 MemoryDB API 發生的問題。

- 確認內存 DB 已正常運作。

  若要執行此操作，您只要開啟瀏覽器視窗，並提交查詢請求給 MemoryDB 服務 (例如 https://memory-db.us-east-1.amazonaws.com)。若出現 MissingAuthenticationTokenException 或未知情況異常，即確認服務可供使用，並可回應請求。
- 檢查請求的結構。

  每項 InthenyDB 操作都會在內存 DB API 參考。再次檢查您是否正確使用參數。若要提供有關可能出錯的概念，請查看範例請求或使用者案例，來查看那些範例是否執行類似操作。
- 查看論壇。

  MemoryDB 具有開發論壇，您可以在其中搜尋其他人在過程中所遇到問題的解決方案。若要檢視論壇，請參閱

  https://forums.aws.amazon.com/。

適用於 Redis 的 Amazon 內存數據庫的配額

您的 AWS 帳戶有每項 AWS 服務的預設配額 (先前稱為限制)。除非另有說明，否則每個配額都是區域特定規定。您可以請求提高某些配額，而其他配額無法提高。

若要請求增加配額，請參閱 Service Quotas 使用者指南中的請求提高配額。如果 Service Quotas 中尚未提供配額，請使用增加服務配額表單。

您的 AWS 帳戶具有以下與 MemoryDB 相關的配額。

資源	預設
每個區域的節點	300
每個執行個體類型的叢集節點	90
每個碎片的節點數	6
每區域參數群組數	150
每區域子網路群組數	150
每個子網路群組的子網路	20
每使用者群組使用者數	100
使用者總數	1000
使用者群組數	100

MemoryDB 使用者指南的文件歷史記錄

下表說明 MemoryDB 的說明文件版本。

變更	描述	日期
MemoryDB 現在支援使用 IAM 驗證使用者	IAM 身分驗證可讓您使用身分，來驗證與 MemoryDB Functions 的連線。AWS Identity and Access Management這可讓您強化安全模型，並簡化許多管理安全任務。如需詳細資訊，請參閱以 IAM 進行身分驗證。	2023 年 5 月 10 日
MemoryDB 現在支援 Redis 7 版	此版本為 MemoryDB to Redis 增加一些新功能：Redis Functions、ACL 改善與增強 I/O 多工處理。如需詳細資訊，請參閱 Redis 引擎版本。	2023 年 5 月 9 日
內存數據庫現在提供保留節點	相較於隨需節點定價，預留節點可提供您更多的 discount。預留節點並非實體，而是一種套用到您帳戶中隨需節點用量的計費 discount。如需詳細資訊，請參閱 MemoryDB 保留節點。	2022 年 12 月 27 日
內存數據庫現在支持數據分層	MemoryDB 用於 Redis 資料分層。您可以使用資料分層作為較低成本的方式，將叢集擴展至最多數百 TB 的容量。如需詳細資訊，請參閱資料分層。	2022 年 11 月 3 日

MemoryDB 現在支援原生JavaScript物件標記法 (JSON) 格式	原生JavaScript物件標記法 (JSON) 格式是一種簡單、無結構描述的方式，對 Redis 叢集內的複雜資料集進行編碼。您可以在 Redis 叢集內，使用JavaScript物件標記法 (JSON) 格式，原生儲存和存取資料，並更新儲存在這些叢集中的 JSON 資料，不需管理自訂程式碼來序列化和還原序列化。如需詳細資訊，請參閱 JSON 入門。	2022 年 5 月 25 日
內存數據庫現在支持 AWS PrivateLink	AWSPrivateLink可讓您以私有方式存取 MemoryDB API 操作，無需網際網路閘道、NAT 連線或 AWS Direct Connect 連線。如需詳細資訊，請參閱 MemoryDB API 和界面 VPC 端點 ()。AWS PrivateLink	2022 年 1 月 24 日
初始版本	MemoryDB 使用者指南的初始版本。如需詳細資訊，請參閱什麼是 MemoryDB？	2021 年 8 月 19 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。