使用阿帕奇冰山 AWS

AWS 规定指引

AWS 規定指引: 使用阿帕奇冰山 AWS

Table of Contents

在 AWS 上使用阿帕奇冰山

Amazon Web Services（貢獻者）

2024 年 4 月 (文件歷史記錄)

Apache Iceberg 是一種開放原始碼的表格格式，可簡化資料表管理，同時改善效 AWS Amazon EMR， AWS Glue Amazon 雅典娜和亞馬 Amazon Redshift 等分析服務包括對 Apache Iceberg 的原生支持，因此您可以在亞馬遜簡單存儲服務（亞馬遜 S3）之上輕鬆地構建交易數據湖。 AWS

本技術指南提供有關不同 Apache Iceberg 入門的指引 AWS 服務，並包含 AWS 在大規模執行 Apache Iceberg 的同時最佳化成本和效能的最佳實務和建議。

本指南適用於任何正在使用 Apache Iceberg 的人 AWS，從想要快速開始使用 Apache Iceberg 的新手使用者到想要最佳化和調整其現有 Apache Iceberg 工作負載的進階使用者。 AWS

在本指南中：

現代資料湖

現代資料湖中的進階使用案例

在成本、可擴充性和彈性方面，資料湖是儲存資料的最佳選擇之一。您可以使用資料湖以低成本保留大量結構化和非結構化資料，並將此資料用於不同類型的分析工作負載，從商業智慧報告到大數據處理、即時分析、機器學習和生成人工智慧 (AI)，以協助指導更好的決策。

儘管有這些好處，資料湖最初並不是使用類似資料庫的功能來設計。資料湖不提供原子性、一致性、隔離性和耐久性 (ACID) 處理語意的支援，您可能需要使用多種不同的技術，有效地最佳化和管理數百或數千名使用者的資料。資料湖不提供下列功能的原生支援：

- 隨著業務的數據變化，執行有效的記錄級更新和刪除
- 隨著資料表成長到數百萬個檔案和數十萬個分割區，管理查詢效能
- 確保多個並行編寫器和讀者的資料一致性
- 在寫入作業失敗時防止資料損毀
- 隨著時間的推移而不會（部分）重寫數據集的情況下演變

在處理變更資料擷取 (CDC) 或與隱私權、刪除資料和串流資料擷取有關的使用案例中，這些挑戰已變得特別普遍，這可能會導致資料表次最佳化。

使用傳統 Hive 格式表格的資料湖僅支援整個檔案的寫入作業。這使得更新和刪除難以實施，耗時且成本高昂。此外，需要在符合 ACID 標準的系統中提供並行控制和保證，以確保數據的完整性和一致性。

為了協助克服這些挑戰，Apache Iceberg 提供了其他類似資料庫的功能，可簡化資料湖的最佳化和管理開銷，同時仍支援 Amazon 簡易儲存服務 (Amazon S3) 等具成本效益的系統上的儲存。

阿帕奇冰山簡介

Apache Iceberg 是一種開放原始碼資料表格式，提供資料湖資料表中的功能，歷史上只能在資料庫或資料倉儲中使用。它專為擴充性和效能而設計，非常適合用於管理超過數百 GB 的資料表。冰山表的一些主要特點是：

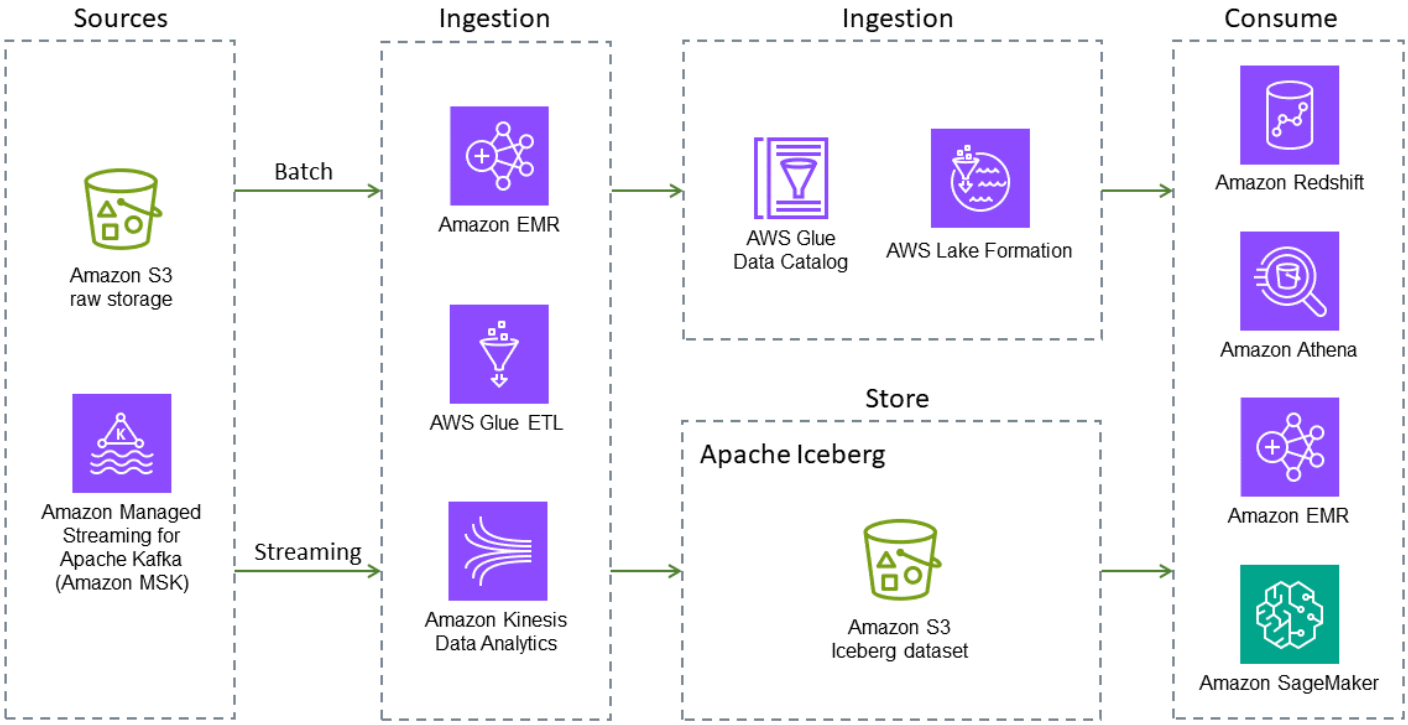
- 刪除、更新和合併。Iceberg 支援用於資料倉儲的標準 SQL 命令，可與資料湖表一起使用。
- 快速掃描規劃和進階篩選。Iceberg 儲存引擎可使用的中繼資料，例如分割區和資料行層級統計資料，以加快規劃和執行查詢的速度。

- 完整的架構演進。Iceberg 支持添加，刪除，更新或重命名列，而不會產生副作用。
- 分區演化。您可以在資料磁碟區或查詢模式變更時更新資料表的分割區配置。Iceberg 支援變更資料表分割的資料行，或在複合資料分割中新增資料欄，或從複合資料分割中移除資料行。
- 隱藏的分區。此功能可防止自動讀取不必要的分區。如此一來，使用者就不需要瞭解資料表的分割詳細資料，或在查詢中新增額外的篩選器。
- 版本回滾。用戶可以通過恢復到事務前狀態快速糾正問題。
- 時間旅行。使用者可以查詢表格的特定先前版本。
- 可序列化隔離。表更改是原子的，因此讀者永遠不會看到部分或未提交的更改。
- 並發編寫器。Iceberg 使用樂觀的並發性來允許多個交易取得成功。在發生衝突的情況下，其中一個作者必須重試該事務。
- 打開文件格式。冰山支持多種開源文件格式，包括阿帕奇鑲木地板，阿帕奇阿夫羅和阿帕奇 ORC。

總而言之，使用 Iceberg 格式的資料湖受益於交易一致性、速度、規模和結構描述演進。如需有關這些和其他冰山功能的詳細資訊，請參閱 Apache 冰山文件。

AWS 阿帕奇冰山的支持

Apache 冰山受到流行的開源數據處理框架的支持 AWS 服務 ，並由 Amazon EMR，Amazon Athena，Amazon Redshift 和. AWS Glue 下圖說明以 Iceberg 為基礎的資料湖的簡化參考架構。



以下 AWS 服務 提供了本地冰山集成。還有其他 AWS 服務 可以間接或通過打包冰山庫與冰山進行交互。

- Amazon S3 因其耐用性、可用性、可擴展性、安全性、合規性和稽核功能而成為建立資料湖的最佳場所。Iceberg 的設計和建置是為了與 Amazon S3 無縫互動，並為冰山文件中列出的許多 Amazon S3 功能提供支援。
- Amazon EMR 是針對 PB 規模資料處理、互動式分析和機器學習的巨量資料解決方案，使用開放原始碼架構 (例如 Apache Spark、Flink、Trito 和 Hive)。Amazon EMR 可以在自訂的 Amazon 彈性運算雲端 (Amazon EC2) 叢集、Amazon Elastic Kubernetes Service (Amazon EKS) 或亞馬遜 EMR 無伺服器上執行。 AWS Outposts
- Amazon Athena 是以開放原始碼架構為基礎建置的無伺服器互動式分析服務。它支援開放式表格和檔案格式，並提供簡化、靈活的方式來分析數 PB 的資料所在位置。Athena 針對冰山的讀取、時間旅行、寫入和 DDL 查詢提供原生支援，並將其用 AWS Glue Data Catalog 於冰山中繼存放區。
- Amazon Redshift 是 PB 規模的雲端資料倉儲，同時支援叢集型和無伺服器部署選項。Amazon Redshift Spectrum 可以查詢在 Amazon S3 上註冊 AWS Glue Data Catalog 並存放的外部表。Redshift 頻譜還提供了對冰山存儲格式的支持。
- AWS Glue是一種無伺服器資料整合服務，可讓您更輕鬆地探索、準備、移動和整合來自多個來源的資料，以進行分析、機器學習 (ML) 和應用程式開發。 AWS Glue 3.0 及更高版本支持數據湖的冰山框架。您可 AWS Glue 以使用在 Amazon S3 中的冰山表上執行讀取和寫入操作，或使用. AWS Glue Data Catalog還支持其他操作，例如插入，更新，Spark 查詢和 Spark 寫入。
- AWS Glue Data Catalog提供支援 Iceberg 資料表的 Hive 中繼資料庫相容資料目錄服務。
- AWS Glue 編目程式提供自動化功能，以在. AWS Glue Data Catalog
- Amazon SageMaker 支持使用 Iceberg 格式在 Amazon SageMaker 功能商店中存儲功能集。
- AWS Lake Formation為存取資料提供粗略且精細的存取控制許可，包括 Athena 或 Amazon Redshift 使用的冰山表。要了解有關冰山表的權限支持的更多信息，請參閱 Lake Formation 文檔。

AWS 提供支持 Iceberg 的廣泛服務，但涵蓋所有這些服務都超出了本指南的範圍。以下各節涵蓋了 Amazon EMR 上的星火（批量和 AWS Glue結構化流）以及 Amazon Athena SQL。以下部分提供了在 Athena SQL 中的冰山支持的快速瀏覽。

在 Amazon Athena SQL 中開始使用阿帕奇冰山表

Amazon Athena 為 Apache 冰山提供內置的支持。除了設定 Athena 說明文件的入門一節中詳述的服務必要條件外，您無需任何其他步驟或組態即可使用 Iceberg。本節提供在 Athena 中建立資料表的簡要介紹。如需詳細資訊，請參閱本指南稍後的使用 Athena SQL 使用 Apache 冰山資料表。

您可以 AWS 通過使用不同的引擎創建冰山表。這些表跨無縫工作 AWS 服務。若要使用 Athena SQL 建立您的第一個冰山資料表，您可以使用下列樣板程式碼。

```
CREATE TABLE <table_name> (
    col_1 string,
    col_2 string,
    col_3 bigint,
    col_ts timestamp)
PARTITIONED BY (col_1, <<<partition_transform>>>(col_ts))
LOCATION 's3://<bucket>/<folder>/<table_name>/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

以下各節提供在 Athena 中建立分割和未分割的 Iceberg 資料表的範例。如需詳細資訊，請參閱 A thena 文件中詳細說明的冰山語法。

建立未分割資料表

下列範例陳述式會自訂樣板 SQL 程式碼，以便在 Athena 建立未分割的 Iceberg 資料表。您可以將此陳述式新增至 A thenaconsole 中的查詢編輯器，以建立資料表。

```
CREATE TABLE athena_iceberg_table (
    color string,
    date string,
    name string,
    price bigint,
    product string,
    ts timestamp)
LOCATION 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/athena_iceberg_table/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

如需使用查詢編輯器的 step-by-step 指示，請參閱 Athena 文件中的入門指示。

建立分割的資料表

下列陳述式會使用 Iceberg 隱藏分割的概念，根據日期建立資料分割資料表。它會使用day()轉換，以時間戳記資料行的dd-mm-yyyy格式衍生每日分割區。Iceberg 不會將此值存儲為數據集中的新列。相反地，當您撰寫或查詢資料時，會即時衍生值。

```
CREATE TABLE athena_iceberg_table_partitioned (
    color string,
    date string,
    name string,
    price bigint,
    product string,
    ts timestamp)
PARTITIONED BY (day(ts))
LOCATION 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/athena_iceberg_table/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

使用單一 CTAS 陳述式建立資料表並載入資料

在前幾節的已分割和未分割範例中，Iceberg 資料表會建立為空白資料表。您可以使用INSERT 或陳述式將資料載入資料MERGE表。或者，您可以使用CREATE TABLE AS SELECT (CTAS)語句在一個步驟中創建數據並將其加載到 Iceberg 表中。

CTAS 是 Athena 在單一陳述式中建立資料表並載入資料的最佳方式。下列範例說明如何使用 CTAS，從 Athena 現有的隱藏/鑲木地板資料表 (iceberg_ctas_table) 建立冰山資料表 (hive_table)。

```
CREATE TABLE iceberg_ctas_table WITH (
  table_type = 'ICEBERG',
  is_external = false,
  location = 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/iceberg_ctas_table/'
) AS
SELECT * FROM "iceberg_db"."hive_table" limit 20
---
SELECT * FROM "iceberg_db"."iceberg_ctas_table" limit 20
```

若要進一步了解 CTAS，請參閱 Athena CTAS 文件。

插入、更新和刪除資料

Athena 支援使用INSERT INTO、UPDATE、和 DELETE FRO M 陳述式，將資料寫入冰山資料表的不同方式。MERGE INTO

注意：UPDATE，MERGE INTO，並DELETE FROM使用位置刪除的 merge-on-read 方法。Athena SQL 目前不支援此 copy-on-write 方法。

例如，下列陳述式會使用將資料新增INSERT INTO至 Iceberg 資料表：

```
INSERT INTO "iceberg_db"."ice_table" VALUES (
    'red', '222022-07-19T03:47:29', 'PersonNew', 178, 'Tuna', now()
)

SELECT * FROM "iceberg_db"."ice_table"
where color = 'red' limit 10;
```

輸出範例：

如需詳細資訊，請參閱 A thena 文件。

查詢冰山表

您可以使用 Athena SQL 對您的冰山資料表執行一般 SQL 查詢，如前面的範例所示。

除了常見的查詢之外，Athena 也支援冰山表格的時間旅行查詢。如前所述，您可以透過 Iceberg 資料表中的更新或刪除來變更現有記錄，因此您可以方便地使用時間移動查詢根據時間戳記或快照 ID 回溯舊版資料表。

例如，下列陳述式會更新的顏色值Person5，然後顯示 2023 年 1 月 4 日起較早的值：

```
UPDATE ice_table SET color='new_color' WHERE name='Person5'

SELECT * FROM "iceberg_db"."ice_table" FOR TIMESTAMP AS OF TIMESTAMP '2023-01-04
 12:00:00 UTC'
```

輸出範例：

Results (15)

Copy Download results

Search rows

1

#	color	date	name	price	product	ts
1	cyan	222022-07-19T03:47:29	Person5	353	Keyboard	2023-01-03 10:15:52.268000 UTC
2	lime	222022-07-19T03:47:29	Person1	833	Towels	2023-01-03 10:15:52.268000 UTC
3	turquoise	222022-07-19T03:47:29	Person1	1319	Shirt	2023-01-03 10:15:52.268000 UTC
4	blue	222022-07-19T03:47:29	Person3	163	Sausages	2023-01-03 10:15:52.268000 UTC

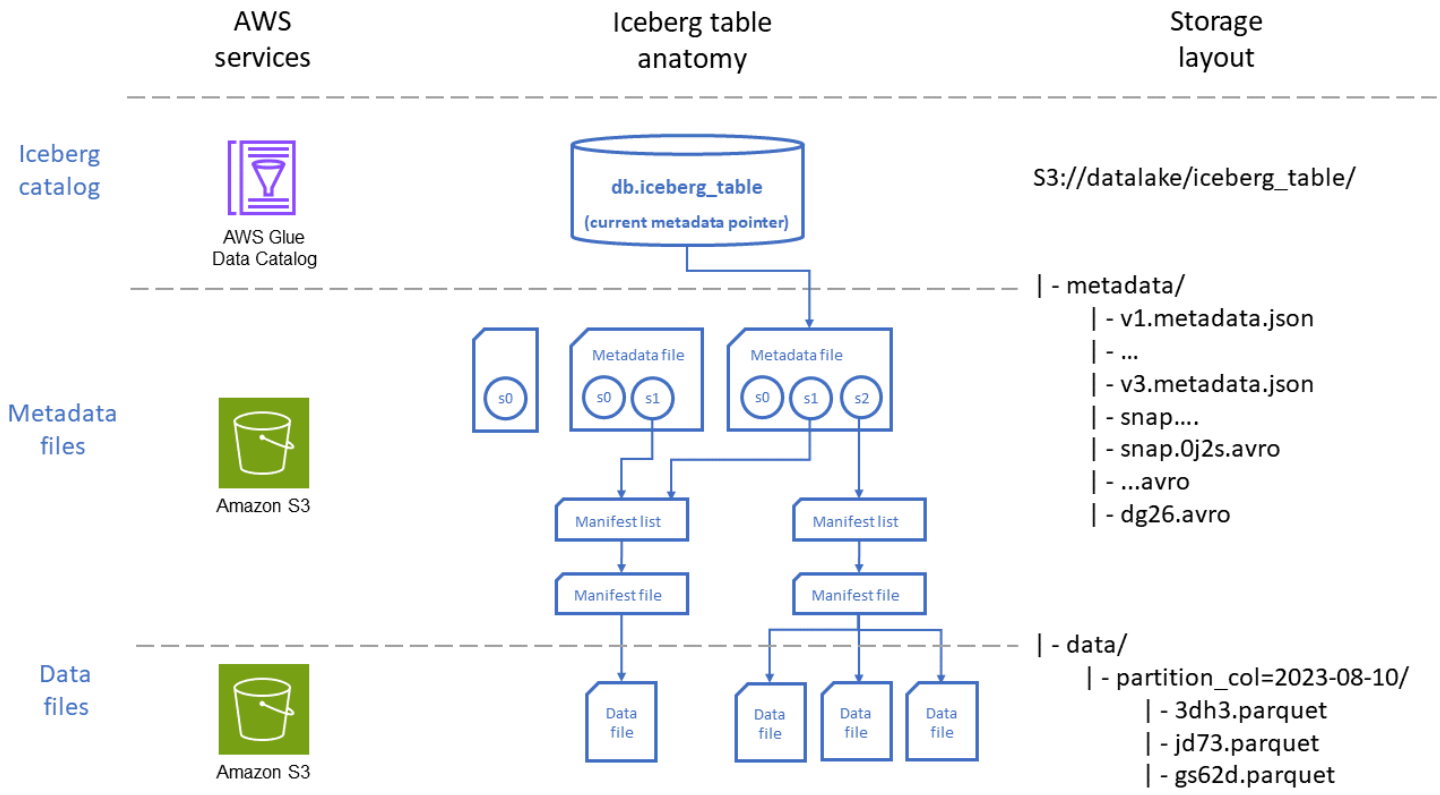
如需時間旅行查詢的語法和其他範例，請參閱 A thena 文件。

冰山, 表, 解剖學

現在，我們已經介紹了使用 Iceberg 桌子的基本步驟，讓我們深入了解冰山餐桌的複雜細節和設計。

為了啟用本指南前面描述的功能，Iceberg 採用了數據和元數據文件的分層層設計。這些層可以智慧地管理中繼資料，以最佳化查詢規劃和執行。

下圖透過兩個方面描繪了 Iceberg 表的組織： AWS 服務 用於在 Amazon S3 中存放資料表和檔案放置。



如圖所示，冰山表由三個主要層組成：

- Iceberg 目錄： AWS Glue Data Catalog 與 Iceberg 原生整合，對於大多數使用案例而言，是執行在. AWS與 Iceberg 資料表互動的服務 (例如 Athena) 會使用目錄尋找資料表的目前快照版本，以便讀取或寫入資料。
- 中繼資料層：中繼資料檔案 (亦即資訊清單檔案和資訊清單清單檔案) 會追蹤資訊，例如資料表的結構描述、資料分割策略和資料檔案位置，以及資料行層級統計資料 (例如儲存在每個資料檔案中之記錄的最小和最大範圍)。這些中繼資料檔案存放在表格路徑內的 Amazon S3 中。
  - 清單文件包含每個數據文件的記錄，包括其位置，格式，大小，校驗和和其他相關信息。
  - 資訊清單清單提供資訊清單檔案的索引。隨著資料表中的資訊清單檔案數量增加，將該資訊分解為較小的子區段有助於減少需要由查詢掃描的資訊清單檔案數目。
  - 中繼資料檔案包含整個 Iceberg 資料表的相關資訊，包括資訊清單清單、結構描述、分割中繼資料、快照檔案，以及用來管理資料表中繼資料的其他檔案。
- 數據層：該層包含具有查詢將運行的數據記錄的文件。這些文件可以存儲在不同的格式，包括阿帕奇鑲木地板，阿帕奇阿夫羅和 Apache 的 OR C。
  - 數據文件包含一個表的數據記錄。
  - 刪除文件對 Iceberg 表中的行級刪除和更新操作進行編碼。Iceberg 有兩種類型的刪除文件，如冰山文檔中所述。這些文件是通過操作通過使用 merge-on-read 模式創建的。

與阿帕奇冰山在 Amazon EMR 工作

Amazon EMR 使用開放原始碼架構 (例如 Apache Spark、Apache Hive、Flink 和 Trito)，在雲端提供 PB 規模的資料處理、互動式分析和機器學習。

> **Note**
>
> 本指南使用 Apache 星火為例。

Amazon EMR 支持多種部署選項：Amazon EC2 上的 Amazon EMR，Amazon EKS 上的 Amazon EMR，Amazon EMR 無服務器和亞馬遜 EMR。 AWS Outposts若要為您的工作負載選擇部署選項，請參閱 Amazon EMR 常見問答集。

版本和功能兼容性

Amazon EMR 版本 6.5.0 及更高版本本地支持阿帕奇冰山。如需每個 Amazon EMR 版本支援的冰山版本清單，請參閱 Amazon EMR 文件中的冰山版本歷史記錄。 同時檢閱在 Amazon EMR 上使用 Iceberg 的考量和限制，以瞭解 Amazon EMR 在不同架構上支援哪些冰山功能。

我們建議您使用最新的 Amazon EMR 版本，以享受最新支援的冰山版本。本節中的程式碼範例和組態假設您使用的是 Amazon EMR 版本 em r-6.9.0。

使用冰山建立 Amazon EMR 叢集

若要在已安裝冰山的 Amazon EC2 上建立 Amazon EMR 叢集，請遵循亞馬遜 EM R 文件中的指示。

具體而言，您的叢集應設定下列分類：

```
[{
    "Classification": "iceberg-defaults",
    "Properties": {
        "iceberg.enabled": "true"
    }
}]
```

您也可以選擇使用 Amazon EMR 無伺服器或 Amazon EKS 上的 Amazon EMR 作為您的冰山工作負載的部署選項，從 Amazon EMR 6.6.0 開始。

在 Amazon EMR 中開發冰山應用

若要為您的 Iceberg 應用程式開發 Spark 程式碼，您可以使用 Amazon EMR Studio，這是一個以網路為基礎的整合式開發環境 (IDE)，適用於在 Amazon EMR 叢集上執行的全受管 Jupyter 筆記本電腦。

使用 Amazon EMR 工作室筆記本

您可以在 Amazon EMR 工作室工作區筆記本中以互動方式開發 Spark 應用程式，並將這些筆記本連接到 Amazon EC2 叢集上的 Amazon EMR 或 Amazon EKS 受管端點上的 Amazon EMR。有關在 Amazon Amazon EC2 和 Amazon EKS 上的 Amazon EMR 上設置 EMR 工作室的說明，請參閱 AWS 服務 文檔。

要在 EMR 工作室中使用冰山，請按照以下步驟操作：

1. 按照使用已安裝冰山的叢集中的指示，啟動已啟用冰山的 Amazon EMR 叢集。
2. 建立 EMR 工作室。如需指示，請參閱設定 Amazon EMR 工作室。
3. 開啟 EMR Studio 工作區筆記本，並執行下列程式碼做為筆記本中的第一個儲存格，以設定 Spark 工作階段以使用 Iceberg：

```
%%configure -f
{
    "conf": {
        "spark.sql.catalog.<catalog_name>": "org.apache.iceberg.spark.SparkCatalog",
        "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/YOUR-
FOLDER-NAME/",
        "spark.sql.catalog.<catalog_name>.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog",
        "spark.sql.catalog.<catalog_name>.io-impl":
 "org.apache.iceberg.aws.s3.S3FileIO",
        "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions"
    }
}
```

其中：

- <catalog_name>是您的冰山星火工作階段目錄名稱。將其取代為目錄的名稱，並記得在與此型錄相關聯的所有模型組態中變更參考。然後，在您的程式碼中，您應該使用完整資料表名稱 (包括 Spark 工作階段目錄名稱) 來參考 Iceberg 資料表，如下所示：

```
<catalog_name>.<database_name>.<table_name>
```

- `<catalog_name>.warehouse`指向您要存放資料和中繼資料的 Amazon S3 路徑。
- 若要將目錄設為 AWS Glue Data Catalog，請`<catalog_name>.catalog-impl` 將設定為`org.apache.iceberg.aws.glue.GlueCatalog`。若要指向任何自訂目錄實作的實作類別，都需要此金鑰。本指南後面的「一般最佳做法」一節說明了不同的 Iceberg 支援的目錄。
- 用`org.apache.iceberg.aws.s3.S3FileIO`作，以便`<catalog_name>.io-impl`利用 Amazon S3 多部分上傳以實現高平行性。

4. 現在，您可以開始在筆記本中以互動方式開發 Iceberg 的 Spark 應用程序，就像您對任何其他 Spark 應用程序一樣。

如需使用 Amazon EMR Studio 為 Apache 冰山設定 Spark 的詳細資訊，請參閱部落格文章在 Amazon EMR 上使用 Apache 冰山建置高效能、ACID 合規且不斷發展的資料湖。

在 Amazon EMR 中執行冰山工作

開發適用於 Iceberg 工作負載的 Spark 應用程式程式碼後，您可以在任何支援 Iceberg 的 Amazon EMR 部署選項上執行該程式碼 (請參閱 Amazon EM R 常見問答集)。

與其他 Spark 任務一樣，您可以透過新增步驟或以互動方式將 Spark 任務提交至主節點，將工作提交至 Amazon EC2 叢集上的 Amazon EMR。若要執行星火任務，請參閱下列 Amazon EMR 文件頁面：

- 如需將工作提交至 Amazon EC2 叢集上 Amazon EMR 的不同選項概觀，以及每個選項的詳細指示，請參閱將工作提交到叢集。
- 對於 Amazon EKS 上的 Amazon EMR，請參閱使用運行 Spark 任務。 StartJobRun
- 如需 Amazon EMR 無伺服器資訊，請參閱執行任務。

以下各節提供每個 Amazon EMR 部署選項的範例。

Amazon 在亞馬 Amazon EC2 上的 EMR

您可以使用以下步驟提交冰山星火工作：

1. 在工作站上建立`emr_step_iceberg.json`包含下列內容的檔案：

```
[{
    "Name": "iceberg-test-job",
```

```
    "Type": "spark",
    "ActionOnFailure": "CONTINUE",
    "Args": [
        "--deploy-mode",
        "client",
        "--conf",

  "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
        "--conf",
        "spark.sql.catalog.<catalog_name>=org.apache.iceberg.spark.SparkCatalog",
        "--conf",
        "spark.sql.catalog.<catalog_name>.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog",
        "--conf",
        "spark.sql.catalog.<catalog_name>.warehouse=s3://YOUR-BUCKET-NAME/YOUR-
FOLDER-NAME/",
        "--conf",
        "spark.sql.catalog.<catalog_name>.io-
impl=org.apache.iceberg.aws.s3.S3FileIO",
        "s3://YOUR-BUCKET-NAME/code/iceberg-job.py"
    ]
}]
```

2. 通過自定義以粗體突出顯示的 Iceberg 配置選項來修改特定 Spark 作業的配置文件。
3. 使用 AWS Command Line Interface (AWS CLI) 提交步驟。在`emr_step_iceberg.json`檔案所在的目錄中執行命令。

```
aws emr add-steps --cluster-id <cluster_id> --steps file://emr_step_iceberg.json
```

Amazon EMR Serverless

若要使用以下命令將冰山星火任務提交給 Amazon EMR 無伺服器： AWS CLI

1. 在工作站上建立`emr_serverless_iceberg.json`包含下列內容的檔案：

```
{
    "applicationId": "<APPLICATION_ID>",
    "executionRoleArn": "<ROLE_ARN>",
    "jobDriver": {
        "sparkSubmit": {
            "entryPoint": "s3://YOUR-BUCKET-NAME/code/iceberg-job.py",
```

```
            "entryPointArguments": [],
            "sparkSubmitParameters": "--jars /usr/share/aws/iceberg/lib/iceberg-
spark3-runtime.jar"
        }
    },
    "configurationOverrides": {
        "applicationConfiguration": [{
            "classification": "spark-defaults",
            "properties": {
                "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
                "spark.sql.catalog.<catalog_name>":
 "org.apache.iceberg.spark.SparkCatalog",
                "spark.sql.catalog.<catalog_name>.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog",
                "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/
YOUR-FOLDER-NAME/",
                "spark.sql.catalog.<catalog_name>.io-impl":
 "org.apache.iceberg.aws.s3.S3FileIO",
                "spark.jars":"/usr/share/aws/iceberg/lib/iceberg-spark3-runtime.jar",

 "spark.hadoop.hive.metastore.client.factory.class":"com.amazonaws.glue.catalog.metastore.AWS
            }
        }],
        "monitoringConfiguration": {
            "s3MonitoringConfiguration": {
                "logUri": "s3://YOUR-BUCKET-NAME/emr-serverless/logs/"
            }
        }
    }
}
```

2. 通過自定義以粗體突出顯示的 Iceberg 配置選項來修改特定 Spark 作業的配置文件。
3. 使用提交工作 AWS CLI。在`emr_serverless_iceberg.json`檔案所在的目錄中執行命令：

```
aws emr-serverless start-job-run --cli-input-json file://emr_serverless_iceberg.json
```

若要使用 EMR 工作室主控台將冰山星火任務提交給 Amazon EMR 無伺服器：

1. 請遵循 Amazon EMR 無伺服器文件中的指示進行。

2. 對於 Job 配置，請使用為 Spark 提供的冰山配置， AWS CLI 並自定義冰山突出顯示的字段。如需詳細指示，請參閱 Amazon EMR 文件中的將 Apache 冰山與無伺服器搭配使用。

Amazon EMR 在 Amazon EKS

要通過使用以下命令向 Amazon EKS 上的 Amazon EMR 提交冰山火花任務： AWS CLI

1. 在工作站上建立emr_eks_iceberg.json包含下列內容的檔案：

```
{
    "name": "iceberg-test-job",
    "virtualClusterId": "<VIRTUAL_CLUSTER_ID>",
    "executionRoleArn": "<ROLE_ARN>",
    "releaseLabel": "emr-6.9.0-latest",
    "jobDriver": {
        "sparkSubmitJobDriver": {
            "entryPoint": "s3://YOUR-BUCKET-NAME/code/iceberg-job.py",
            "entryPointArguments": [],
            "sparkSubmitParameters": "--jars local:///usr/share/aws/iceberg/lib/
iceberg-spark3-runtime.jar"
        }
    },
    "configurationOverrides": {
        "applicationConfiguration": [{
            "classification": "spark-defaults",
            "properties": {
                "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
                "spark.sql.catalog.<catalog_name>":
 "org.apache.iceberg.spark.SparkCatalog",
                "spark.sql.catalog.<catalog_name>.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog",
                "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/
YOUR-FOLDER-NAME/",
                "spark.sql.catalog.<catalog_name>.io-impl":
 "org.apache.iceberg.aws.s3.S3FileIO",
                "spark.hadoop.hive.metastore.client.factory.class":
 "com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory"
            }
        }],
        "monitoringConfiguration": {
            "persistentAppUI": "ENABLED",
```

```
            "s3MonitoringConfiguration": {
                "logUri": "s3://YOUR-BUCKET-NAME/emr-serverless/logs/"
            }
        }
    }
}
```

2. 通過自定義以粗體突出顯示的 Iceberg 配置選項來修改 Spark 作業的配置文件。
3. 使用提交工作 AWS CLI。在`emr_eks_iceberg.json`檔案所在的目錄中執行下列命令：

```
aws emr-containers start-job-run --cli-input-json file://emr_eks_iceberg.json
```

有關詳細說明，請參閱在 EKS 上的 Amazon EMR 文檔中使用阿帕奇冰山與 Amazon EMR 在 EKS 上。

Amazon EMR 的最佳實踐

本節提供在 Amazon EMR 中調整 Spark 任務的一般準則，以最佳化資料讀取和寫入冰山資料表。有關特定於 Iceberg 的最佳實踐，請參閱本指南後面的最佳實踐部分。

- 使用最新版本的 Amazon EMR — Amazon EMR 通過 Amazon EMR 星火運行時提供開箱即用的 Spark 優化。 AWS 改善每個新版本的 Spark 執行階段引擎效能。
- 決定 Spark 工作負載的最佳基礎架構 — Spark 工作負載可能需要不同類型的硬體來處理不同的工作特性，以確保最佳效能。Amazon EMR 支援多種執行個體類型 (例如運算優化、記憶體優化、一般用途和儲存最佳化)，以涵蓋所有類型的處理需求。當您上架新的工作負載時，建議您使用一般執行個體類型 (例如 M5 或 M6g) 進行基準測試。監控 Ganglia 和 Amazon 的作業系統 (OS) 和 YARN 指標， CloudWatch 以判斷尖峰負載時的系統瓶頸 (CPU、記憶體、儲存和 I/O)，並選擇適當的硬體。
- 微調 `spark.sql.shuffle.partitions` — 將內`spark.sql.shuffle.partitions`容設定為叢集中虛擬核心 (vCore) 總數，或設定為該值的倍數 (通常是 vCore 總數的 1 到 2 倍)。當您使用雜湊和範圍分割做為寫入分配模式時，此設定會影響 Spark 的平行處理原則。它在寫入組織數據之前請求洗牌，以確保分區對齊。
- 啟用受管擴展 — 對於幾乎所有使用案例，我們建議您啟用受管擴展和動態配置。不過，如果您的工作負載具有可預測模式，建議您停用自動調整規模和動態配置。啟用受管擴展後，建議您使用 Spot 執行個體降低成本。針對任務節點使用 Spot 執行個體，而非核心或主節點。使用 Spot 執行個體時，請使用具有每個叢集多個執行個體類型的執行個體叢集，以確保 Spot 可用性
- 盡可能使用廣播聯接-廣播（mapside）聯接是最佳的聯接，只要您的其中一個表足夠小以適應最小節點的內存（按 MB 的順序），並且您正在執行 equi（=）連接。支援除完整外部聯結以外的所有聯

結類型。廣播聯接將較小的表作為哈希表廣播，跨內存中的所有工作節點。小桌子廣播後，您將無法對其進行更改。由於哈希表位於 Java 虛擬機（JVM）中的本地，因此可以通過使用散列連接，根據連接條件輕鬆地將其與大表合併。由於最小的隨機播放額外負荷，廣播連接可提供高效能。

- 調整記憶體回收器 — 如果記憶體回收 (GC) 週期很慢，請考慮從預設的 parallel 記憶體回收器切換至 G1GC 以獲得更好的效能。若要最佳化 GC 效能，您可以微調 GC 參數。要跟踪 GC 性能，您可以使用 Spark UI 對其進行監視。理想情況下，GC 時間應小於或等於總工作執行階段的 1%。

與阿帕奇冰山一起工作 AWS Glue

AWS Glue是一種無伺服器資料整合服務，可讓您更輕鬆地探索、準備、移動和整合來自多個來源的資料，以進行分析、機器學習 (ML) 和應用程式開發。的核心功能之一 AWS Glue 是能夠以簡單且符合成本效益的方式執行擷取、轉換和載入 (ETL) 作業。這有助於分類您的資料、清理資料、豐富資料，以及在各種資料存放區和資料串流之間可靠地移動資料。

AWS Glue 工作封裝使用 Apache 星火或 Python 執行階段定義轉換邏輯的指令碼。 AWS Glue 工作可以在批次和串流模式下執行。

當您在中建立 Iceberg 工作時 AWS Glue，視版本而定 AWS Glue，您可以使用原生 Iceberg 整合或自訂 Iceberg 版本，將 Iceberg 相依性附加至工作。

使用原生冰山整合

AWS Glue 3.0 和 4.0 版本本地支持事務數據湖格式，如阿帕奇冰山，阿帕奇胡迪，和 Linux 基金會三角洲湖在 AWS Glue 星火。此整合功能可簡化中開始使用這些架構所需的組態步驟 AWS Glue。

若要為您的 AWS Glue Job 啟用 Iceberg 支援，請設定 Job：選擇工作的 [工作詳細資料] 索引標籤，捲動至 [進階屬性] 下的 [工作參數]，然後將金鑰設定為`--datalake-formats`及其值為`iceberg`。AWS Glue

如果您使用筆記本編寫工作，您可以使用下列`%%configure`魔法設定第一個筆記本儲存格中的參數：

```
%%configure
{
  "--conf" : <job-specific Spark configuration discussed later>,
  "--datalake-formats" : "iceberg"
}
```

使用自定義的冰山版本

在某些情況下，您可能希望保留對工作的 Iceberg 版本的控制權，並按照自己的步調對其進行升級。例如，升級到更新版本可以解除對新功能的存取權限和效能增強功能。若要搭配使用特定的 Iceberg 版本 AWS Glue，您可以使用自訂連接器或您自己的 JAR 檔案。

使用自訂連接器

AWS Glue 支援連接器，這是可選的程式碼套件，可協助存取 AWS Glue Studio. 您可以在中訂閱連接器 AWS Marketplace，也可以建立自訂連接器。

> **Note**
>
> AWS Marketplace 提供了阿帕奇冰山連接器 AWS Glue. 但是，我們建議您改用自訂連接器來保留對 Iceberg 版本的控制權。

例如，若要為 Iceberg 版本 0.13.1 建立客戶連接器，請依照下列步驟執行：

1. 將文件`iceberg-spark-runtime-3.1_2.12-0.13.1.jar`上`bundle-2.17.161.jar`傳`url-connection-client-2.17.161.jar`到 Amazon S3 存儲桶。您可以從各自的 Apache Maven 存儲庫中下載這些文件。
2. 在AWS Glue Studio 主控台上，建立自訂 Spark 連接器：
   a. 在導覽窗格中，選擇 [資料連線]。(如果您使用的是舊版導覽，請選擇 [連接器]、[建立自訂連接器])。
   b. 在 [連接器] 方塊中選擇 [建立自訂連接器]。
   c. 在 [建立自訂連接器] 頁面上：
      - 在 Amazon S3 中指定 JAR 檔案的路徑。
      - 輸入連接器的名稱。
      - 選擇星火作為連接器類型。
      - 對於 [類別名稱]，指定使用`format`運算子載入 Spark 資料來源時所使用的完整資料來源類別名稱 (或其別名)。
      - (選擇性) 提供連接器的描述。
3. 選擇 Create connector (建立連接器)。

在中使用連接器時 AWS Glue，您必須建立連接器的連接。連接包含連接至特定資料倉庫所需的性質。您在 ETL 任務中將連線用於資料來源和資料目標。連接器和連線搭配運作，以方便存取資料存放區。

若要使用您建立的自訂 Iceberg 連接器來建立連線：

1. 在AWS Glue Studio 主控台上，選取您的自訂 Iceberg 連接器。

2. 依照提示提供詳細資料 (例如 VPC 和工作所需的其他網路組態)，然後選擇 [建立連線]。

您現在可以在 AWS Glue ETL 工作中使用連線。視您建立工作的方式而定，有不同的方式可將連線附加至工作：

- 如果您使用建立視覺化工作 AWS Glue Studio，則可以從 [資料來源內容 — 連接器] 索引標籤上的 [連線] 清單中選取連線。
- 如果您在筆記本中開發工作，請使用`%connections`魔術來設置連接名稱：

```
%glue_version 3.0

%connections <name-of-the iceberg-connection>

%%configure
{
  "--conf" : "job-specific Spark configurations, to be discussed later",
  "--datalake-formats" : "iceberg"
}
```

- 如果您使用指令碼編輯器來編寫 Job，請在 [工作詳細資料] 索引標籤的 [進階內容]、[其他網路連線] 下方指定連線。

如需本節中程序的詳細資訊，請參閱 AWS Glue 文件 AWS Glue Studio中的使用連接器和連線。

使用您自己的 JAR 檔案

在中 AWS Glue，您也可以使用 Iceberg，而無需使用連接器。當您想要保留對 Iceberg 版本的控制權並快速更新時，此方法非常有用。若要使用此選項，請將所需的 Iceberg JAR 檔案上傳到您選擇的 S3 儲存貯體，並參考 AWS Glue 工作中的檔案。例如，如果您正在使用冰山 1.0.0，則所需的 JAR 文件是`iceberg-spark-runtime-3.0_2.12-1.0.0.jar`、`url-connection-client-2.15.40.jar`和。`bundle-2.15.40.jar`您也可以將工作的`--user-jars-first`參數設定`true`為，來排定類別路徑中其他 JAR 檔案的優先順序。

冰山的火花配置 AWS Glue

本節討論為冰山資料集編寫 AWS Glue ETL 工作所需的 Spark 組態。 您可以通過使用 Spark 鍵與所有 S `--conf` park 配置鍵和值的逗號分隔列表來設置這些配置。您可以在筆記本或 AWS Glue Studio 控制台的 Job 參數部分中使用`%%configure`魔術。

```
%glue_version 3.0

%connections <name-of-the iceberg-connection>

%%configure
{
  "--conf" : "spark.sql.extensions=org.apache.iceberg.spark.extensions...",
  "--datalake-formats" : "iceberg"
}
```

使用以下屬性配置 Spark 會話：

- <catalog_name>是您的冰山星火工作階段目錄名稱。將其取代為目錄的名稱，並記得在與此型錄相關聯的所有模型組態中變更參考。然後，在您的代碼中，您應該使用完全限定的表名稱（包括 Spark 會話目錄名稱）引用您的 Iceberg 表，如下所示：<catalog_name>.<database_name>.<table_name>。
- <catalog_name>.<warehouse>指向您要存放資料和中繼資料的 Amazon S3 路徑。
- 若要將目錄設為 AWS Glue Data Catalog，請<catalog_name>.catalog-impl將設定為org.apache.iceberg.aws.glue.GlueCatalog。需要此金鑰才能指向任何自訂目錄實作的實作類別。有關 Iceberg 支援的目錄，請參閱本指南後???面的「一般最佳做法」一節。
- 用org.apache.iceberg.aws.s3.S3FileIO作以<catalog_name>.io-impl利用 Amazon S3 多部分上傳以實現高平行性。

例如，如果您有名為的目錄glue_iceberg，您可以使用多個--conf金鑰來設定工作，如下所示：

```
%%configure
{
  "--datalake-formats" : "iceberg",
  "--conf" :
 "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
  "--conf" : "spark.sql.catalog.glue_iceberg=org.apache.iceberg.spark.SparkCatalog",
  "--conf" : "spark.sql.catalog.glue_iceberg.warehouse=s3://<your-warehouse-dir>=>/",
  "--conf" : " spark.sql.catalog.glue_iceberg.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog ",
  "--conf" : " spark.sql.catalog.glue_iceberg.io-
impl=org.apache.iceberg.aws.s3.S3FileIO
}
```

或者，您可以使用代碼將上述配置添加到 Spark 腳本中，如下所示：

```
spark = SparkSession.builder\

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
                    .config("spark.sql.catalog.glue_iceberg",
 "org.apache.iceberg.spark.SparkCatalog")\
                    .config("spark.sql.catalog.glue_iceberg.warehouse","s3://<your-
warehouse-dir>/")\
                    .config("spark.sql.catalog.glue_iceberg.catalog-impl",
 "org.apache.iceberg.aws.glue.GlueCatalog") \
                    .config("spark.sql.catalog.glue_iceberg.io-impl",
 "org.apache.iceberg.aws.s3.S3FileIO") \
                    .getOrCreate()
```

AWS Glue 工作的最佳做法

本節提供調整 Spark 工作的一般準則， AWS Glue 以最佳化 Iceberg 資料表的讀取和寫入資料。有關特定於 Iceberg 的最佳實踐，請參閱本指南後面的最佳實踐部分。

- 使用最新版本的 AWS Glue 並儘可能升級 — 新版本 AWS Glue 提供效能改善、縮短啟動時間和新功能。它們還支持最新的冰山版本可能需要的較新 Spark 版本。如需可用 AWS Glue 版本及其支援的 Spark 版本清單，請參閱AWS Glue 文件。
- 最佳化 AWS Glue 工作記憶體 — 請遵循中「最佳化記憶體管理」 AWS 部落格文章中的建議 AWS Glue。
- 使用 AWS Glue Auto Scaling 整 — 當您啟用「 AWS Glue 自動調整 Auto Scaling 例」時，會根據您的 AWS Glue 工作負載動態調整工作程式的數量。這有助於降低尖峰負載期間的工 AWS Glue 作成本，因為當工作量很小且工作者閒置時，可 AWS Glue 縮減工作人員的數量。若要使用「 AWS Glue Auto Scaling」，您可以指定工 AWS Glue 作可擴充至的最大 Worker 數目。如需詳細資訊，請參閱 AWS Glue 文件AWS Glue 中的〈使用 auto 調整比例〉。
- 使用自訂連接器或新增程式庫相依性-Iceberg 的 AWS Glue 原生整合最適合開始使用 Iceberg。但是，對於生產工作負載，我們建議您使用自訂容器或新增程式庫相依性 (如本指南前面所述)，以完全控制 Iceberg 版本。這種方法可幫助您從最新的 Iceberg 功能和 AWS Glue 工作的性能改進中受益。
- 啟用 Spark UI 以進行監視和除錯 — 您也可以在中使用 Spark UI AWS Glue 來檢查您的 Iceberg 工作，方法是在有向非循環圖 (DAG) 中將 Spark 工作的不同階段視覺化，並詳細監控工作。Spark UI 提供了一種有效的方法來排除故障和優化冰山工作。例如，您可以識別擁有大量洗牌或磁碟溢出的瓶頸階段，以識別調整機會。如需詳細資訊，請參閱 AWS Glue 文件中的使用 Apache Spark 網路使用者介面監視工作。

使用 Apache 星火與阿帕奇冰山表的工作

本節提供使用 Apache 星火與冰山表進行交互的概述。這些示例是可以在 Amazon EMR 或上運行的樣板代碼。 AWS Glue

注意：與冰山表進行交互的主要接口是 SQL，因此大多數示例將 Spark SQL 與 DataFrames API 相結合。

創建和編寫冰山表

您可以使用星火 SQL 和星火 DataFrames 創建和數據添加到冰山表。

使用星火 SQL

若要撰寫冰山資料集，請使用標準 Spark SQL 陳述式，例如`CREATE TABLE`和`INSERT INTO`。

未分割資料表

以下是使用 Spark SQL 創建未分區的冰山表的示例：

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions (
        c_customer_sk             int,
        c_customer_id             string,
        c_first_name              string,
        c_last_name               string,
        c_birth_country           string,
        c_email_address           string)
    USING iceberg
    OPTIONS ('format-version'='2')
""")
```

若要將資料插入未分割資料表，請使用標準`INSERT INTO`陳述式：

```
spark.sql(f"""
INSERT INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions
SELECT c_customer_sk, c_customer_id, c_first_name, c_last_name, c_birth_country,
 c_email_address
FROM another_table
```

```
""")
```

分割的資料表

以下是使用 Spark SQL 創建分區冰山表的示例：

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions (
        c_customer_sk             int,
        c_customer_id             string,
        c_first_name              string,
        c_last_name               string,
        c_birth_country           string,
        c_email_address           string)
    USING iceberg
    PARTITIONED BY (c_birth_country)
    OPTIONS ('format-version'='2')
""")
```

要使用 Spark SQL 將數據插入分區的 Iceberg 表中，請執行全局排序，然後寫入數據：

```
spark.sql(f"""
INSERT INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions
SELECT c_customer_sk, c_customer_id, c_first_name, c_last_name, c_birth_country,
 c_email_address
FROM another_table
ORDER BY c_birth_country
""")
```

應用 DataFrames 程式介面

要編寫冰山數據集，您可以使用 DataFrameWriterV2 API。

要創建一個 Iceberg 表並將數據寫入到它，請使用 df.writeTo(t) 函數。如果資料表存在，請使用.append()函數。如果沒有，請使.create().用以下示例使用.createOrReplace()，這是相當於的.create()變體CREATE OR REPLACE TABLE AS SELECT。

未分割資料表

若要使用 API 建立並填入未分割的DataFrameWriterV2冰山資料表：

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions") \
    .tableProperty("format-version", "2") \
    .createOrReplace()
```

若要使用 API 將資料插入現有的未分割 Iceberg 資料表：DataFrameWriterV2

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions") \
    .append()
```

分割的資料表

若要使用 DataFrameWriterV2 API 建立並填入分區的 Iceberg 資料表，您可以使用本機排序來擷取資料：

```
input_data.sortWithinPartitions("c_birth_country") \
    .writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions") \
    .tableProperty("format-version", "2") \
    .partitionedBy("c_birth_country") \
    .createOrReplace()
```

若要使用 DataFrameWriterV2 API 將資料插入分區的 Iceberg 資料表中，您可以使用全域排序來擷取資料：

```
input_data.orderBy("c_birth_country") \
    .writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions") \
    .append()
```

更新冰山表中的數據

下面的例子演示了如何更新冰山表中的數據。此範例會修改資料行中具有偶數的所有資料c_customer_sk列。

```
spark.sql(f"""
UPDATE {CATALOG_NAME}.{db.name}.{table.name}
SET c_email_address = 'even_row'
WHERE c_customer_sk % 2 == 0
""")
```

此作業使用預設 copy-on-write 策略，因此會重寫所有受影響的資料檔案。

在冰山表中提升數據

Upserting 數據是指在單個事務中插入新的數據記錄和更新現有數據記錄。要將數據提升到冰山表中，請使用語句。SQL MERGE INTO

下面的例子提高了表中的表格內容{UPSERT_TABLE_NAME} 的內容：{TABLE_NAME}

```
spark.sql(f"""
    MERGE INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME} t
    USING {UPSERT_TABLE_NAME} s
        ON t.c_customer_id = s.c_customer_id
    WHEN MATCHED THEN UPDATE SET t.c_email_address = s.c_email_address
    WHEN NOT MATCHED THEN INSERT *
""")
```

- 如果中{UPSERT_TABLE_NAME}已有相同的客戶記錄c_customer_id，則{UPSERT_TABLE_NAME}記錄c_email_address值會覆寫現有值 (更新作業)。{TABLE_NAME}
- 如果中{UPSERT_TABLE_NAME}不存在的客戶記錄{TABLE_NAME}，則會將記{UPSERT_TABLE_NAME}錄新增至 {TABLE_NAME} (插入作業)。

刪除冰山表中的數據

若要從 Iceberg 資料表中刪除資料，請使用DELETE FROM運算式並指定符合要刪除之資料列的篩選器。

```
spark.sql(f"""
DELETE FROM {CATALOG_NAME}.{db.name}.{table.name}
WHERE c_customer_sk % 2 != 0
""")
```

如果過濾器與整個分區匹配，Iceberg 會執行僅元數據刪除並保留數據文件。否則，它只會重寫受影響的資料檔案。

delete 方法採用受WHERE子句影響的數據文件，並創建它們的副本，而不刪除的記錄。然後，它會建立指向新資料檔案的新資料表快照集。因此，刪除的記錄仍然存在於資料表的較舊快照集中。例如，如果您擷取資料表的上一個快照，您會看到剛才刪除的資料。如需有關移除不需要的舊快照及相關資料檔案以進行清理的詳細資訊，請參閱本指南稍後的 < 使用壓縮維護檔案 > 一節。

讀取資料

您可以在 Spark 中閱讀冰山表的最新狀態，同時使用 Spark SQL 和 DataFrames.

使用星火 SQL 的示例：

```
spark.sql(f"""
SELECT * FROM {CATALOG_NAME}.{db.name}.{table.name} LIMIT 5
""")
```

使用 DataFrames API 的示例：

```
df = spark.table(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}").limit(5)
```

使用時間旅行

在 Iceberg 表中的每個寫操作（插入，更新，更新，刪除）創建一個新的快照。然後，您可以使用這些快照進行時間旅行 — 回到過去並檢查過去表格的狀態。

如需如何使用`snapshot-id`和計時值擷取資料表快照記錄的詳細資訊，請參閱本指南稍後的 < 存取中繼資料 > 一節。

下面的時間旅行查詢顯示基於特定的表的狀態`snapshot-id`。

使用星火 SQL：

```
spark.sql(f"""
SELECT * FROM {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME} VERSION AS OF {snapshot_id}
""")
```

使用 DataFrames 應用程式介面：

```
df_1st_snapshot_id = spark.read.option("snapshot-id", snapshot_id) \
     .format("iceberg") \
     .load(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}") \
     .limit(5)
```

下列時間行程查詢會根據在特定時間戳記之前建立的最後一個快照 (以毫秒為單位`as-of-timestamp`) 來顯示資料表的狀態。

使用星火 SQL：

```
spark.sql(f"""
SELECT * FROM dev.{db.name}.{table.name} TIMESTAMP AS OF '{snapshot_ts}'
""")
```

使用 DataFrames 應用程式介面：

```
df_1st_snapshot_ts = spark.read.option("as-of-timestamp", snapshot_ts) \
                          .format("iceberg") \
                          .load(f"dev.{DB_NAME}.{TABLE_NAME}") \
                          .limit(5)
```

使用增量查詢

您也可以使用 Iceberg 快照以增量方式讀取附加的資料。

注意：此作業目前支援從append快照讀取資料。它不支持從操作（例如replace，，overwrite或delete）中獲取數據。 此外，Spark SQL 語法不支援增量讀取作業。

下列範例會擷取快照 start-snapshot-id (獨佔) 與 end-snapshot-id (含) 之間附加至 Iceberg 資料表的所有記錄。

```
df_incremental = (spark.read.format("iceberg")
    .option("start-snapshot-id", snapshot_id_start)
    .option("end-snapshot-id", snapshot_id_end)
    .load(f"glue_catalog.{DB_NAME}.{TABLE_NAME}")
)
```

訪問元數據

冰山提供了通過 SQL 訪問其元數據。您可以透過查詢命名空間來存取任何指定 table (<table_name>) 的中繼資料<table_name>.<metadata_table>。 如需中繼資料表的完整清單，請參閱 Iceberg 文件中的檢查表。

下面的示例演示了如何訪問 Iceberg 歷史元數據表，該表顯示了 Iceberg 表的提交（更改）的歷史記錄。

使用星火 SQL (與%%sql魔術) 從 Amazon EMR 工作室筆記本:

```
Spark.sql(f“””
SELECT * FROM {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}.history LIMIT 5
""")
```

使用 DataFrames 應用程式介面：

```
spark.read.format("iceberg").load("{CATALOG_NAME}.{DB_NAME}.
{TABLE_NAME}.history").show(5,False)
```

輸出範例：

Type: Table Pie Scatter Line Area Bar

made_current_at	snapshot_id	parent_id	is_current_ancestor
2023-01-09 02:50:17.547000+00:00	7501027970051178613	6598755163776233735	True
2023-01-12 05:39:29.567000+00:00	7069175828427777019	7501027970051178613	True
2023-01-12 05:39:58.807000+00:00	5173022175861138222	7069175828427777019	True
2023-01-12 05:40:18.499000+00:00	3703414997660223390	5173022175861138222	True
2023-01-12 05:40:41.827000+00:00	3807904412292252460	3703414997660223390	True

使用 Amazon Athena SQL 與阿帕奇冰山表

Amazon Athena 為 Apache 冰山提供內建支援，不需要額外的步驟或設定。本節提供支援功能的詳細概觀，以及使用 Athena 與 Iceberg 資料表互動的高階指引。

版本和功能兼容性

> **Note**
>
> 以下各節假設您使用的是 Athena 引擎第 3 版。

冰山表規格支持

Apache 的冰山表規範指定了冰山表應該如何行為。Athena 支援資料表格式第 2 版，因此您使用主控台、CLI 或 SDK 建立的任何 Iceberg 表格本質上都會使用該版本。

如果您使用與另一個引擎（例如 Amazon EMR 上的 Apache Spark）創建的冰山表 AWS Glue，或者確保使用表屬性設置表格式版本。作為參考，請參閱本指南前面的創建和編寫 Iceberg 表一節。

冰山功能支持

您可以使用 Athena 來讀取和寫入冰山表。當您使用`UPDATE`、`MERGE INTO`和`DELETE FROM`陳述式變更資料時，Athena 僅支援 merge-on-read 模式。無法變更此屬性。為了更新或刪除數據 copy-on-write，你必須使用其他引擎，如阿帕奇星火在 Amazon EMR 或 AWS Glue. 下表總結了 Athena 的冰山功能支援。

		支援 DDL		DML 支援		AWS Lake Formation 為了安全（可選）
	資料表格式	建立資料表	結構描述演進	讀取資料	寫入資料	列/欄存取控制
Amazon Athena	2 版	✓	✓	✓	X C opy-on-write	✓

		支援 DDL	DML 支援		AWS Lake Formation 為了安全（可選）
				✓ 米 erge-on-read	✓

> **Note**
>
> Athena 不支援增量查詢。

使用冰山桌

如需在 Athena 使用冰山的快速入門，請參閱本指南稍早的 A thena SQL 中的「開始使用冰山表」一節。

下表列出限制和建議。

情況	限制	建議
產生資料表 DDL	使用其他引擎建立的冰山表格可能具有未暴露在 Athena 中的屬性。對於這些表格，無法產生 DDL。	在建立資料表的引擎中使用對等陳述式 (例如 Spark `SHOW CREATE TABLE` 陳述式)。
寫入冰山表的對象中的隨機 Amazon S3 前綴	依預設，使用 Athena 建立的冰山表格會啟用`write.object-storage.enabled` 屬性。	要禁用此行為並獲得對冰山表屬性的完全控制，請使用另一個引擎（例如 Amazon EMR 上的 Spark）創建一個冰山表或。 AWS Glue
增量查詢	Athena 目前不支援。	若要使用增量查詢來啟用增量資料擷取管道，請使用 Amazon EMR 上的 Spark 或. AWS Glue

將現有的表遷移到冰山

若要將目前的 Athena 或資料 AWS Glue 表 (也稱為 Hive 資料表) 遷移為 Iceberg 格式，您可以使用就地或完整資料遷移：

- 就地遷移是在現有數據文件之上生成 Iceberg 的元數據文件的過程。
- 完整資料遷移會建立 Iceberg 中繼資料層，並將現有資料檔案從原始資料表重新寫入新的 Iceberg 資料表。

下列各節提供可用於移轉表格的 API 概觀，以及選擇移轉策略的指引。有關這兩種策略的更多信息，請參閱 Iceberg 文檔中的表遷移部分。

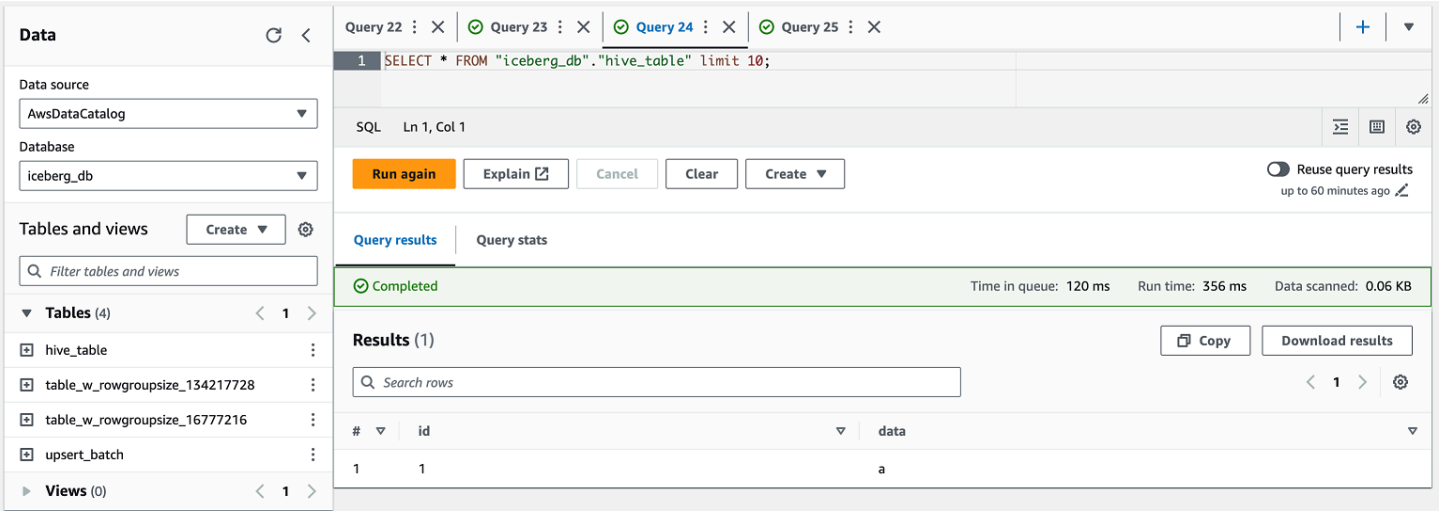
就地移轉

就地移轉不需要重新寫入所有資料檔案。而是會產生 Iceberg 中繼資料檔案，並將其連結至您現有的資料檔案。Iceberg 提供了三種實施就地遷移的選項：

- 使用程`snapshot`序，如冰山文件中的快照表和 Spark 程序：快照一節所述。
- 使用程`add_files`序，如冰山文件中的「新增檔案和星火程序：add_files」一節所述。
- 使用程`migrate`序，如移轉資料表和 Spark 程序：在冰山文件中移轉一節所述。

目前，遷移過程不能直接與 AWS Glue Data Catalog-它僅適用於 Hive 中繼存儲。如果您需要使用`migrate`程序而非`snapshot`或`add_files`，則可以將臨時 Amazon EMR 叢集與 Hive 中繼存放區 (HMS) 搭配使用。這種方法需要冰山版本 1.2 或更高版本。

假設您要創建以下 Hive 表：



您可以在 Athena 主控台中執行此程式碼，以建立此 Hive 資料表：

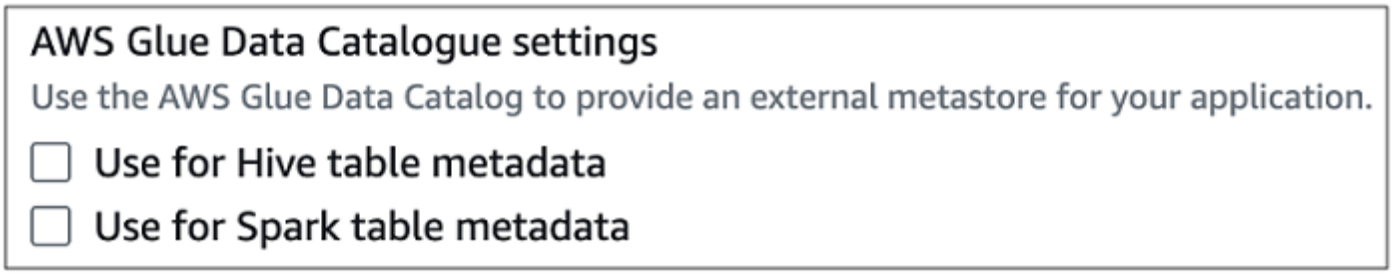
```
CREATE EXTERNAL TABLE 'hive_table'(
  'id' bigint,
  'data' string)
USING parquet
LOCATION
  's3://datalake-xxxx/aws_workshop/iceberg_db/hive_table'

INSERT INTO iceberg_db.hive_table VALUES (1, 'a')
```

如果您的 Hive 表被分區，包括分區語句，並根據 Hive 要求添加分區。

```
ALTER TABLE default.placeholder_table_for_migration ADD
    PARTITION (date = '2023-10-10')
```

步驟：

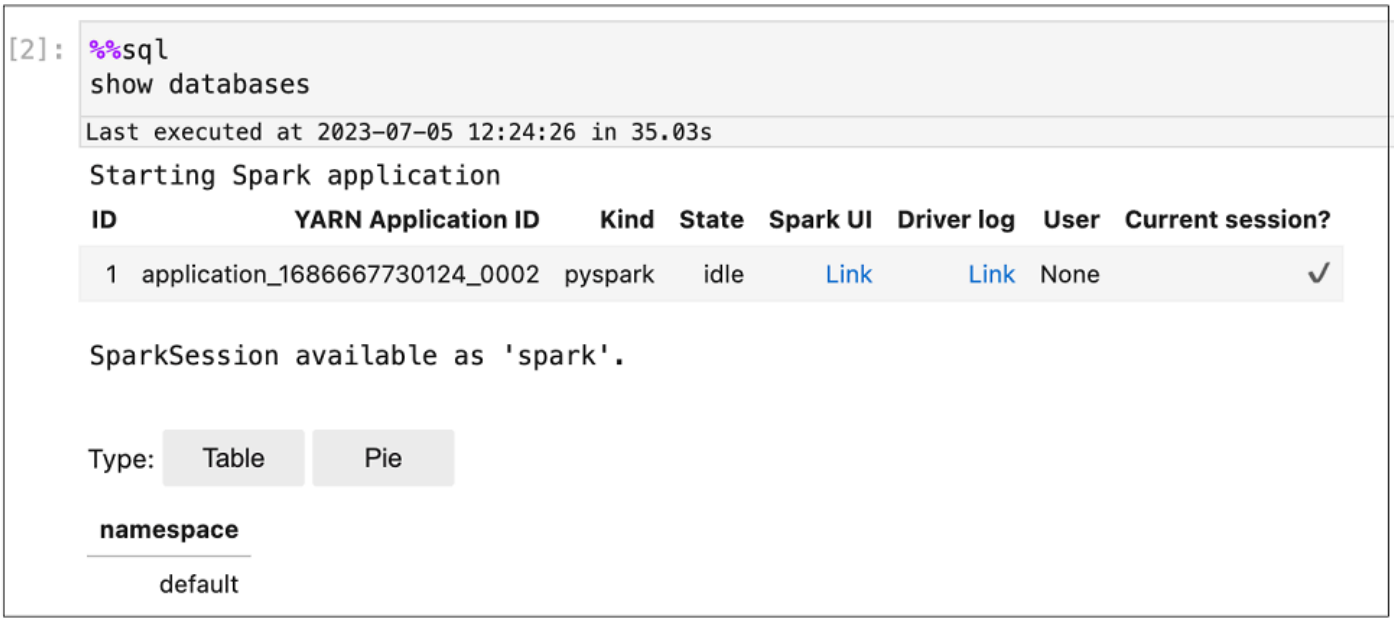
1. 在不啟用 AWS Glue Data Catalog 整合的情況下建立 Amazon EMR 叢集，也就是說，請勿選取 Hive 或 Spark 表格中繼資料的核取方塊。這是因為您將使用叢集中可用的原生 Hive 中繼存放區 (HMS) 來進行此因應措施。



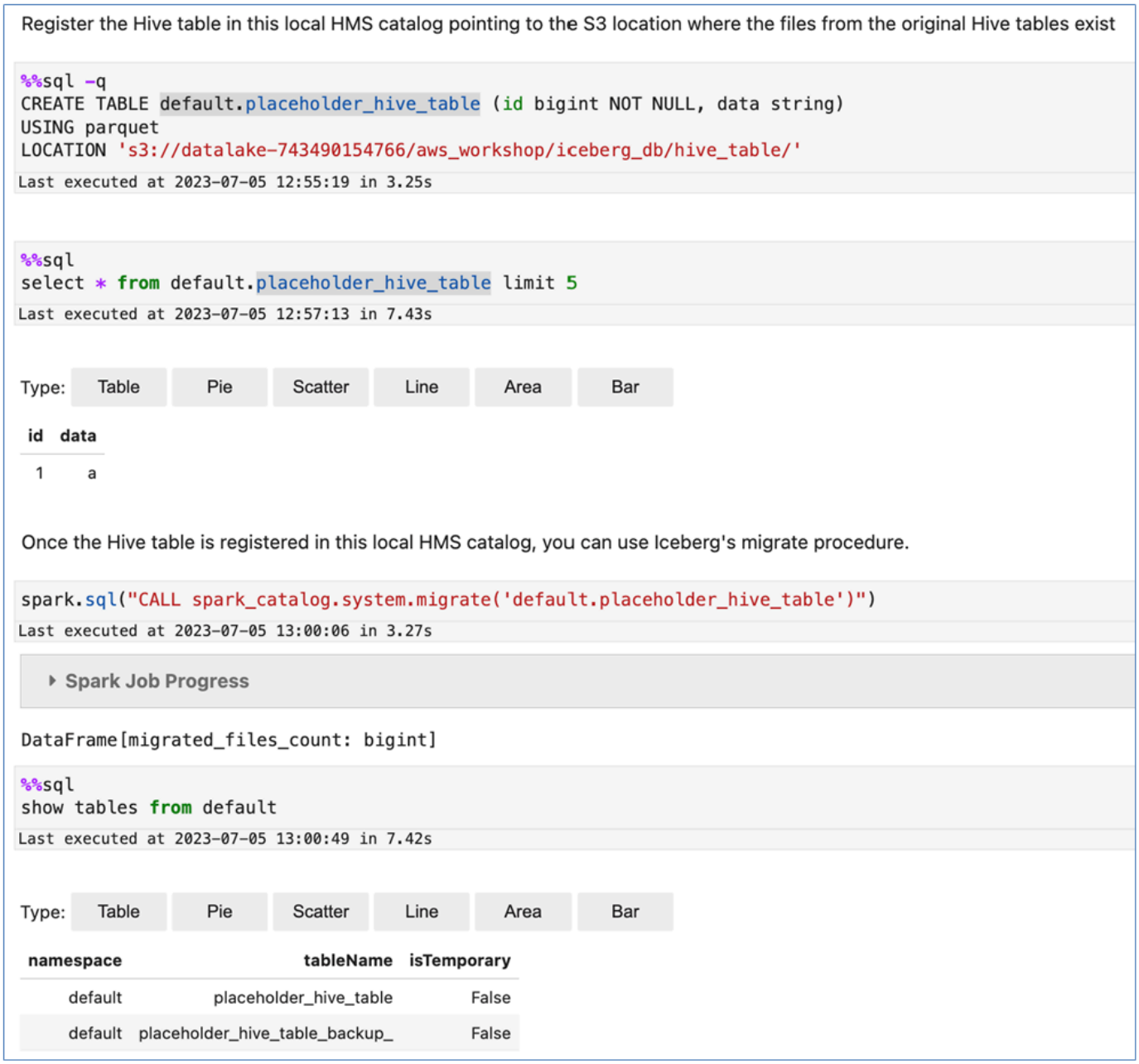
2. 設定 Spark 工作階段以使用冰山蜂巢目錄實作。

```
"spark.sql.extensions":"org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
"spark.sql.catalog.spark_catalog": "org.apache.iceberg.spark.SparkSessionCatalog",
        "spark.sql.catalog.spark_catalog.type": "hive",
```

3. AWS Glue Data Catalog 透過執行`show databases`或`show tables`，驗證您的 Amazon EMR 叢集未連線到。

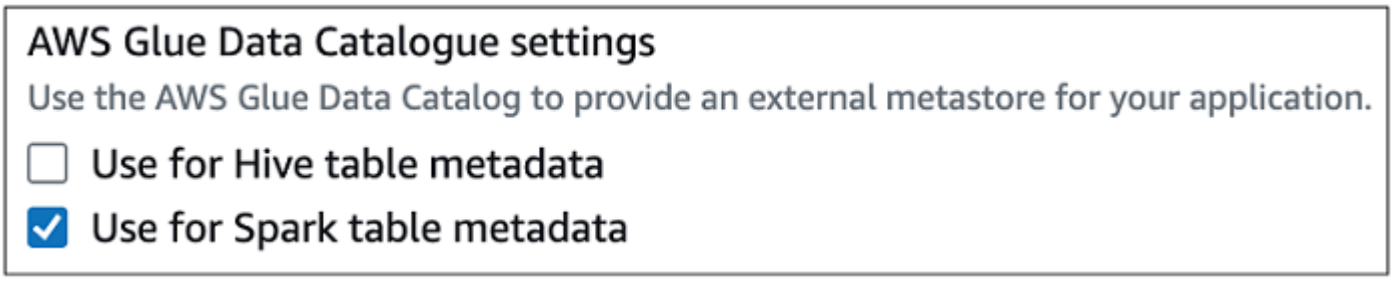


4. 在 Amazon EMR 叢集的 Hive 中繼存放區中註冊 Hive 表格，然後使用冰山`migrate`程序。



此程序會在與 Hive 資料表相同的位置建立 Iceberg 中繼資料檔案。

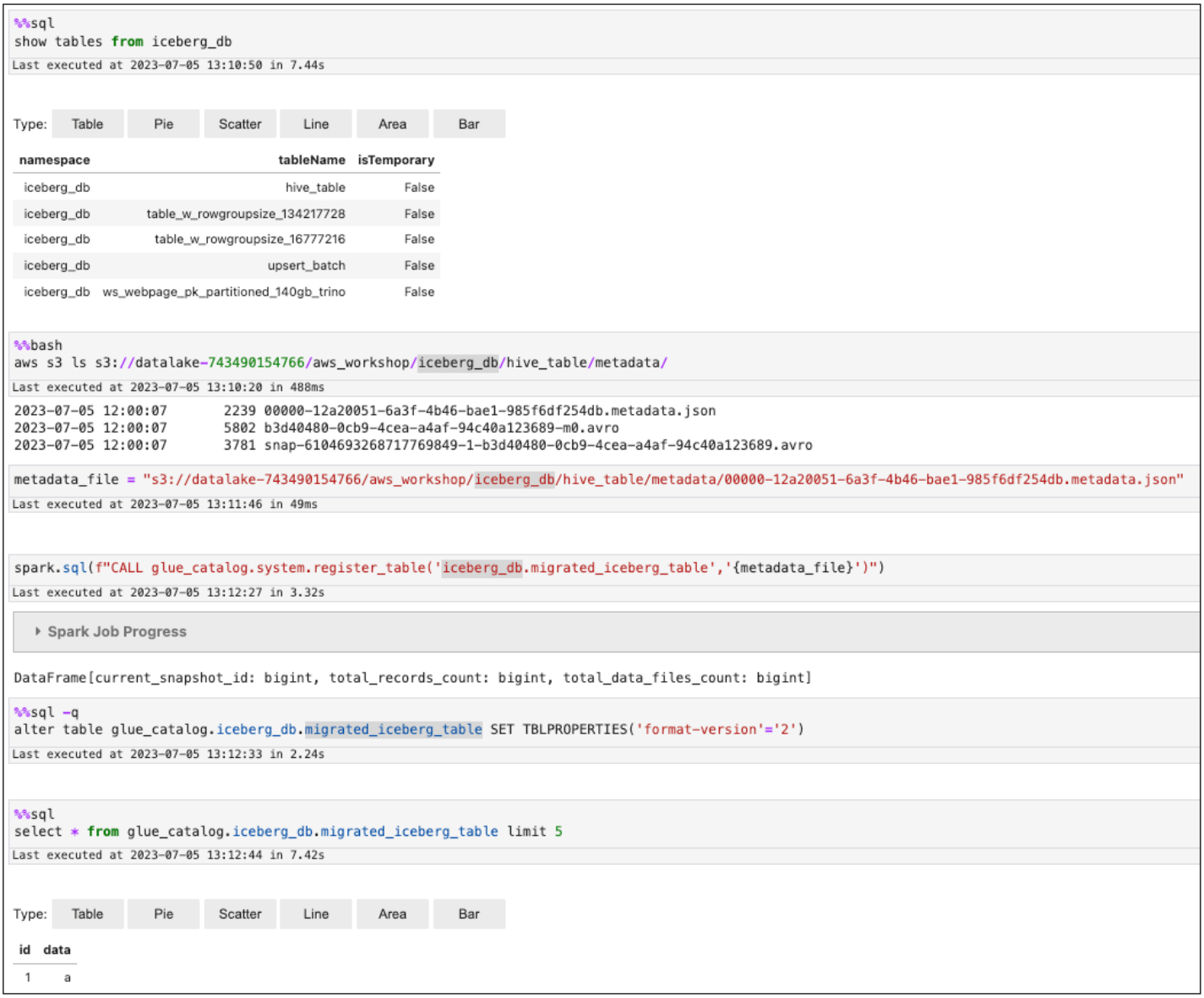
5. 在中註冊已移轉的冰山表格。 AWS Glue Data Catalog
6. 切換回已啟用 AWS Glue Data Catalog 整合的 Amazon EMR 叢集。



7. 在 Spark 會話中使用以下冰山配置。

```
"spark.sql.extensions":"org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
        "spark.sql.catalog.glue_catalog": "org.apache.iceberg.spark.SparkCatalog",
        "spark.sql.catalog.glue_catalog.warehouse": "s3://datalake-xxxx/
aws_workshop",
        "spark.sql.catalog.glue_catalog.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog",
        "spark.sql.catalog.glue_catalog.io-impl":
 "org.apache.iceberg.aws.s3.S3FileIO",
```

您現在可以從 Amazon EMR， AWS Glue或 Athena 查詢此表。



完整資料遷移

完整資料移轉會重新建立資料檔案以及中繼資料。與就地遷移相比，此方法需要更長的時間並且需要額外的計算資源 不過，此選項有助於改善表格品質：您可以驗證資料、進行結構描述和分割區變更、處理資料等等。若要實作完整資料移轉，請使用下列其中一個選項：

- 在 Amazon EMR 上的星火中使用CREATE TABLE ... AS SELECT（CTA）聲明 AWS Glue，或 Athena。 您可以使用 AND TBLPROPERTIES 子句，為新的 Iceberg 資料表設定分割區規格PARTITIONED BY和資料表屬性。您可以根據需要微調結構定義和分割新資料表，而不是直接從來源資料表繼承它們。

- 從來源資料表中讀取資料，並使用 Amazon EMR 上的 Spark 或將資料寫入為新的冰山資料表 AWS Glue (請參閱 Iceberg 文件中的建立資料表)。

選擇移轉策略

若要選擇最佳的移轉策略，請考慮下表中的問題。

問題	建議
什麼是數據文件格式（例如，CSV 或阿帕奇木地板）？	• 如果您的表格檔案格式為「實木地板」、「ORC」或「Avro」，請考慮就地移轉。 • 對於 CSV、JSON 等其他格式，請使用完整資料遷移。
是否要更新或合併資料表結構定義？	• 如果您想要使用 Iceberg 原生功能來演進資料表結構定義，請考慮就地移轉。例如，您可以在移轉後重新命名資料行。（可以在 Iceberg 元數據層中更改模式。） • 如果您要刪除資料檔案中的整個資料行，建議您使用完整的資料移轉。
資料表是否會因改變分割區策略而受益？	• 如果 Iceberg 的分割方法符合您的需求 (例如，使用新的分割區配置儲存新資料，而現有的分割區保持原樣)，請考慮就地遷移。 • 如果您想要在資料表中使用隱藏的分割區，請考慮進行完整的資料遷移。如需有關隱藏分割區的詳細資訊，請參閱最佳做法一節。
表格會受益於新增或變更排序順序策略嗎？	• 新增或變更資料的排序順序需要重寫資料集。在此情況下，請考慮使用完整資料移轉。

問題	建議
	• 對於重新寫入所有資料表分割區的大型資料表而言，請考慮使用就地移轉，並針對最常存取的磁碟分割執行壓縮 (啟用排序功能)。
表格是否有許多小檔案？	• 將小文件合併為較大的文件需要重寫數據集。在此情況下，請考慮使用完整資料移轉。
	• 對於重新寫入所有資料表分割區的大型資料表而言，請考慮使用就地移轉，並針對最常存取的磁碟分割執行壓縮 (啟用排序功能)。

最佳化 Apache 冰山工作負載的最佳做法

Iceberg 是一種表格格式，旨在簡化資料湖管理並增強工作負載效能。不同的用例可能會優先考慮不同方面的優先級，例如成本，讀取性能，寫入性能或數據保留，因此 Iceberg 提供了管理這些權衡的配置選項。本節提供有關優化和微調 Iceberg 工作負載以滿足您的需求的見解。

主題

一般最佳實務

無論您的使用案例為何，當您在使用 Apache Iceberg 時 AWS，我們建議您遵循下列一般最佳作法。

- 使用冰山格式版本 2。

  Athena 默認使用冰山格式版本 2。

  當您在 Amazon EMR 上使用 Spark 或 AWS Glue 建立冰山資料表時，請依照冰山文件中所述指定格式版本。
- 使用 AWS Glue Data Catalog 做為您的資料目錄。

  AWS Glue Data Catalog 依預設，Athena 使用。

  當您在 Amazon EMR 上使用 Spark 或 AWS Glue 與冰山合作時，請將以下組態新增至 Spark 工作階段，以使用 AWS Glue 資料型錄。如需詳細資訊，請參閱本指南前面的 AWS Glue 適用於冰山的 Spark 組態一節。

  ```
  "spark.sql.catalog.<your_catalog_name>.catalog-impl":
   "org.apache.iceberg.aws.glue.GlueCatalog"
  ```
- 使用作 AWS Glue Data Catalog 為鎖定管理員。

Athena 默認情況下使用 AWS Glue Data Catalog 作為鎖定管理器冰山表。

當您在 Amazon EMR 上使用 Spark 或 AWS Glue 與冰山一起工作時，請確保配置 Spark 會話配置以使用作 AWS Glue Data Catalog 為鎖定管理器。有關更多信息，請參閱冰山文檔中的樂觀鎖定。

- 使用標準 (ZSTD) 壓縮。

  冰山的默認壓縮編解碼器是 gzip，可以通過使用 table 屬性進行修改。`write.<file_type>.compression-codec`Athena 已經使用 ZSTD 作為冰山表的默認壓縮編解碼器。

  通常，我們建議使用 ZSTD 壓縮編解碼器，因為它可以在 GZIP 和 Snappy 之間取得平衡，並且在不影響壓縮率的情況下提供良好的讀/寫性能。此外，壓縮級別可以根據您的需求進行調整。如需詳細資訊，請參閱 Athena 文件中的 Athena 中的 ZSTD 壓縮等級。

  Snappy 可能提供最佳的整體讀取和寫入性能，但壓縮率低於 GZIP 和 ZSTD。如果您優先考慮效能 (即使這意味著在 Amazon S3 中儲存較大的資料量)，Snappy 可能是最佳選擇。

最佳化讀取效能

本節討論您可以調整以最佳化讀取效能的表格屬性，而不受引擎影響。

分割

與 Hive 表一樣，Iceberg 使用分區作為索引的主要層，以避免讀取不必要的元數據文件和數據文件。資料行統計資料也會被視為索引的次要層，以進一步改善查詢規劃，進而提高整體執行時間。

分割您的資料

若要減少查詢 Iceberg 資料表時掃描的資料量，請選擇符合預期讀取模式的平衡分割區策略：

- 識別查詢中經常使用的資料行。這些都是理想的分區候選人。例如，如果您通常查詢特定日期的資料，則分區資料行的自然範例就是日期資料行。
- 請選擇低基數分割區資料欄，以避免建立過多的分割區。分割區過多可能會增加資料表中的檔案數量，這可能會對查詢效能產生負面影響。根據經驗法則，可以將「太多分區」定義為大多數分區中的數據大小小於設置值的 2-5 倍的情況`target-file-size-bytes`。

> **Note**
>
> 如果您通常在高基數資料行上使用篩選器 (例如，可以有數千個值的`id`資料欄) 進行查詢，請使用 Iceberg 的隱藏分割功能搭配儲存貯體轉換，如下一節所述。

使用隱藏的分區

如果您的查詢通常會篩選資料表資料行的衍生項目，請使用隱藏的分割區，而不是明確建立新的資料行來當做資料分割使用。如需有關此功能的詳細資訊，請參閱 Iceberg 文件。

例如，在具有時間戳記資料行的資料集中 (例如，`2023-01-01 09:00:00`)，而不是使用剖析日期建立新資料欄 (例如，`2023-01-01`)，而是使用 partition transform 從時間戳記擷取日期部分，然後即時建立這些分區。

隱藏分區的最常見用例是：

- 當數據具有時間戳列時，按日期或時間進行分區。Iceberg 提供了多個轉換來提取時間戳的日期或時間部分。
- 當分割資料行具有高基數且會產生太多分割區時，在資料行的雜湊函數上進行分割。Iceberg 的儲存貯體轉換會使用分割資料行上的雜湊函數，將多個分割區值組合在一起，成為較少的隱藏 (儲存貯體) 分割區。

有關所有可用分區轉換的概述，請參閱 Iceberg 文檔中的分區轉換。

用於隱藏分割的資料行可以透過使用一般 SQL 函數 (例如`year()`和`month()`)，成為查詢述詞的一部分。謂詞也可以與運算符 (如`BETWEEN`和`AND`) 結合使用。

> **Note**
>
> Iceberg 無法針對產生不同資料類型的函數執行分割區修剪，`substring(event_time, 1, 10) = '2022-01-01'`例如。

使用分割區演進

當現有的分區策略不是最佳化時，請使用 Iceberg 的分區演進。例如，如果您選擇的每小時分割區太小 (每個分割區只有幾 MB)，請考慮轉換為每日或每月分割區。

當資料表的最佳分割區策略最初不清楚時，您可以使用這個方法，而且您想要在獲得更多深入見解時調整您的磁碟分割策略。分割區演進的另一個有效用途是資料磁碟區發生變化，而且目前的分割策略隨著時間的推移而變得較不

如需有關如何進化分割區的指示，請參閱冰山文件中的 ALTER TABLE SQL 延伸模組。

調整檔案大小

最佳化查詢效能需要將資料表中的小型檔案數量降到最低。為了獲得良好的查詢性能，我們通常建議將實木複合地板和 ORC 文件保持在 100 MB 以上。

檔案大小也會影響冰山資料表的查詢規劃。隨著表格中的檔案數量增加，中繼資料檔案的大小也會增加。較大的中繼資料檔案會導致較慢的查詢規劃。因此，當資料表大小增加時，請增加檔案大小以減輕中繼資料的指數擴充。

使用以下最佳實踐在 Iceberg 表中創建適當大小的文件。

設定目標檔案和資料列群組大小

Iceberg 提供了以下關鍵配置參數，用於調整數據文件佈局。建議您使用這些參數來設定目標檔案大小以及資料列群組或刪除大小。

Parameter (參數)	預設值	註解
`write.target-file-size-bytes`	512 MB	此參數指定 Iceberg 將建立的最大檔案大小。但是，某些文件的寫入大小可能小於此限制。
`write.parquet.row-group-size-bytes`	128 MB	Parquet 和 ORC 都以塊形式存儲數據，以便引擎可以避免讀取整個文件進行某些操作。
`write.orc.stripe-size-bytes`	64 MB	
`write.distribution-mode`	無，對於冰山版本 1.1 及更低版本	Iceberg 請求 Spark 在寫入存儲之前對其任務之間的數據進行排序。

Parameter (參數)	預設值	註解
	哈希，從冰山 1.2 版開始	

- 根據您預期的資料表大小，請遵循下列一般準則：
  - 小型表格 (最多幾 GB) — 將目標檔案大小縮減為 128 MB。同時將資料列群組或資料分割大小 (例如，減少到 8 或 16 MB)。
  - 中型到大型表格 (從幾 GB 到數百 GB) — 預設值是這些表格的良好起點。如果您的查詢非常有選擇性，請調整資料列群組或資料條大小 (例如，調整為 16 MB)。
  - 非常大的資料表 (數百 GB 或 TB) — 將目標檔案大小增加到 1024 MB 以上，如果查詢通常會提取大量資料集，請考慮增加資料列群組或資料帶大小。

- 若要確保寫入 Iceberg 資料表的 Spark 應用程式會建立適當大小的檔案，請將`write.distribution-mode`內容設定為`hash`或`range`。有關這些模式之間差異的詳細說明，請參閱 Iceberg 文檔中的編寫分發模式。

這些是一般指引。我們建議您執行測試，以找出最適合您特定資料表和工作負載的值。

運行定期壓實

上表中的配置設置了寫入任務可以創建的最大文件大小，但不保證文件具有該大小。為了確保正確的檔案大小，請定期執行壓縮，將小型檔案合併成較大的檔案。有關運行壓實的詳細指導，請參閱本指南後面的冰山壓實。

優化列統計

Iceberg 使用資料行統計資料來執行檔案修剪，藉由減少查詢掃描的資料量來改善查詢效能。若要受益於資料行統計資料，請確定 Iceberg 收集查詢篩選器中常用之所有資料行的統計資料。

根據預設，Iceberg 僅針對每個資料表中的前 100 個資料行收集統計資料，如 table 屬性`write.metadata.metrics.max-inferred-column-defaults`所定義。如果您的資料表有超過 100 個資料行，而您的查詢經常參考前 100 個資料行以外的資料行 (例如，您可能有篩選資料行 132) 的查詢，請確定 Iceberg 收集這些資料行的統計資料。有兩個選項可以實現這一目標：

- 建立 Iceberg 資料表時，請重新排序欄，讓查詢所需的欄落在設定的欄範圍內`write.metadata.metrics.max-inferred-column-defaults` (預設值為 100)。

注意：如果您不需要 100 個資料欄的統計資料，您可以將`write.metadata.metrics.max-inferred-column-defaults`設定調整為想要的值 (例如 20)，然後重新排序資料欄，讓您需要讀取和寫入查詢的資料欄落在資料集左側的前 20 個資料欄內。

- 如果您在查詢篩選中只使用幾個資料欄，您可以停用測量結果收集的整體特性，並選擇性地選擇要收集統計資料的個別資料欄，如下列範例所示：

```
.tableProperty("write.metadata.metrics.default", "none")
.tableProperty("write.metadata.metrics.column.my_col_a", "full")
.tableProperty("write.metadata.metrics.column.my_col_b", "full")
```

注意：在這些資料欄上排序資料時，資料欄統計資料最有效。如需詳細資訊，請參閱本指南稍後的 < 設定排序順序 > 一節。

選擇正確的更新策略

當您的使用案例可接受較慢的寫入作業時，請使用 copy-on-write 策略來最佳化讀取效能。這是冰山使用的默認策略。

C opy-on-write 導致更好的讀取效能，因為檔案會以讀取最佳化的方式直接寫入儲存空間。但是，與之相比 merge-on-read，每個寫入作業需要更長的時間，而且耗用更多的運算資源 這提出了讀取和寫入延遲之間的經典折衷。通常，對 copy-on-write 於大多數更新都在同一個表分區中並置（例如，每日批次載入）中的使用案例非常理想。

C opy-on-write 配置（`write.update.mode`、`write.delete.mode`和`write.merge.mode`）可以在表格層級設置，也可以在應用程序端獨立設置。

使用 ZSTD 壓縮

您可以使用 table 屬性`write.<file_type>.compression-codec`修改冰山使用的壓縮編解碼器。我們建議您使用 ZSTD 壓縮轉碼器來改善表格的整體效能。

默認情況下，冰山 1.3 及更早版本使用 GZIP 壓縮，與 ZSTD 相比，它提供了較慢的讀/寫性能。

附註：某些引擎可能會使用不同的預設值。這是與 Athena 或 Amazon EMR 版本 7.x 創建的冰山表的情況。

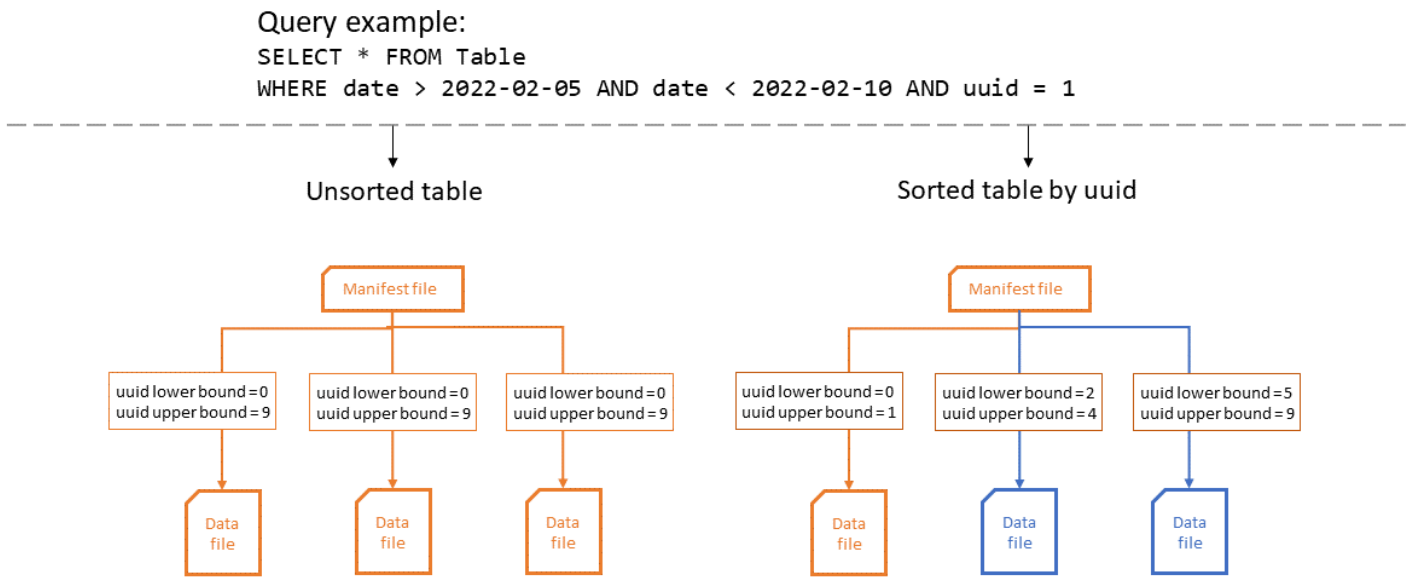
設定排序順序

若要改善 Iceberg 資料表的讀取效能，建議您根據查詢篩選器中常用的一或多個資料行來排序資料表。排序與 Iceberg 的列統計信息相結合，可以使文件修剪更有效率，從而導致更快的讀取操作。對於在查詢篩選器中使用排序欄的查詢，排序也可減少 Amazon S3 請求的數量。

您可以使用 Spark 執行資料定義語言 (DDL) 陳述式，在資料表層級設定階層排序順序。有關可用選項，請參閱 I ceberg 文檔。設定排序順序後，編寫者會將此排序套用至 Iceberg 資料表中的後續資料寫入作業。

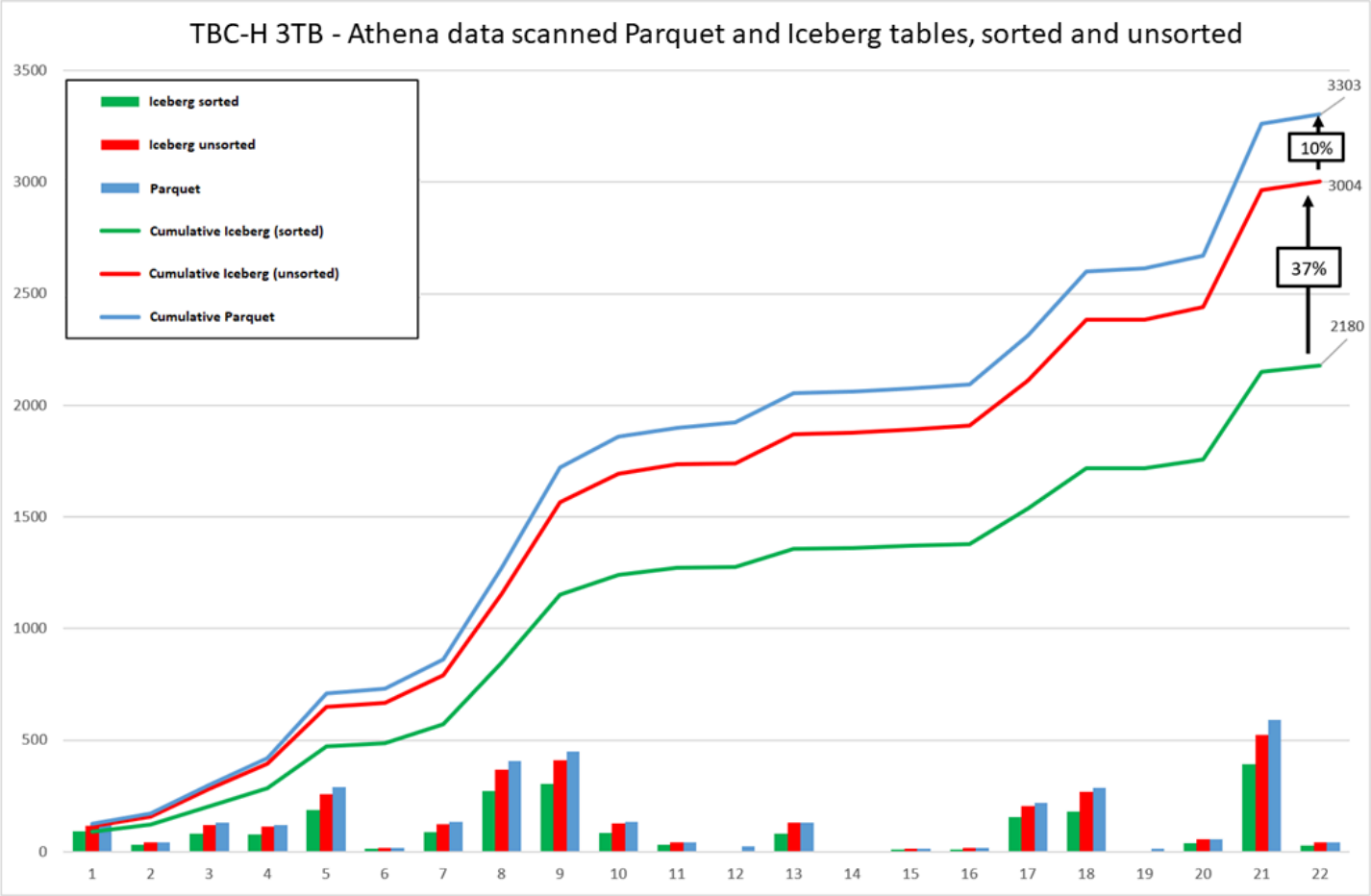
例如，在以 date (yyyy-mm-dd) 分區的資料表中，大部分查詢會篩選依據uuid，您可以使用 DDL 選項Write Distributed By Partition Locally Ordered來確定 Spark 會寫入具有非重疊範圍的檔案。

下圖說明排序表格時，資料行統計資料的效率如何提升。在示例中，排序表只需打開一個文件，並且從 Iceberg 的分區和文件中獲益最大。在未排序的表中，任何uuid可能存在於任何數據文件中，因此查詢必須打開所有數據文件。

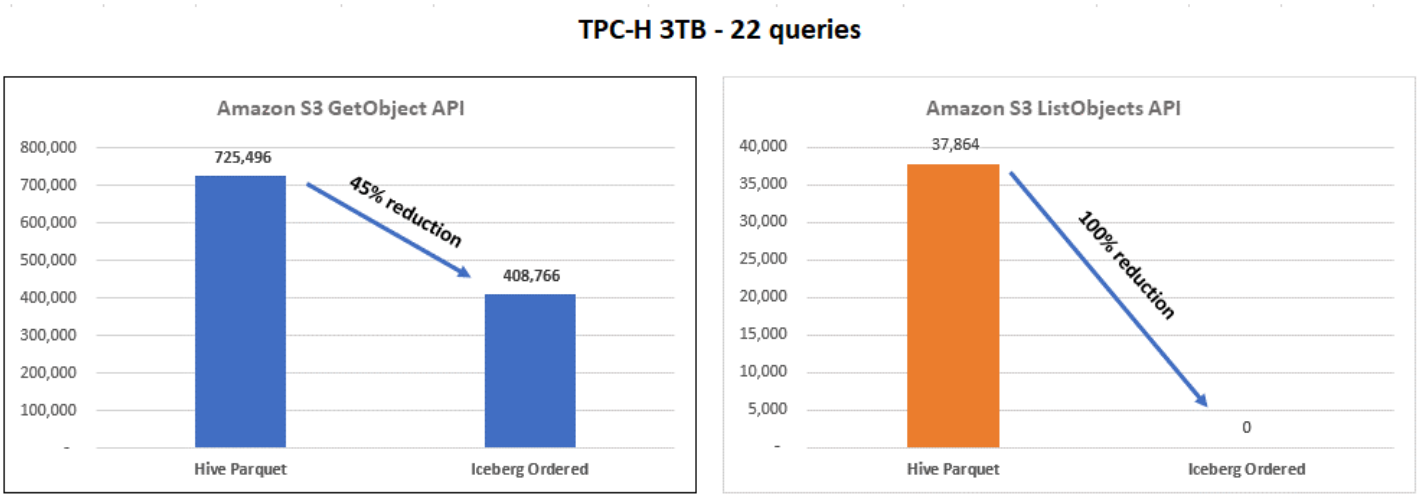


變更排序順序不會影響現有的資料檔案。您可以使用冰山壓實在那些應用排序順序。

使用 Iceberg 排序表格可能會降低工作負載的成本，如下圖所示。



這些圖表總結了運行 Hive（鑲木地板）表的 TPC-H 基準結果與冰山排序表格進行比較。不過，其他資料集或工作負載的結果可能會有所不同。



最佳化寫入效能

本節討論您可以調整這些資料表屬性，以最佳化 Iceberg 資料表上的寫入效能，而不受引擎影響。

設定資料表發佈模式

Iceberg 提供了多種寫入分配模式，用於定義寫入數據如何在 Spark 任務之間分配。如需可用模式的概觀，請參閱 Iceberg 文件中的撰寫發佈模式。

對於優先考慮寫入速度的使用案例，特別是在串流工作負載中，設`write.distribution-mode`定為`none`. 這樣可以確保冰山不要求額外的 Spark 洗牌，並在 Spark 任務中可用時寫入數據。此模式特別適用於 Spark 結構化串流應用程式。

> **Note**
>
> 將寫入分配模式設定為`none`傾向於產生許多小型檔案，這會降低讀取效能。我們建議定期壓縮，將這些小檔案合併為適當大小的檔案，以提高查詢效能。

選擇正確的更新策略

當您的使用案例可接受較慢的最新資料讀取作業時，請使用 merge-on-read策略來最佳化寫入效能。

使用時 merge-on-read，Iceberg 會將更新和刪除作為單獨的小文件寫入存儲。讀取資料表時，讀取器必須將這些變更與基底檔案合併，才能傳回資料的最新檢視。這會導致讀取作業的效能降低，但會加快更新和刪除的寫入速度。一般而言， merge-on-read 非常適合串流具有更新的工作負載，或是散佈在許多資料表分割區的更新較少的作業。

您可以在表格層級設定 merge-on-read 組態 (`write.update.mode`、`write.delete.mode`和`write.merge.mode`)，或在應用程式端獨立設定。

使用 merge-on-read 需要執行定期壓縮，以防止讀取效能隨著時間的推移而降低。壓縮功能會與現有資料檔案協調更新和刪除，以建立一組新的資料檔案，從而消除讀取端造成的效能損失。默認情況下，除非將`delete-file-threshold`屬性的默認值更改為較小的值，否則 Iceberg 的壓縮不會合併刪除文件（請參閱 Iceberg 文檔）。要了解有關壓實的更多信息，請參閱本指南後面的冰山壓實部分。

選擇正確的文件格式

冰山支持以鑲木地板，ORC 和 Avro 格式編寫數據。實木複合地板是預設格式。實木複合地板和 ORC 是單欄格式，可提供卓越的讀取性能，但通常寫入速度較慢。這代表了讀取和寫入性能之間的典型折衷。

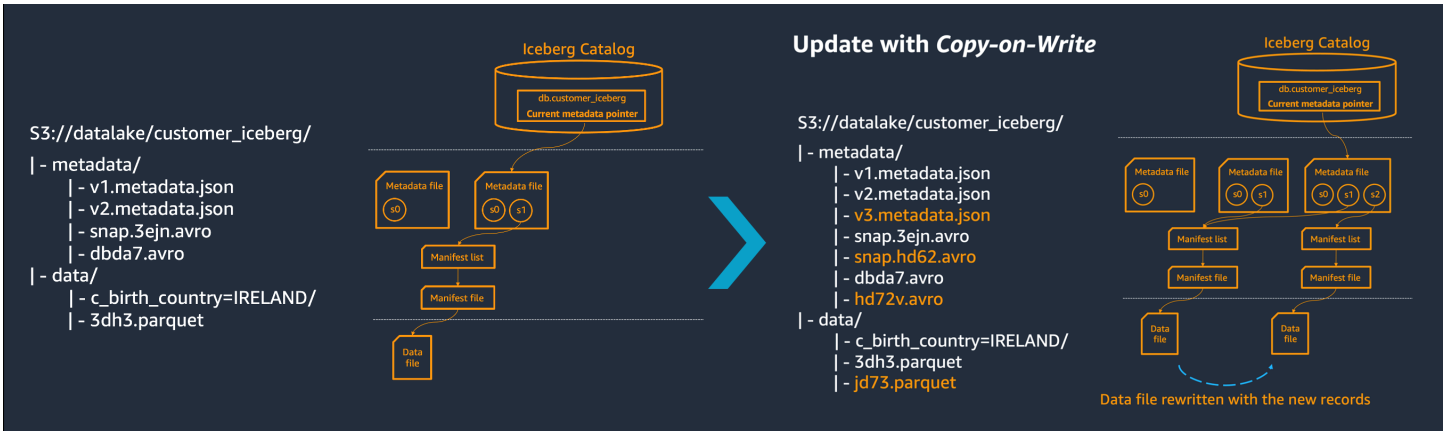
如果寫入速度對您的使用案例很重要 (例如串流工作負載)，請考慮在寫入器選項Avro中write-format將設定為，以 Avro 格式寫入。由於 Avro 是以資料列為基礎的格式，因此可提供較快的寫入時間，而且會降低讀取效能。

若要改善讀取效能，請執行定期壓縮，將小型 Avro 檔案合併並轉換為較大的 Parquet 檔案。壓實過程的結果由write.format.default表格設置控制。冰山的默認格式是實木複合地板，所以如果你寫在 Avro，然後運行壓實，冰山將 Avro 文件轉換成鑲木地板文件。範例如下：

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS glue_catalog.{DB_NAME}.{TABLE_NAME} (
        Col_1 float,
        <<<…other columns…>>
        ts timestamp)
    USING iceberg
    PARTITIONED BY (days(ts))
    OPTIONS (
      'format-version'='2',
      write.format.default'=parquet)
""")

query = df \
    .writeStream \
    .format("iceberg") \
    .option("write-format", "avro") \
    .outputMode("append") \
    .trigger(processingTime='60 seconds') \
    .option("path", f"glue_catalog.{DB_NAME}.{TABLE_NAME}") \
    .option("checkpointLocation", f"s3://{BUCKET_NAME}/checkpoints/iceberg/")

    .start()
```

最佳化儲存

更新或刪除 Iceberg 表格中的資料會增加資料副本的數量，如下圖所示。執行壓縮也是如此：它會增加 Amazon S3 中的資料複本數量。這是因為 Iceberg 將所有表底層的文件視為不可變。



請遵循本節中的最佳做法來管理儲存成本。

啟用 S3 智慧型分層

使用 Amazon S3 智慧型分層儲存類別，在存取模式變更時，自動將資料移至最具成本效益的存取層。此選項沒有營運額外負荷，也不會影響效能。

備註：請勿在 S3 智慧型分層與 Iceberg 表格中使用選用層 (例如存檔存取和深度封存存取)。若要封存資料，請參閱下一節中的準則。

您也可以使用 Amazon S3 生命週期規則來設定將物件移至另一個 Amazon S3 儲存類別的規則，例如 S3 標準 — IA 或 S3 單區域 — IA (請參閱 Amazon S3 文件中支援的轉換和相關限制)。

封存或刪除歷史快照

對於每個已提交的事務（插入，更新，合併到，壓縮）到 Iceberg 表，都會創建一個新版本或表的快照。隨著時間的推移，Amazon S3 中的版本數量和中繼資料檔案數量會累積。

對於快照隔離、資料表復原和時間行程查詢等功能，必須保留資料表的快照。不過，儲存成本會隨著您保留的版本數量而增加。

下表說明您可以實作以根據資料保留需求管理成本的設計模式。

設計模式	解決方案	使用案例
刪除舊快照	• 使用 Athena 的真空聲明刪除舊的快照。此作業不會產生任何運算成本。	這種方法可刪除不再需要的快照以降低儲存成本。您可以根據資料保留需求，設定應保留多少快照或保留多久。

設計模式	解決方案	使用案例
	• 或者，您可以使用 Amazon EMR 上的 Spark 或 AWS Glue 移除快照。如需詳細資訊，請參閱冰山文件中的過期快照。	此選項會執行快照的硬刪除。您無法復原或將時間移轉至過期的快照。
設定特定快照的保留原則	1. 使用標籤來標記特定快照，並在 Iceberg 中定義保留政策。如需詳細資訊，請參閱 Iceberg 文件中的歷史標籤。 例如，您可以在 Amazon EMR 上的 Spark 中使用下列 SQL 陳述式，每月保留一個快照，為期一年： `ALTER TABLE glue_catalog.db.table CREATE TAG 'EOM-01' AS OF VERSION 30 RETAIN 365 DAYS` 2. 在 Amazon EMR 上使用 Spark，或移 AWS Glue 除剩餘未標記的中繼快照。	此模式有助於符合要求您在過去特定時間點顯示資料表狀態的業務或法律要求。將保留原則放在特定標記的快照上，您可以移除已建立的其他 (未標記) 快照。如此一來，您就可以滿足資料保留需求，而無需保留每個建立的快照。

設計模式	解決方案	使用案例
封存舊快照	1. 使用 Amazon S3 標籤使用 Spark 標記對象。(Amazon S3 標籤與冰山標籤不同；如需詳細資訊，請參閱 Iceberg 文件。) 例如： `spark.sql.catalog.my_catalog.s3.delete-enabled=false and \ spark.sql.catalog.my_catalog.s3.delete.tags.my_key=to_archive` 2. 在 Amazon EMR 上使用星火或刪 AWS Glue 除快照。當您使用範例中的設定時，此程序會為物件加上標籤，並將其從 Iceberg 表格中繼資料分離，而不是從 Amazon S3 刪除它們。 3. 使用 S3 生命週期規則將標記為`to_archive` 其中一個 S3 Glacier 儲存類別的物件轉移。 4. 若要查詢封存的資料： • 還原封存的物件。 • 使用 Iceberg 中的註冊表程序將快照註冊為目錄中的資料表。 如需詳細指示，請參閱部 AWS 落格文章改善在 Amazon S3 資	此模式可讓您以較低的成本保留所有資料表版本和快照集。 您必須先將這些版本還原為新資料表，就無法計時移動或復原至封存的快照。這通常是可以接受的稽核目的。 您可以將此方法與先前的設計模式結合使用，為特定快照設定保留原則。

設計模式	解決方案	使用案例
	料湖上建置的 Apache Iceberg 資料表的營運效率。	

刪除孤立檔案

在某些情況下，Iceberg 應用程序可能會在您提交交易之前失敗。這將在 Amazon S3 中保留數據文件。因為沒有提交，所以這些文件不會與任何表關聯，因此您可能必須異步清理它們。

若要處理這些刪除，您可以在 Amazon Athena 使用真空陳述式。這個陳述式會移除快照，也會刪除孤立的檔案。這是非常具成本效益的，因為 Athena 不會針對此作業的運算成本收取費用。此外，您不必在使用`VACUUM`陳述式時排程任何其他作業。

或者，您可以在 Amazon EMR 上使用星火或運 AWS Glue 行該`remove_orphan_files`過程。此作業具有運算成本，且必須獨立排程。如需詳細資訊，請參閱 I ceberg 文件。

通過使用壓實維護表

Iceberg 包含的功能可讓您在將資料寫入資料表後執行資料表維護作業。某些維護作業著重於簡化中繼資料檔案，而其他維護作業則會增強資料在檔案中叢集的方式，以便查詢引擎能夠有效率地找到必要的資訊，以回應使用者要求。本節著重於與緊湊相關的最佳化。

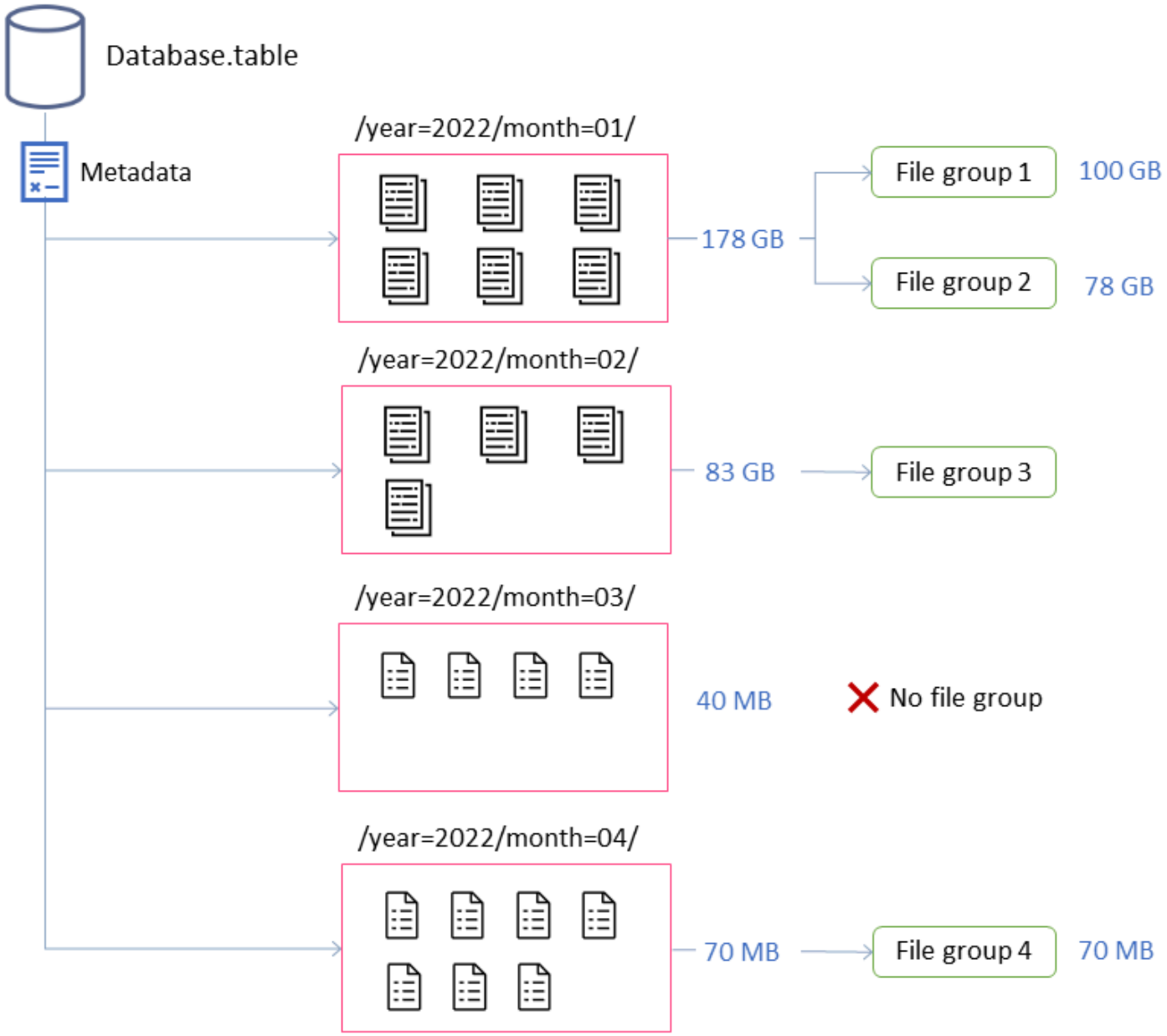
冰山壓實

在冰山，您可以使用壓實來執行四個任務：

- 將小型檔案合併為大小通常超過 100 MB 的大型檔案。這種技術被稱為垃圾箱包裝。
- 合併刪除文件與數據文件。刪除文件是由使用該 merge-on-read 方法的更新或刪除生成的。
- （重新）按照查詢模式對數據進行排序。無需任何排序順序，也可以使用適合寫入和更新的排序順序來寫入資料。
- 使用空間填充曲線對資料進行叢集，以針對不同的查詢模式進行最佳化，尤其是 z 順序排序。

上 AWS，您可以通過 Amazon 雅典娜或使用 Spark 在亞馬遜 EMR 或運行冰山表壓實和維護操作。AWS Glue

當您使用 rewrite_data_files 程序執行壓縮時，您可以調整數個旋鈕來控制壓縮行為。下圖顯示垃圾桶裝箱的預設行為。了解 bin 打包壓縮是理解分層排序和 Z 順序排序實現的關鍵，因為它們是 bin 打包接口的擴展，並以類似的方式操作。主要區別在於排序或叢集資料所需的額外步驟。



在此示例中，冰山表由四個分區組成。每個分區都有不同的大小和不同數量的文件。如果您啟動 Spark 應用程式來執行壓縮，應用程式總共會建立四個要處理的檔案群組。文件組是一個冰山抽象，表示將由單個 Spark 作業進行處理的文件集合。也就是說，運行壓縮的 Spark 應用程序將創建四個 Spark 作業來處理數據。

調整壓實行為

下列索引鍵屬性控制如何選取壓縮資料檔案：

- 依預設，最大檔案群組 (星火工作) 的資料限制設定為 100 GB。對於沒有分割區或具有跨越數百 GB 之分割區的表格而言，此屬性尤其重要。透過設定此限制，您可以細分作業以規劃工作並進行進度，同時防止叢集上的資源耗盡。

  注意：每個文件組都單獨排序。因此，如果您想要執行磁碟分割層級的排序，您必須調整此限制以符合分割區大小。
- 最小檔案大小 _ 位元組或最小檔案 _ 大小 _ 預設值是在資料表層級設定的目標檔案大小的 75%。例如，如果資料表的目標大小為 512 MB，任何小於 384 MB 的檔案都會包含在要壓縮的檔案集中。
- 最大 _ 文件大小 _ 字節或最大 _ 文件 _ 大小 _ 默認比率默認為目標文件大小的 180 百分比。如同設定最小檔案大小的兩個屬性一樣，這些屬性可用來識別壓縮工作的候選檔案。
- MIN_INPUT_FILES 指定如果資料表分割區大小小於目標檔案大小，則要壓縮的檔案數目下限。此屬性的值用於確定是否值得根據文件數量壓縮文件（默認為 5）。
- DELETE_FILE_閾值指定一個文件被包含在壓縮之前的刪除操作的最小數目。除非另有指定，否則壓縮不會將刪除文件與數據文件結合在一起。若要啟用此功能，您必須使用此屬性來設定臨界值。此臨界值特定於個別資料檔案，因此，如果您將其設定為 3，則只有在有三個或更多個參考資料檔案的刪除檔案時，才會重寫資料檔案。

這些屬性可讓您深入瞭解前一個圖表中檔案群組的形成。

例如，標示的分割區`month=01`包含兩個檔案群組，因為它超過 100 GB 的最大大小限制。相反地，`month=02`分割區包含單一檔案群組，因為它小於 100 GB。該`month=03`分區不滿足五個文件的默認最低輸入文件要求。結果，它不會被壓縮。最後，雖然`month=04`分區不包含足夠的數據來形成所需大小的單個文件，但這些文件將被壓縮，因為該分區包含五個以上的小文件。

您可以設置星火在 Amazon EMR 或 AWS Glue運行這些參數。對於 Amazon Athena，您可以使用以前綴`optimize_`開頭的表格屬性來管理類似的屬性。

運行壓實與星火在 Amazon EMR 或 AWS Glue

本節說明如何正確調整 Spark 叢集的大小以執行 Iceberg 的壓實用程式。下列範例使用 Amazon EMR 無伺服器，但您可以在 Amazon EC2 或 Amazon EKS 或中使用相同的方法。 AWS Glue

您可以利用檔案群組與 Spark 工作之間的關聯來規劃叢集資源。要按順序處理文件組，考慮到每個文件組 100 GB 的最大大小，您可以設置以下 Spark 屬性：

- spark.dynamicAllocation.enabled = FALSE
- spark.executor.memory = 20 GB
- spark.executor.instances = 5

如果您想要加速壓縮，可以增加 parallel 壓縮的檔案群組數目，以水平方式縮放。您也可以使用手動或動態擴展來擴展 Amazon EMR。

- 手動縮放 (例如，係數為 4)
  - MAX_CONCURRENT_FILE_GROUP_REWRITES= 4 (我們的因素)
  - spark.executor.instances=5 (在示例中使用的值) x4 (我們的因子) = 20
  - spark.dynamicAllocation.enabled = FALSE
- 動態縮放
  - spark.dynamicAllocation.enabled= TRUE (預設值，不需要採取任何動作)
  - 最大 _ 同步 _ 檔案 _ 群組 _ 重寫 = N (將此值與預設為 100 對齊；根據範例中的執行程式組態spark.dynamicAllocation.maxExecutors，您可以將此值設定為 20) N

  這些是協助調整叢集大小的準則。不過，您也應該監視 Spark 任務的效能，以找出適合您工作負載的最佳設定。

與 Amazon Athena 運行壓實

Athena 透過 OPTIMIZ E 陳述式將冰山壓實用程式做為受管理功能的實作提供。您可以使用這個陳述式來執行壓縮，而不必評估基礎結構。

此陳述式會使用 bin 封裝演算法，將小型檔案分組成較大的檔案，並將刪除檔案與現有資料檔案合併。若要使用分層排序或 Z 順序排序來叢集資料，請使用 Amazon EMR 上的 Spark 或. AWS Glue

您可以在陳述式中傳遞資料表屬性，或使用OPTIMIZE陳述式建立資料表之後，變更資料表建立時ALTER TABLE陳述式的預設行為。CREATE TABLE如需預設值，請參閱 A thena 文件。

執行壓實的建議

使用案例	建議
根據時間表運行垃圾箱包裝壓實	• 如果您不知道表格包含多少小檔案，請使用 Athena 中的OPTIMIZE陳述式。Athena 定價

使用案例	建議
	模式是以掃描的資料為基礎，因此如果沒有要壓縮的檔案，則這些作業不會產生相關費用。為了避免在 Athena 表上遇到超時，請 per-table-partition 基礎`OPTIMIZE`上運行。 • 如果您希望壓縮大量小檔案，請使用 Amazon EMR 或 AWS Glue 動態擴展。
根據事件執行垃圾桶包裝壓實	• 如果您希望壓縮大量小檔案，請使用 Amazon EMR 或 AWS Glue 動態擴展。
執行壓縮以排序資料	• 使用 Amazon EMR AWS Glue，或者，因為排序是一項昂貴的操作，可能需要將資料溢滿到磁碟。
運行壓實以使用 z 順序排序對數據進行分組	• 使用 Amazon EMR AWS Glue，或者，因為 Z 順序排序是非常昂貴的操作，可能需要將資料溢滿到磁碟。
在可能由於延遲到達數據而被其他應用程序更新的分區上運行壓縮	• 使用 Amazon EMR 或 AWS Glue. 啟用已啟用冰山部分進度的屬性。當您使用此選項時，Iceberg 會將壓縮輸出拆分為多個提交。如果發生衝突 (也就是說，在壓縮執行時更新資料檔案)，此設定會將重試的成本限制為包含受影響檔案的認可，以降低重試的成本。否則，您可能必須重新壓縮所有文件。

在 Amazon S3 中使用冰山工作負載

本節討論可用來優化冰山與 Amazon S3 互動的冰山屬性。

防止熱分割 (HTTP 503 錯誤)

某些在 Amazon S3 上執行的資料湖應用程式可處理數百萬或數十億個物件，並處理 PB 級的資料。這可能會導致接收大量流量的前置詞，這通常是透過 HTTP 503 (服務無法使用) 錯誤偵測到的。若要避免這個問題，請使用下列冰山屬性：

- 設定`write.distribution-mode`為`hash`或`range`以便 Iceberg 寫入大型檔案，從而減少 Amazon S3 請求。這是首選配置，應該解決大多數情況。
- 如果由於工作負載中的大量資料而繼續發生 503 錯誤，則可以在 Iceberg `true` 中設置`write.object-storage.enabled`。這會指示 Iceberg 對物件名稱進行雜湊處理，並將負載分配到多個隨機化的 Amazon S3 前綴。

如需這些屬性的詳細資訊，請參閱 Iceberg 文件中的撰寫屬性。

使用冰山維護操作來釋放未使用的數據

要管理冰山表，您可以使用冰山核心 API，冰山客戶端（例如 Spark）或 Amazon Athena 等託管服務。若要從 Amazon S3 刪除舊檔案或未使用的檔案，建議您只使用 Iceberg 原生 API 來移除快照、移除舊的中繼資料檔案，以及刪除孤立檔案。

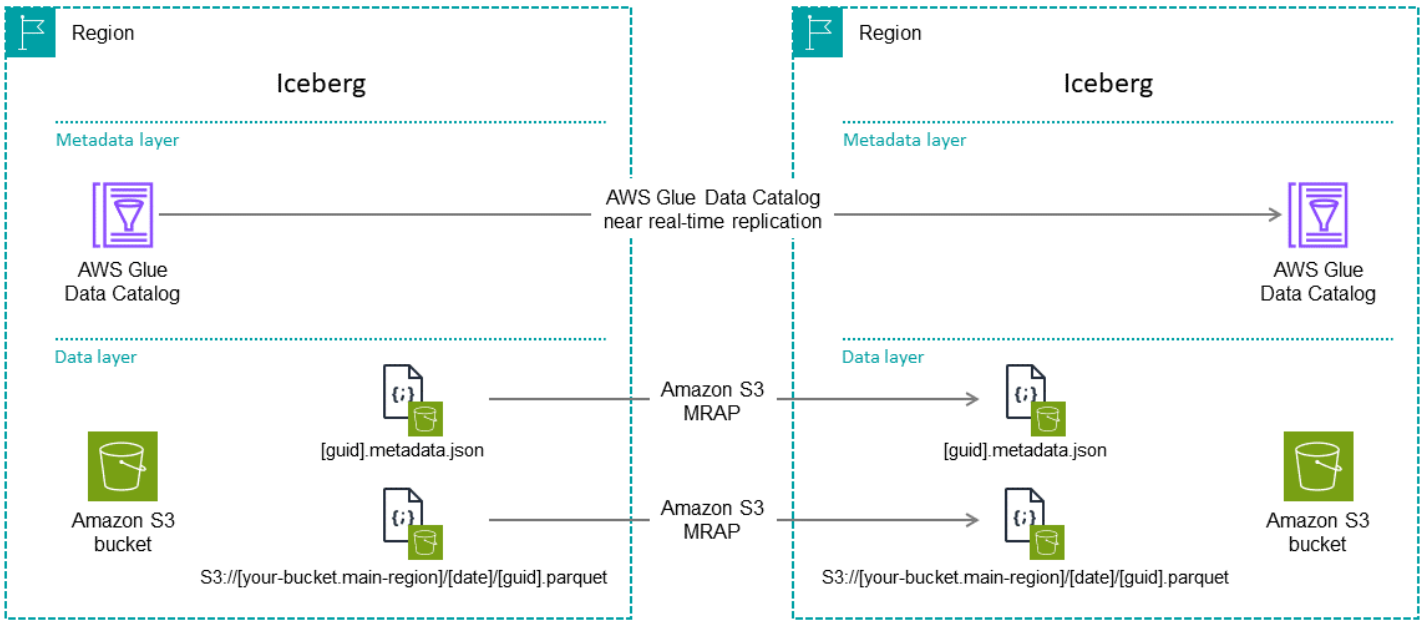
透過 Boto3、Amazon S3 開發套件或 AWS Command Line Interface (AWS CLI) 使用 Amazon S3 API，或使用任何其他非冰山方法覆寫或移除冰山資料表的 Amazon S3 檔案，會導致資料表損毀和查詢失敗。

跨資料複寫 AWS 區域

將 Iceberg 表存放在 Amazon S3 時，可以使用 Amazon S3 中的內建功能，例如跨區域複寫 (CRR) 和多區域存取點 (MRAP)，在多個 AWS 區域複寫資料。MRAP 為應用程式提供全域端點，以存取位於多個 AWS 區域儲存貯體的 S3 儲存貯體。Iceberg 不支援相對路徑，但您可以透過將儲存貯體對應到存取點，使用 MRAP 來執行 Amazon S3 操作。MRAP 也能與 Amazon S3 跨區域複寫程序無縫整合，這會造成長達 15 分鐘的延遲時間。您必須同時複製資料和中繼資料檔案。

> **Important**
>
> 目前，與 MRAP 的冰山集成僅適用於阿帕奇星火。如果您需要容錯移轉至次要 AWS 區域，則必須計劃將使用者查詢重新導向至容錯移轉區域中的 Spark SQL 環境 (例如 Amazon EMR)。

CRR 和 MRAP 功能可協助您為 Iceberg 資料表建置跨區域複寫解決方案，如下圖所示。



若要設定此跨區域複寫架構：

1. 使用 MRAP 位置建立資料表。這可確保 Iceberg 中繼資料檔案指向 MRAP 位置，而不是實體值區位置。
2. 使用 Amazon S3 MRAP 複寫冰山檔案。 MRAP 支援 15 分鐘的服務等級協定 (SLA) 進行資料複寫。Iceberg 可防止讀取操作在複製過程中引入不一致性。
3. 使表格在次要區域 AWS Glue Data Catalog 中可用。您可以從兩個選項中進行選擇：
   - 使用複寫設定管道以複製 Iceberg 表格中繼資料 AWS Glue Data Catalog 。此公用程式可在 GitHub Glue 目錄和 Lake Formation 權限複製存放庫中使用。此事件導向機制會根據事件記錄檔複寫目標區域中的資料表。
   - 當您需要容錯移轉時，請在次要區域中註冊表格。對於此選項，您可以使用先前的公用程式或 Iceberg 註冊ter_table 程序，並將其指向最新的檔案。`metadata.json`

監控阿帕奇冰山工作負載

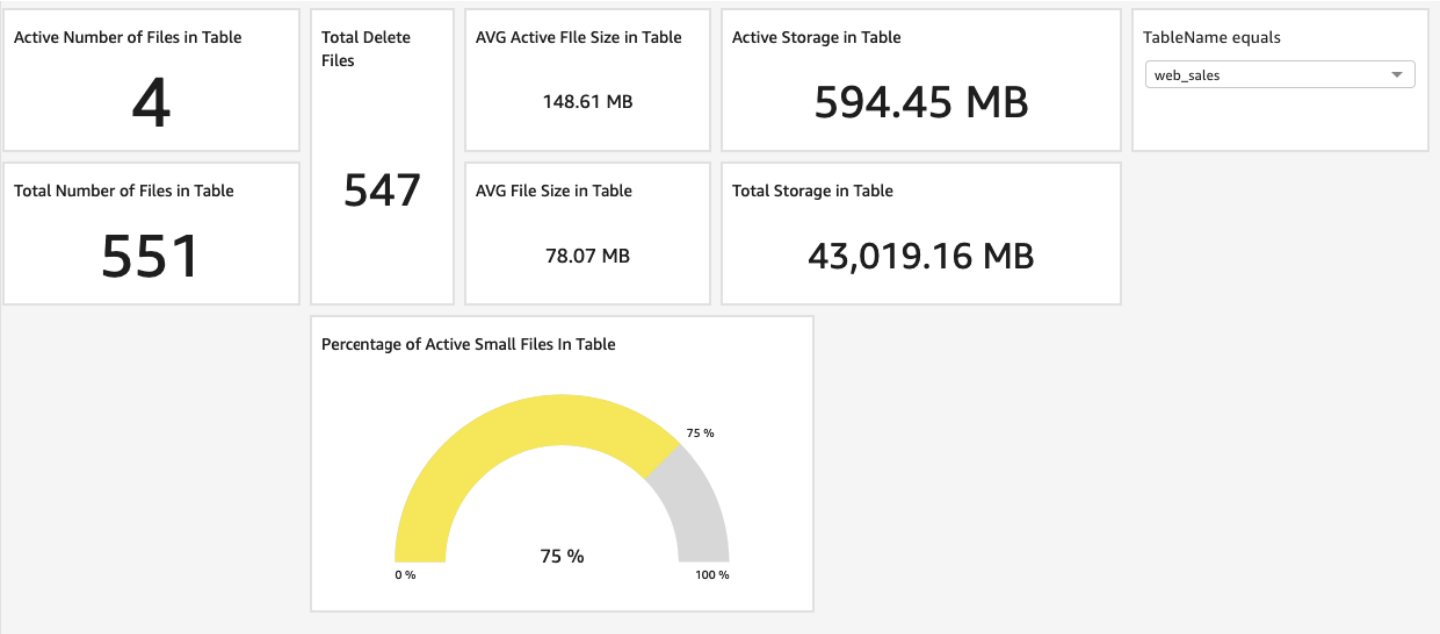
若要監控 Iceberg 工作負載，您有兩種選擇：分析中繼資料表或使用指標記者。指標記者是在冰山 1.2 版中引入的，並且僅適用於 REST 和 JDBC 目錄。

如果您正在使用 AWS Glue Data Catalog，則可以通過在 Iceberg 公開的元數據表格頂部設置監視來獲得有關 Iceberg 表的健康情況的見解。

監控對於效能管理和疑難排解至關重要。例如，當 Iceberg 資料表中的分割區達到特定百分比的小檔案時，您的工作負載可以啟動壓縮工作，將檔案合併為較大的檔案。這樣可以防止查詢放慢速度超出可接受的層級。

表格層級監控

下面的屏幕顯示了在 Amazon 創建的表監控儀表板 QuickSight。此儀表板使用 Spark SQL 查詢 Iceberg 中繼資料表，並擷取詳細的指標，例如使用中檔案數量和總儲存空間。然後，此信息存儲在 AWS Glue 表中用於操作目的。最後， QuickSight 儀表板 (如下圖所示) 是使用 Amazon Athena 建立的。此資訊可協助您識別並解決系統中的特定問題。



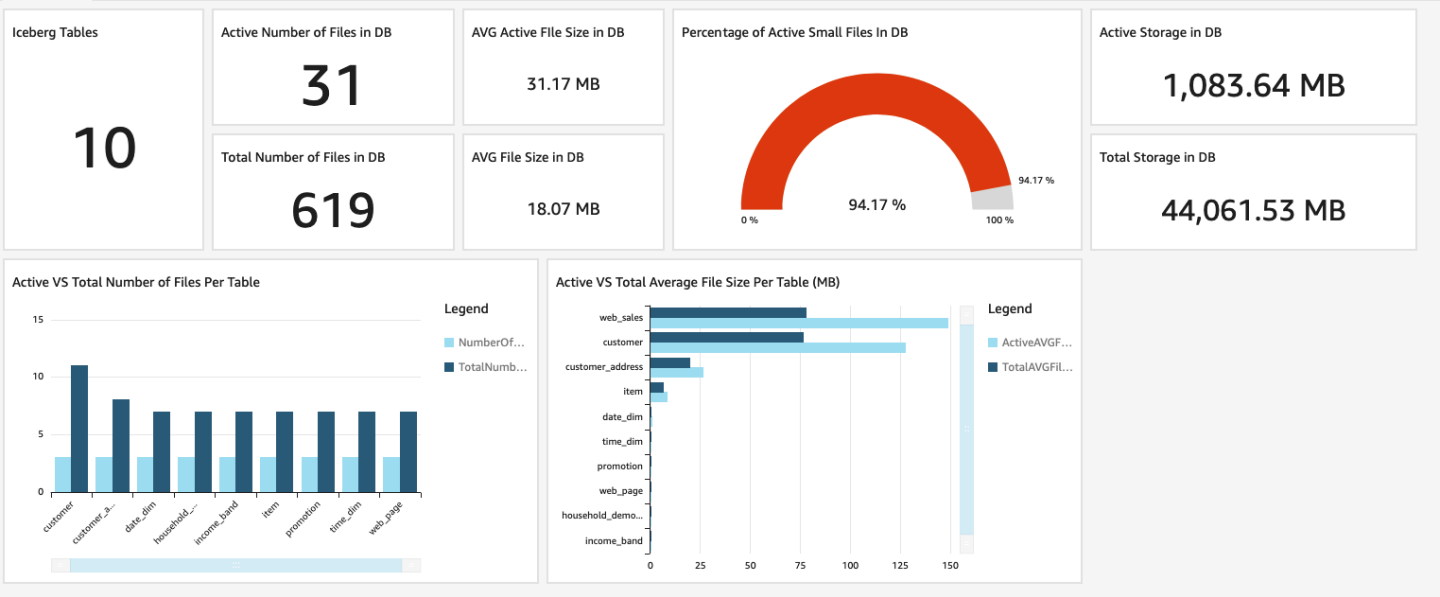
範例 QuickSight 儀表板會收集下列冰山表的關鍵績效指標 (KPI)：

KPI	Description	查詢
檔案數	冰山資料表中的檔案數目 (適用於所有快照)	`select count(*) from <catalog.database.table_name>.all_files`
作用中的檔案數	冰山資料表最後一個快照中的作用中檔案數	`select count(*) from <catalog.database.table_name>.files`
平均檔案大小	Iceberg 表格中所有檔案的平均檔案大小 (以 MB 為單位)	`select avg(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`
平均現行檔案大小	Iceberg 表格中作用中檔案的平均檔案大小 (以 MB 為單位)	`select avg(file_size_in_bytes)/1000000 from <catalog.database.table_name>.files`
小檔案的百分比	使用中檔案小於 100 MB 的百分比	`select cast(sum( case when file_size_in_bytes < 100000000 then 1 else 0 end)*100/count(*) as decimal(10,2)) from <catalog.database.table_name>.files`
儲存空間總大小	表格中所有檔案的總大小，不包括孤立檔案和 Amazon S3 物件版本 (如果已啟用)	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`

KPI	Description	查詢
作用中儲存體總大小	指定資料表目前快照集中所有檔案的總大小	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.files`

資料庫層級監視

下列範例顯示在中建立的監視儀表板， QuickSight 以提供 Iceberg 表格集合之資料庫層級 KPI 概觀。



此資料面板會收集下列 KPI：

KPI	Description	查詢
檔案數	Iceberg 資料庫中的檔案數目 (適用於所有快照)	此儀表板使用上一節提供的資料表層級查詢，並合併結果。
作用中的檔案數	Iceberg 資料庫中的作用中檔案數量 (以 Iceberg 資料表的最後一個快照為基礎)	

KPI	Description	查詢
平均檔案大小	Iceberg 資料庫中所有檔案的平均檔案大小 (以 MB 為單位)	
平均現行檔案大小	Iceberg 資料庫中所有使用中檔案的平均檔案大小 (以 MB 為單位)	
小檔案的百分比	冰山資料庫中使用中檔案小於 100 MB 的百分比	
儲存空間總大小	資料庫中所有檔案的總大小，不包括孤立檔案和 Amazon S3 物件版本 (如果已啟用)	
作用中儲存體總大小	資料庫中所有表格目前快照集中所有檔案的總大小	

預防性維護

透過設定前幾節中討論的監視功能，您可以從預防角度而非反應角度來進行資料表維護。例如，您可以使用表格層次和資料庫層次的測量結果來排定下列動作：

- 當一個表達到 N 個小文件時，使用 bin 打包壓縮來分組小文件。
- 當表達到給定分區中的 N 個刪除文件時，使用 bin 壓縮來合併刪除文件。
- 當總儲存空間比使用中儲存空間高出 X 倍時，移除快照以移除已壓縮的小型檔案。

阿帕奇冰山上的治理和訪問控制 AWS

阿帕奇冰山與整合 AWS Lake Formation 以簡化資料控管。此整合可讓資料湖管理員將儲存格層級存取權限指派給 Iceberg 表格。如需使用 Amazon Athena 查詢冰山資料表的範例 AWS Lake Formation，請參閱部 AWS 落格文章使用 Amazon Athena 與 Apache 冰山表互動，以及使用. AWS Lake Formation

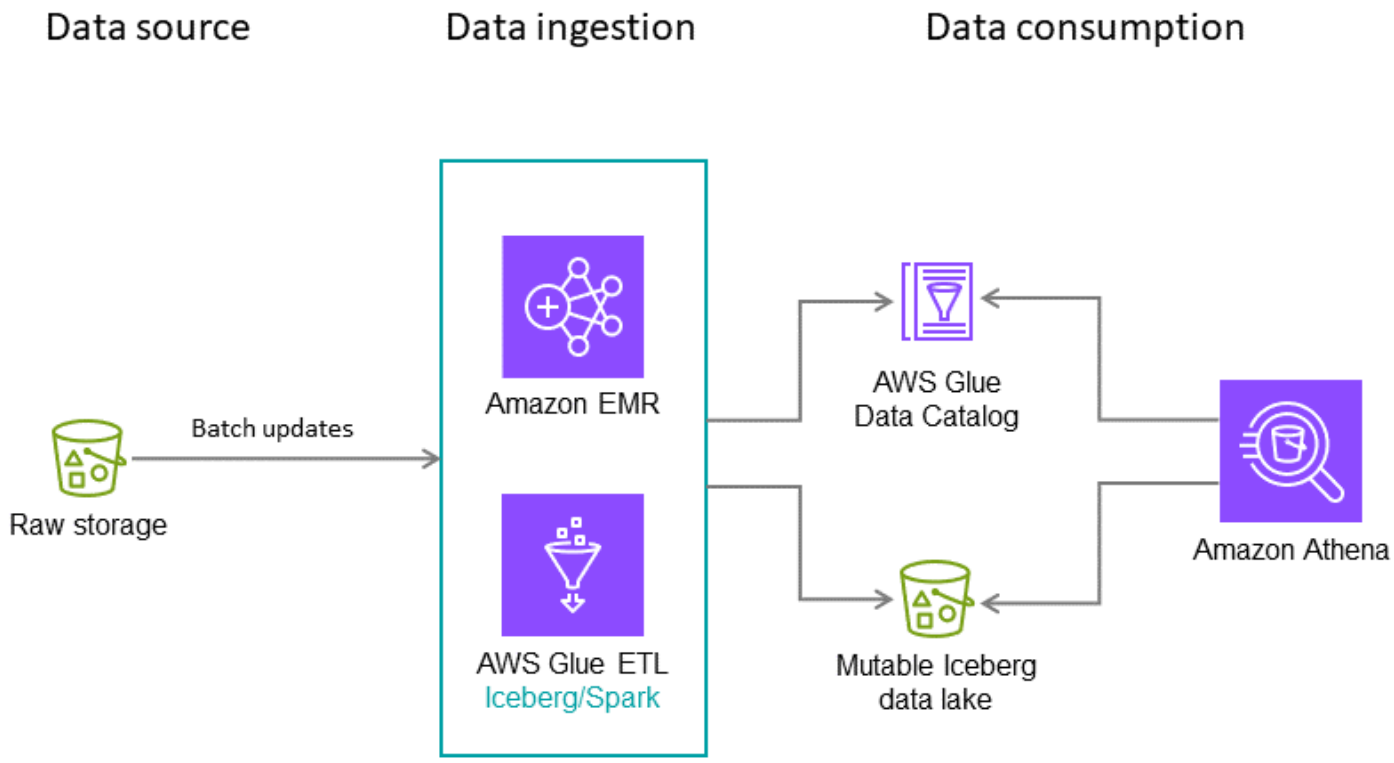
阿帕奇冰山上的參考架構 AWS

本節提供如何在不同使用案例 (例如批次擷取) 中套用最佳實務的範例，以及結合批次和串流資料擷取的資料湖。

每晚批次擷取

對於這個假設的用例，假設您的 Iceberg 表每晚會導入信用卡交易。每個批次只包含增量更新，必須將這些更新合併到目標資料表中。每年接收數次完整的歷史資料。對於這種情況，我們建議使用以下架構和配置。

注意：這只是一個例子。最佳配置取決於您的數據和需求。



建議：

- 文件大小：128 MB，因為阿帕奇星火任務處理 128 MB 塊的數據。
- 寫入類型： copy-on-write。如本指南前面所詳述，此方法有助於確保資料以讀取最佳化的方式撰寫。

- 分區變量：年/月/日。在我們假設的使用案例中，我們最頻繁地查詢最近的資料，雖然我們偶爾會針對過去兩年的資料執行完整的資料表掃描。磁碟分割的目標是根據使用案例的需求來驅動快速讀取作業。
- 排序順序：時間戳
- 資料目錄： AWS Glue Data Catalog

結合批次和近乎即時擷取的資料湖

您可以在 Amazon S3 上佈建資料湖，以跨帳戶和區域共用批次和串流資料。如需架構圖和詳細資訊，請參閱 AWS 部落格文章使用 Apache Iceberg 建立交易資料湖 AWS Glue，以及使用 AWS Lake Formation 和 Amazon Athena 的跨帳戶資料共用。

資源

- 使用中的冰山框架 AWS Glue（AWS Glue 文檔）
- 冰山（Amazon EMR 文檔）
- 使用阿帕奇冰山表（Amazon Athena 文檔）
- Amazon S3 文件
- Amazon QuickSight 文檔
- Glue 目錄和 Lake Formation 權限複製（GitHub 存儲庫）
- 阿帕奇冰山文檔
- 阿帕奇星火文檔

貢獻者

以下人員在 AWS 撰寫、共同撰寫和審閱本指南。

貢獻者

- 卡洛斯·羅德里格斯, 解決方案架構師, 大
- Imtiaz（Taz）Sayd，解決方案架構師技術領導者，分析
- 夏娜施珀斯，解決方案架構師，大數據
- 普拉珊特·辛格，軟件開發工程師，Amazon EMR
- 斯蒂法諾·桑多納，解決方案架構師，大數據
- 阿倫 A K，解決方案架構師，大數據和 ETL
- 弗朗西斯科·莫里略，解決方案架構師
- 蘇森·菲利普斯，分析架構師，Amazon EMR
- 卡拉 OK 系列, 解決方案架構師
- 與那坦·多蘭, 分析專家
- 蓋·巴查爾，解決方案架構師
- 索菲亞·齊伯曼, 解決方案架構師, 流
- 伊斯梅爾·馬赫魯夫，解決方案架構師，分析
- 丹樓梯, 專業解決方案建築師
- 薩克蒂米什拉, 解決方案架構師

審稿

- 里克·西爾斯, 總經理, Amazon EMR
- 琳達·奧康納, Amazon EMR 專家
- 伊恩·邁耶斯, 主任, Amazon EMR
- 維妮塔·安南斯，Amazon EMR 產品管理總監
- 傑森·伯科維茨，產品經理， AWS Lake Formation
- 馬赫什米什拉, 產品經理, Amazon Redshift
- 弗拉基米爾·茲拉特金, 經理, 解決方案架構, 大數據
- 卡蒂克·普拉巴卡爾，分析建築師，Amazon EMR

- Amazon EMR 軟體開發工程師葉傑
- 維傑 · 耆那教, 產品經理
- 評論, Amazon S3 產品經理
- 莫莉·布朗, 總經理, AWS Lake Formation
- 設計師, 解決方案架構師, 數據
- 陳格文 , 產品行銷經理

文件歷史紀錄

下表描述了本指南的重大變更。如果您想收到有關未來更新的通知，可以訂閱 RSS 摘要。

變更	描述	日期
初次出版	—	2024 年 4 月 30 日

AWS 規定指引詞彙

以下是 AWS 規範性指引所提供的策略、指南和模式中常用的術語。若要建議項目，請使用詞彙表末尾的提供意見回饋連結。

數字

7 R

將應用程式移至雲端的七種常見遷移策略。這些策略以 Gartner 在 2011 年確定的 5 R 為基礎，包括以下內容：

- 重構/重新架構 – 充分利用雲端原生功能來移動應用程式並修改其架構，以提高敏捷性、效能和可擴展性。這通常涉及移植作業系統和資料庫。範例：將您的現場部署 Oracle 資料庫遷移到與 Amazon Aurora PostgreSQL 相容的版本。
- 平台轉換 (隨即重塑) – 將應用程式移至雲端，並引入一定程度的優化以利用雲端功能。範例：將您的現場部署 Oracle 資料庫遷移到 Amazon Relational Database Service 服務 (Amazon RDS)，適用於 AWS 雲端.
- 重新購買 (捨棄再購買) – 切換至不同的產品，通常從傳統授權移至 SaaS 模型。範例：將您的客戶關係管理 (CRM) 系統遷移至 SalesForce.com。
- 主機轉換 (隨即轉移) – 將應用程式移至雲端，而不進行任何變更以利用雲端功能。範例：將您的現場部署 Oracle 資料庫遷移至中 EC2 執行個體上的 Oracle 資料庫 AWS 雲端。
- 重新放置 (虛擬機器監視器等級隨即轉移) – 將基礎設施移至雲端，無需購買新硬體、重寫應用程式或修改現有操作。您可以將伺服器從內部部署平台遷移到相同平台的雲端服務。範例：將 Microsoft Hyper-V應用程式移轉至 AWS.
- 保留 (重新檢視) – 將應用程式保留在來源環境中。其中可能包括需要重要重構的應用程式，且您希望將該工作延遲到以後，以及您想要保留的舊版應用程式，因為沒有業務理由來進行遷移。
- 淘汰 – 解除委任或移除來源環境中不再需要的應用程式。

A

ABAC

請參閱以屬性為基礎的存取控制。

抽象的服務

請參閱受管理服務。

酸

請參閱原子性、一致性、隔離性、耐用性。

主動-主動式遷移

一種資料庫遷移方法，其中來源和目標資料庫保持同步 (透過使用雙向複寫工具或雙重寫入操作)，且兩個資料庫都在遷移期間處理來自連接應用程式的交易。此方法支援小型、受控制批次的遷移，而不需要一次性切換。它比主動-被動遷移更具彈性，但需要更多的工作。

主動-被動式遷移

一種資料庫遷移方法，其中來源和目標資料庫保持同步，但只有來源資料庫處理來自連接應用程式的交易，同時將資料複寫至目標資料庫。目標資料庫在遷移期間不接受任何交易。

聚合函數

在一組資料列上運作，並計算群組的單一傳回值的 SQL 函數。彙總函式的範例包括SUM和MAX。

AI

請參閱人工智慧。

艾奧運

請參閱人工智慧作業。

匿名化

永久刪除資料集中個人資訊的程序。匿名化可以幫助保護個人隱私。匿名資料不再被視為個人資料。

反模式

一種經常使用的解決方案，用於解決方案的生產力適得其反，效果不佳或效果低於替代方案。

應用控制

一種安全性方法，只允許使用核准的應用程式，以協助保護系統免受惡意軟體的攻擊。

應用程式組合

有關組織使用的每個應用程式的詳細資訊的集合，包括建置和維護應用程式的成本及其商業價值。此資訊是產品組合探索和分析程序的關鍵，有助於識別要遷移、現代化和優化的應用程式並排定其優先順序。

人工智慧 (AI)

電腦科學領域，致力於使用運算技術來執行通常與人類相關的認知功能，例如學習、解決問題和識別模式。如需詳細資訊，請參閱什麼是人工智慧？

人工智慧操作 (AIOps)

使用機器學習技術解決操作問題、減少操作事件和人工干預以及提高服務品質的程序。如需有關如何在 AWS 遷移策略中使用 AIOps 的詳細資訊，請參閱操作整合指南。

非對稱加密

一種加密演算法，它使用一對金鑰：一個用於加密的公有金鑰和一個用於解密的私有金鑰。您可以共用公有金鑰，因為它不用於解密，但對私有金鑰存取應受到高度限制。

原子性、一致性、隔離性、耐久性 (ACID)

一組軟體屬性，即使在出現錯誤、電源故障或其他問題的情況下，也能確保資料庫的資料有效性和操作可靠性。

屬性型存取控制 (ABAC)

根據使用者屬性 (例如部門、工作職責和團隊名稱) 建立精細許可的實務。如需詳細資訊，請參閱 AWS Identity and Access Management (IAM) 文件 AWS中的 ABAC。

授權資料來源

儲存資料主要版本的位置，被認為是最可靠的資訊來源。您可以將授權資料來源中的資料複製到其他位置，以便處理或修改資料，例如匿名化、編輯或將其虛擬化。

可用區域

一個獨立的位置，與其他 AWS 區域 可用區域中的故障隔離，並為相同區域中的其他可用區域提供廉價、低延遲的網路連線能力。

AWS 雲端採用架構 (AWS CAF)

指導方針和最佳做法的架構，可協 AWS 助組織制定有效率且有效的計畫，以順利移轉至雲端。AWS CAF 將指導組織到六個重點領域，稱為觀點：業務，人員，治理，平台，安全性和運營。業務、人員和控管層面著重於業務技能和程序；平台、安全和操作層面著重於技術技能和程序。例如，人員層面針對處理人力資源 (HR)、人員配備功能和人員管理的利害關係人。針對此觀點，AWS CAF 為人員開發、訓練和通訊提供指導，以協助組織為成功採用雲端做好準備。如需詳細資訊，請參閱 AWS CAF 網站和 AWS CAF 白皮書。

AWS 工作負載資格架構 (AWS WQF)

可評估資料庫移轉工作負載、建議移轉策略並提供工作預估的工具。 AWS WQF 包含在 AWS Schema Conversion Tool ()AWS SCT中。它會分析資料庫結構描述和程式碼物件、應用程式程式碼、相依性和效能特性，並提供評估報告。

B

壞機器人

旨在破壞或對個人或組織造成傷害的機器人。

BCP

請參閱業務連續性規劃。

行為圖

資源行為的統一互動式檢視，以及一段時間後的互動。您可以將行為圖與 Amazon Detective 搭配使用來檢查失敗的登入嘗試、可疑的 API 呼叫和類似動作。如需詳細資訊，請參閱偵測文件中的行為圖中的資料。

大端序系統

首先儲存最高有效位元組的系統。另請參閱 「位元順序」。

二進制分類

預測二進制結果的過程 (兩個可能的類別之一)。例如，ML 模型可能需要預測諸如「此電子郵件是否是垃圾郵件？」等問題 或「產品是書還是汽車？」

Bloom 篩選條件

一種機率性、記憶體高效的資料結構，用於測試元素是否為集的成員。

藍/綠部署

建立兩個獨立但相同環境的部署策略。您可以在一個環境中執行目前的應用程式版本 (藍色)，而在另一個環境 (綠色) 中執行新的應用程式版本。此策略可協助您以最小的影響快速回復。

機器人

透過網際網路執行自動化工作並模擬人類活動或互動的軟體應用程式。某些漫遊器是有用的或有益的，例如用於索引 Internet 上信息的網絡爬蟲。其他一些機器人（稱為不良機器人）旨在破壞或對個人或組織造成傷害。

殭屍網絡

受惡意軟件感染並受到單一方（稱為機器人牧民或機器人操作員）控制的機器人網絡。殭屍網絡是擴展機器人及其影響的最著名機制。

分支

程式碼儲存庫包含的區域。儲存庫中建立的第一個分支是主要分支。您可以從現有分支建立新分支，然後在新分支中開發功能或修正錯誤。您建立用來建立功能的分支通常稱為功能分支。當準備好發佈功能時，可以將功能分支合併回主要分支。如需詳細資訊，請參閱關於分支 (GitHub 文件)。

防碎玻璃訪問

在特殊情況下，並透過核准的程序，使用者可以快速取得他 AWS 帳戶 們通常沒有存取權限的存取權。如需詳細資訊，請參閱 AWS Well-Architected 指南中的實作防破玻璃程序指標。

棕地策略

環境中的現有基礎設施。對系統架構採用棕地策略時，可以根據目前系統和基礎設施的限制來設計架構。如果正在擴展現有基礎設施，則可能會混合棕地和綠地策略。

緩衝快取

儲存最常存取資料的記憶體區域。

業務能力

業務如何創造價值 (例如，銷售、客戶服務或營銷)。業務能力可驅動微服務架構和開發決策。如需詳細資訊，請參閱在 AWS上執行容器化微服務白皮書的圍繞業務能力進行組織部分。

業務連續性規劃 (BCP)

一種解決破壞性事件 (如大規模遷移) 對營運的潛在影響並使業務能夠快速恢復營運的計畫。

C

咖啡

請參閱AWS 雲端採用架構。

金絲雀部署

向最終用戶發行版本的緩慢和增量版本。當您有信心時，您可以部署新版本並完全取代目前的版本。

CCoE

請參閱雲端卓越中心。

CDC

請參閱變更資料擷取。

變更資料擷取 (CDC)

追蹤對資料來源 (例如資料庫表格) 的變更並記錄有關變更的中繼資料的程序。您可以將 CDC 用於各種用途，例如稽核或複寫目標系統中的變更以保持同步。

混沌工程

故意引入故障或破壞性事件來測試系統的彈性。您可以使用 AWS Fault Injection Service (AWS FIS) 執行實驗來 stress 您的 AWS 工作負載並評估其回應。

CI/CD

請參閱持續整合和持續交付。

分類

有助於產生預測的分類程序。用於分類問題的 ML 模型可預測離散值。離散值永遠彼此不同。例如，模型可能需要評估影像中是否有汽車。

用戶端加密

在目標 AWS 服務 接收資料之前，在本機加密資料。

雲端卓越中心 (CCoE)

一個多學科團隊，可推動整個組織的雲端採用工作，包括開發雲端最佳實務、調動資源、制定遷移時間表以及領導組織進行大規模轉型。如需詳細資訊，請參閱 AWS 雲端 企業策略部落格上的 CCoE 文章。

雲端運算

通常用於遠端資料儲存和 IoT 裝置管理的雲端技術。雲計算通常連接到邊緣計算技術。

雲端運作模式

在 IT 組織中，這是用來建置、成熟和最佳化一或多個雲端環境的作業模型。如需詳細資訊，請參閱建立您的雲端作業模型。

採用雲端階段

組織移轉至下列四個階段時通常會經歷 AWS 雲端：

- 專案 – 執行一些與雲端相關的專案以進行概念驗證和學習用途
- 基礎 – 進行基礎投資以擴展雲端採用 (例如，建立登陸區域、定義 CCoE、建立營運模型)
- 遷移 – 遷移個別應用程式
- 重塑 – 優化產品和服務，並在雲端中創新

這些階段是 Stephen Orban 在 AWS 雲端 企業策略部落格部落格文章「邁向雲端優先的旅程與採用階段」中所定義的。如需其與 AWS 移轉策略之間關聯的詳細資訊，請參閱移轉準備指南。

CMDB

請參閱組態管理資料庫。

程式碼儲存庫

透過版本控制程序來儲存及更新原始程式碼和其他資產 (例如文件、範例和指令碼) 的位置。常見的雲存儲庫包括 GitHub 或 AWS CodeCommit。程式碼的每個版本都稱為分支。在微服務結構中，每個儲存庫都專用於單個功能。單一 CI/CD 管道可以使用多個儲存庫。

冷快取

一種緩衝快取，它是空的、未填充的，或者包含過時或不相關的資料。這會影響效能，因為資料庫執行個體必須從主記憶體或磁碟讀取，這比從緩衝快取讀取更慢。

冷資料

很少存取且通常是歷史資料。查詢此類資料時，通常可以接受緩慢的查詢。將此資料移至效能較低且成本較低的儲存層或類別可降低成本。

計算機視覺（CV）

一個 AI 領域，它使用機器學習來分析和從數字圖像和視頻等視覺格式中提取信息。例如，提 AWS Panorama 供將 CV 添加到現場部署攝像機網絡的設備，Amazon 為 CV SageMaker 提供圖像處理算法。

配置漂移

對於工作負載，組態會從預期的狀態變更。這可能會導致工作負載變得不合規，而且通常是漸進且無意的。

組態管理資料庫 (CMDB)

儲存和管理有關資料庫及其 IT 環境的資訊的儲存庫，同時包括硬體和軟體元件及其組態。您通常在遷移的產品組合探索和分析階段使用 CMDB 中的資料。

一致性套件

AWS Config 規則和補救動作的集合，您可以組合這些動作來自訂合規性和安全性檢查。您可以使用 YAML 範本，將一致性套件部署為 AWS 帳戶 和區域中的單一實體，或跨組織部署。如需詳細資訊，請參閱文件中的AWS Config 一致性套件。

持續整合和持續交付 (CI/CD)

自動化軟體發行程序的來源、建置、測試、暫存和生產階段的程序。CI/CD 通常被描述為管道。CI/CD 可協助您將程序自動化、提升生產力、改善程式碼品質以及加快交付速度。如需詳細資訊，請參閱持續交付的優點。CD 也可表示持續部署。如需詳細資訊，請參閱持續交付與持續部署。

CV

請參閱電腦視覺。

D

靜態資料

網路中靜止的資料，例如儲存中的資料。

資料分類

根據重要性和敏感性來識別和分類網路資料的程序。它是所有網路安全風險管理策略的關鍵組成部分，因為它可以協助您確定適當的資料保護和保留控制。資料分類是 AWS Well-Architected 架構中安全性支柱的一個組成部分。如需詳細資訊，請參閱資料分類。

資料漂移

生產資料與用來訓練 ML 模型的資料之間有意義的變化，或輸入資料隨著時間的推移有意義的變化。資料漂移可降低機器學習模型預測中的整體品質、準確性和公平性。

傳輸中的資料

在您的網路中主動移動的資料，例如在網路資源之間移動。

資料網格

透過集中式管理和控管，提供分散式、分散式資料擁有權的架構架構。

資料最小化

僅收集和處理絕對必要的數據的原則。在中執行資料最小化 AWS 雲端 可降低隱私權風險、成本和分析碳足跡。

資料周長

您 AWS 環境中的一組預防性護欄，可協助確保只有受信任的身分正在存取來自預期網路的受信任資源。若要取得更多資訊，請參閱〈在上建置資料周長〉 AWS。

資料預先處理

將原始資料轉換成 ML 模型可輕鬆剖析的格式。預處理資料可能意味著移除某些欄或列，並解決遺失、不一致或重複的值。

數據來源

在整個生命週期中追蹤資料來源和歷史記錄的程序，例如資料的產生、傳輸和儲存方式。

資料主體

正在收集和處理資料的個人。

資料倉儲

支援商業智慧 (例如分析) 的資料管理系統。資料倉儲通常包含大量歷史資料，通常用於查詢和分析。

資料庫定義語言 (DDL)

用於建立或修改資料庫中資料表和物件之結構的陳述式或命令。

資料庫處理語言 (DML)

用於修改 (插入、更新和刪除) 資料庫中資訊的陳述式或命令。

DDL

請參閱資料庫定義語言。

深度整體

結合多個深度學習模型進行預測。可以使用深度整體來獲得更準確的預測或估計預測中的不確定性。

深度學習

一個機器學習子領域，它使用多層人工神經網路來識別感興趣的輸入資料與目標變數之間的對應關係。

defense-in-depth

這是一種資訊安全方法，其中一系列的安全機制和控制項會在整個電腦網路中精心分層，以保護網路和其中資料的機密性、完整性和可用性。在上採用此策略時 AWS，您可以在 AWS

Organizations 結構的不同層加入多個控制項，以協助保護資源。例如，一 defense-in-depth 種方法可能會結合多因素驗證、網路分段和加密。

委派的管理員

在中 AWS Organizations，相容的服務可以註冊成 AWS 員帳戶，以管理組織的帳戶並管理該服務的權限。此帳戶稱為該服務的委派管理員。如需詳細資訊和相容服務清單，請參閱 AWS Organizations 文件中的可搭配 AWS Organizations運作的服務。

部署

在目標環境中提供應用程式、新功能或程式碼修正的程序。部署涉及在程式碼庫中實作變更，然後在應用程式環境中建置和執行該程式碼庫。

開發環境

請參閱環境。

偵測性控制

一種安全控制，用於在事件發生後偵測、記錄和提醒。這些控制是第二道防線，提醒您注意繞過現有預防性控制的安全事件。如需詳細資訊，請參閱在 AWS上實作安全控制中的偵測性控制。

發展價值流映射

用於識別限制並排定優先順序，對軟體開發生命週期中的速度和品質產生不利影響的程序。DVSM 擴展了最初為精益生產實踐而設計的價值流映射流程。它著重於創造和通過軟件開發過程中移動價值所需的步驟和團隊。

數字雙胞胎

真實世界系統的虛擬表現法，例如建築物、工廠、工業設備或生產線。數位雙胞胎支援預測性維護、遠端監控和生產最佳化。

維度表

在 star 結構描述中，較小的資料表包含事實資料表中定量資料的相關資料屬性。維度表格屬性通常是文字欄位或離散數字，其行為類似於文字。這些屬性通常用於查詢限制、篩選和結果集標籤。

災難

防止工作負載或系統在其主要部署位置達成其業務目標的事件。這些事件可能是自然災害、技術故障或人為行為造成的結果，例如意外設定錯誤或惡意軟體攻擊。

災難復原 (DR)

您使用的策略和程序，將因災難造成的停機時間和資料遺失降到最低。如需詳細資訊，請參閱 AWS Well-Architected 的架構中的雲端中的工作負載的災難復原 AWS：雲端復原。

DML

請參閱資料庫操作語言。

領域驅動的設計

一種開發複雜軟體系統的方法，它會將其元件與每個元件所服務的不斷發展的領域或核心業務目標相關聯。Eric Evans 在其著作 Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003) 中介紹了這一概念。如需有關如何將領域驅動的設計與 strangler fig 模式搭配使用的資訊，請參閱使用容器和 Amazon API Gateway 逐步現代化舊版 Microsoft ASP.NET (ASMX) Web 服務。

博士

請參閱災難復原。

漂移檢測

追蹤基線組態的偏差。例如，您可以用 AWS CloudFormation 來偵測系統資源中的漂移，也可以用 AWS Control Tower 來偵測 landing zone 中可能會影響法規遵循治理要求的變更。

DVSM

請參閱開發價值流映射。

E

EDA

請參閱探索性資料分析。

邊緣運算

提升 IoT 網路邊緣智慧型裝置運算能力的技術。與雲計算相比，邊緣計算可以減少通信延遲並縮短響應時間。

加密

一種計算過程，將純文本數據（這是人類可讀的）轉換為密文。

加密金鑰

由加密演算法產生的隨機位元的加密字串。金鑰長度可能有所不同，每個金鑰的設計都是不可預測且唯一的。

端序

位元組在電腦記憶體中的儲存順序。大端序系統首先儲存最高有效位元組。小端序系統首先儲存最低有效位元組。

端點

請參閱服務端點。

端點服務

您可以在虛擬私有雲端 (VPC) 中託管以與其他使用者共用的服務。您可以使用其他或 (IAM) 主體建立端點服務， AWS PrivateLink 並將權限授予其他 AWS 帳戶 或 AWS Identity and Access Management (IAM) 主體。這些帳戶或主體可以透過建立介面 VPC 端點私下連接至您的端點服務。如需詳細資訊，請參閱 Amazon Virtual Private Cloud (Amazon VPC) 文件中的建立端點服務。

企業資源規劃

可自動化並管理企業關鍵業務流程 (例如會計、MES 和專案管理) 的系統。

信封加密

使用另一個加密金鑰對某個加密金鑰進行加密的程序。如需詳細資訊，請參閱 AWS Key Management Service (AWS KMS) 文件中的信封加密。

環境

執行中應用程式的執行個體。以下是雲端運算中常見的環境類型：

- 開發環境 – 執行中應用程式的執行個體，只有負責維護應用程式的核心團隊才能使用。開發環境用來測試變更，然後再將開發環境提升到較高的環境。此類型的環境有時稱為測試環境。
- 較低的環境 – 應用程式的所有開發環境，例如用於初始建置和測試的開發環境。
- 生產環境 – 最終使用者可以存取的執行中應用程式的執行個體。在 CI/CD 管道中，生產環境是最後一個部署環境。
- 較高的環境 – 核心開發團隊以外的使用者可存取的所有環境。這可能包括生產環境、生產前環境以及用於使用者接受度測試的環境。

epic

在敏捷方法中，有助於組織工作並排定工作優先順序的功能類別。epic 提供要求和實作任務的高層級描述。例如， AWS CAF 安全史詩包括身份和訪問管理，偵探控制，基礎結構安全性，數據保護和事件響應。如需有關 AWS 遷移策略中的 Epic 的詳細資訊，請參閱計畫實作指南。

ERP

請參閱企業資源規劃。

探索性資料分析 (EDA)

分析資料集以了解其主要特性的過程。您收集或彙總資料，然後執行初步調查以尋找模式、偵測異常並檢查假設。透過計算摘要統計並建立資料可視化來執行 EDA。

F

事實表

[星型架](#)構中的中央表格。它存儲有關業務運營的定量數據。事實資料表通常包含兩種類型的資料欄：包含計量的資料欄，以及包含維度表格外部索引鍵的資料欄。

快速失敗

一種使用頻繁和增量測試來減少開發生命週期的理念。這是敏捷方法的關鍵部分。

故障隔離邊界

在中 AWS 雲端，可用區域、 AWS 區域控制平面或資料平面等界限，可限制故障的影響，並協助改善工作負載的彈性。如需詳細資訊，請參閱AWS 錯誤隔離邊界。

功能分支

請參閱分支。

特徵

用來進行預測的輸入資料。例如，在製造環境中，特徵可能是定期從製造生產線擷取的影像。

功能重要性

特徵對於模型的預測有多重要。這通常表示為可以透過各種技術來計算的數值得分，例如 Shapley Additive Explanations (SHAP) 和積分梯度。如需詳細資訊，請參閱機器學習模型可解釋性：AWS。

特徵轉換

優化 ML 程序的資料，包括使用其他來源豐富資料、調整值、或從單一資料欄位擷取多組資訊。這可讓 ML 模型從資料中受益。例如，如果將「2021-05-27 00:15:37」日期劃分為「2021」、「五月」、「週四」和「15」，則可以協助學習演算法學習與不同資料元件相關聯的細微模式。

FGAC

請參閱精細的存取控制。

精細的存取控制 (FGAC)

使用多個條件來允許或拒絕訪問請求。

閃切遷移

一種資料庫移轉方法，透過變更資料擷取使用連續資料複寫，在最短的時間內移轉資料，而不是使用階段化方法。目標是將停機時間降至最低。

G

地理阻塞

請參閱地理限制。

地理限制 (地理封鎖)

在 Amazon 中 CloudFront，防止特定國家/地區的使用者存取內容分發的選項。您可以使用允許清單或封鎖清單來指定核准和禁止的國家/地區。如需詳細資訊，請參閱 CloudFront 文件中的限制內容的地理分佈。

Gitflow 工作流程

這是一種方法，其中較低和較高環境在原始碼儲存庫中使用不同分支。Gitflow 工作流程被認為是遺留的，基於主幹的工作流程是現代的首選方法。

綠地策略

新環境中缺乏現有基礎設施。對系統架構採用綠地策略時，可以選擇所有新技術，而不會限制與現有基礎設施的相容性，也稱為棕地。如果正在擴展現有基礎設施，則可能會混合棕地和綠地策略。

防護機制

有助於跨組織單位 (OU) 來管控資源、政策和合規的高層級規則。預防性防護機制會強制執行政策，以確保符合合規標準。透過使用服務控制政策和 IAM 許可界限來將其實作。偵測性防護機制可偵測政策違規和合規問題，並產生提醒以便修正。它們是通過使用 AWS Config，Amazon AWS Security Hub GuardDuty， AWS Trusted Advisor亞馬遜檢查 Amazon Inspector 和自定義 AWS Lambda 檢查來實現的。

H

公頃

查看高可用性。

異質資料庫遷移

將來源資料庫遷移至使用不同資料庫引擎的目標資料庫 (例如，Oracle 至 Amazon Aurora)。異質遷移通常是重新架構工作的一部分，而轉換結構描述可能是一項複雜任務。AWS 提供有助於結構描述轉換的 AWS SCT。

高可用性 (HA)

工作負載在遇到挑戰或災難時持續運作的能力，無需干預。HA 系統的設計可自動容錯移轉、持續提供高品質的效能，以及處理不同的負載和故障，並將效能影響降到最低。

歷史學家現代化

一種用於現代化和升級操作技術 (OT) 系統的方法，以更好地滿足製造業的需求。歷史學家是一種類型的數據庫，用於收集和存儲工廠中的各種來源的數據。

異質資料庫遷移

將您的來源資料庫遷移至共用相同資料庫引擎的目標資料庫 (例如，Microsoft SQL Server 至 Amazon RDS for SQL Server)。同質遷移通常是主機轉換或平台轉換工作的一部分。您可以使用原生資料庫公用程式來遷移結構描述。

熱數據

經常存取的資料，例如即時資料或最近的轉譯資料。此資料通常需要高效能的儲存層或類別，才能提供快速的查詢回應。

修補程序

緊急修正生產環境中的關鍵問題。由於其緊迫性，修補程式通常是在典型的 DevOps 發行工作流程之外進行。

超級護理期間

在切換後，遷移團隊在雲端管理和監控遷移的應用程式以解決任何問題的時段。通常，此期間的長度為 1-4 天。在超級護理期間結束時，遷移團隊通常會將應用程式的責任轉移給雲端營運團隊。

I

IaC

查看基礎結構即程式碼。

身分型政策

附加至一或多個 IAM 主體的政策，用於定義其在 AWS 雲端 環境中的許可。

閒置應用程式

90 天期間 CPU 和記憶體平均使用率在 5% 至 20% 之間的應用程式。在遷移專案中，通常會淘汰這些應用程式或將其保留在內部部署。

IIoT

請參閱工業物聯網。

不可變基礎設施

為生產工作負載部署新基礎結構的模型，而不是更新、修補或修改現有基礎結構。不可變的基礎架構本質上比可變基礎架構更加一致、可靠且可預測。如需詳細資訊，請參閱 Well-Architected 的架構中的使用不可變基礎結 AWS 構進行部署最佳作法。

傳入 (輸入) VPC

在 AWS 多帳戶架構中，VPC 可接受、檢查和路由來自應用程式外部的網路連線。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

增量遷移

一種切換策略，您可以在其中將應用程式分成小部分遷移，而不是執行單一、完整的切換。例如，您最初可能只將一些微服務或使用者移至新系統。確認所有項目都正常運作之後，您可以逐步移動其他微服務或使用者，直到可以解除委任舊式系統。此策略可降低與大型遷移關聯的風險。

工業 4.0

Klaus Schwab 於 2016 年推出的一個術語，指的是透過連線能力、即時資料、自動化、分析和 AI/ML 的進步實現製造流程的現代化。

基礎設施

應用程式環境中包含的所有資源和資產。

基礎設施即程式碼 (IaC)

透過一組組態檔案來佈建和管理應用程式基礎設施的程序。IaC 旨在協助您集中管理基礎設施，標準化資源並快速擴展，以便新環境可重複、可靠且一致。

工業物聯網 (IIoT)

在製造業、能源、汽車、醫療保健、生命科學和農業等產業領域使用網際網路連線的感測器和裝置。如需詳細資訊，請參閱建立工業物聯網 (IIoT) 數位轉型策略。

檢查 VPC

在 AWS 多帳戶架構中，集中式 VPC 可管理 VPC（相同或不同 AWS 區域）、網際網路和內部部署網路之間的網路流量檢查。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

物聯網 (IoT)

具有內嵌式感測器或處理器的相連實體物體網路，其透過網際網路或本地通訊網路與其他裝置和系統進行通訊。如需詳細資訊，請參閱什麼是 IoT？

可解釋性

機器學習模型的一個特徵，描述了人類能夠理解模型的預測如何依賴於其輸入的程度。如需詳細資訊，請參閱 AWS 的機器學習模型可解釋性。

IoT

請參閱物聯網。

IT 資訊庫 (ITIL)

一組用於交付 IT 服務並使這些服務與業務需求保持一致的最佳實務。ITIL 為 ITSM 提供了基礎。

IT 服務管理 (ITSM)

與組織的設計、實作、管理和支援 IT 服務關聯的活動。如需有關將雲端操作與 ITSM 工具整合的資訊，請參閱操作整合指南。

ITIL

請參閱 IT 資訊庫。

ITSM

請參閱 IT 服務管理。

L

標籤式存取控制 (LBAC)

強制存取控制 (MAC) 的實作，其中每個使用者和資料本身都明確指派一個安全性標籤值。使用者安全性標籤與資料安全性標籤之間的交集決定了使用者可以看到哪些列與欄。

登陸區域

landing zone 是一個架構良好的多帳戶 AWS 環境，具有可擴展性和安全性。這是一個起點，您的組織可以從此起點快速啟動和部署工作負載與應用程式，並對其安全和基礎設施環境充滿信心。如需有關登陸區域的詳細資訊，請參閱設定安全且可擴展的多帳戶 AWS 環境。

大型遷移

遷移 300 部或更多伺服器。

LBAC

請參閱以標示為基礎的存取控制。

最低權限

授予執行任務所需之最低許可的安全最佳實務。如需詳細資訊，請參閱 IAM 文件中的套用最低權限許可。

隨即轉移

見 7 盧比

小端序系統

首先儲存最低有效位元組的系統。另請參閱 「位元順序」。

較低的環境

請參閱環境。

M

機器學習 (ML)

一種使用演算法和技術進行模式識別和學習的人工智慧。機器學習會進行分析並從記錄的資料 (例如物聯網 (IoT) 資料) 中學習，以根據模式產生統計模型。如需詳細資訊，請參閱機器學習。

主要分支

請參閱分支。

惡意軟體

旨在危及計算機安全性或隱私的軟件。惡意軟件可能會破壞計算機系統，洩漏敏感信息或獲得未經授權的訪問。惡意軟體的例子包括病毒、蠕蟲、勒索軟體、特洛伊木馬程式、間諜軟體和鍵盤記錄程式。

受管理服務

AWS 服務 用於 AWS 操作基礎架構層、作業系統和平台，並且您可以存取端點以儲存和擷取資料。Amazon Simple Storage Service (Amazon S3) 和 Amazon DynamoDB 是受管服務的範例。這些也稱為抽象服務。

製造執行系統

用於跟踪，監控，記錄和控制生產過程的軟件系統，可在現場將原材料轉換為成品。

MAP

請參閱 Migration Acceleration Program。

機制

一個完整的過程，您可以在其中創建工具，推動工具的採用，然後檢查結果以進行調整。機制是一個循環，它加強和改善自己，因為它運行。如需詳細資訊，請參閱 AWS Well-Architected 的架構中建置機制。

成員帳戶

屬於 AWS 帳戶 中組織的管理帳戶以外的所有帳戶 AWS Organizations。一個帳戶一次只能是一個組織的成員。

MES

請參閱製造執行系統。

郵件佇列遙測傳輸 (MQTT)

以發佈/訂閱模式為基礎的輕量型 machine-to-machine (M2M) 通訊協定，適用於資源受限 IoT 裝置。

微服務

一種小型的獨立服務，它可透過定義明確的 API 進行通訊，通常由小型獨立團隊擁有。例如，保險系統可能包含對應至業務能力 (例如銷售或行銷) 或子領域 (例如購買、索賠或分析) 的微服務。微服

務的優點包括靈活性、彈性擴展、輕鬆部署、可重複使用的程式碼和適應力。如需詳細資訊，請參閱使用 AWS 無伺服器服務整合微服務。

微服務架構

一種使用獨立元件來建置應用程式的方法，這些元件會以微服務形式執行每個應用程式程序。這些微服務會使用輕量型 API，透過明確定義的介面進行通訊。此架構中的每個微服務都可以進行更新、部署和擴展，以滿足應用程式特定功能的需求。如需詳細資訊，請參閱上 AWS的實作微服務。

Migration Acceleration Program (MAP)

提供諮詢支援、訓練和服務的 AWS 計畫，協助組織為移轉至雲端建立穩固的營運基礎，並協助抵消移轉的初始成本。MAP 包括用於有條不紊地執行舊式遷移的遷移方法以及一組用於自動化和加速常見遷移案例的工具。

大規模遷移

將大部分應用程式組合依波次移至雲端的程序，在每個波次中，都會以更快的速度移動更多應用程式。此階段使用從早期階段學到的最佳實務和經驗教訓來實作團隊、工具和流程的遷移工廠，以透過自動化和敏捷交付簡化工作負載的遷移。這是 AWS 遷移策略的第三階段。

遷移工廠

可透過自動化、敏捷的方法簡化工作負載遷移的跨職能團隊。移轉工廠團隊通常包括營運、業務分析師和擁有者、移轉工程師、開發人員和 DevOps 專業人員。20% 至 50% 之間的企業應用程式組合包含可透過工廠方法優化的重複模式。如需詳細資訊，請參閱此內容集中的遷移工廠的討論和雲端遷移工廠指南。

遷移中繼資料

有關完成遷移所需的應用程式和伺服器的資訊。每種遷移模式都需要一組不同的遷移中繼資料。移轉中繼資料的範例包括目標子網路、安全性群組和 AWS 帳戶。

遷移模式

可重複的遷移任務，詳細描述遷移策略、遷移目的地以及所使用的遷移應用程式或服務。範例：使 AWS 用應用程式遷移服務將遷移重新託管到 Amazon EC2。

遷移組合評定 (MPA)

這是一種線上工具，可提供驗證要移轉至的商業案例的 AWS 雲端資訊。MPA 提供詳細的組合評定 (伺服器適當規模、定價、總體擁有成本比較、遷移成本分析) 以及遷移規劃 (應用程式資料分析和資料收集、應用程式分組、遷移優先順序，以及波次規劃)。所有 AWS 顧問和 APN 合作夥伴顧問均可免費使用 MPA 工具 (需要登入)。

遷移準備程度評定 (MRA)

使用 AWS CAF 獲得有關組織雲端準備狀態、識別優勢和弱點，以及建立行動計劃以縮小已識別差距的過程。如需詳細資訊，請參閱遷移準備程度指南。MRA 是 AWS 遷移策略的第一階段。

遷移策略

將工作負載移轉至 AWS 雲端. 如需詳細資訊，請參閱本詞彙表中的 7 Rs 項目，並參閱動員您的組織以加速大規模移轉。

機器學習 (ML)

請參閱機器學習。

現代化

將過時的 (舊版或單一) 應用程式及其基礎架構轉換為雲端中靈活、富有彈性且高度可用的系統，以降低成本、提高效率並充分利用創新。如需詳細資訊，請參閱. AWS 雲端

現代化準備程度評定

這項評估可協助判斷組織應用程式的現代化準備程度；識別優點、風險和相依性；並確定組織能夠在多大程度上支援這些應用程式的未來狀態。評定的結果就是目標架構的藍圖、詳細說明現代化程序的開發階段和里程碑的路線圖、以及解決已發現的差距之行動計畫。如需詳細資訊，請參閱評估應用程式的現代化準備程度 AWS 雲端。

單一應用程式 (單一)

透過緊密結合的程序作為單一服務執行的應用程式。單一應用程式有幾個缺點。如果一個應用程式功能遇到需求激增，則必須擴展整個架構。當程式碼庫增長時，新增或改進單一應用程式的功能也會變得更加複雜。若要解決這些問題，可以使用微服務架構。如需詳細資訊，請參閱將單一體系分解為微服務。

MPA

請參閱移轉組合評估。

MQTT

請參閱佇列遙測傳輸的郵件。

多類別分類

一個有助於產生多類別預測的過程 (預測兩個以上的結果之一)。例如，機器學習模型可能會詢問「此產品是書籍、汽車還是電話？」 或者「這個客戶對哪種產品類別最感興趣？」

可變的基礎

一種模型，用於更新和修改生產工作負載的現有基礎結構。為了提高一致性，可靠性和可預測性，AWS Well-Architected 框架建議使用不可變的基礎結構作為最佳實踐。

O

OAC

請參閱原始存取控制。

OAI

請參閱原始存取身分。

OCM

請參閱組織變更管理。

離線遷移

一種遷移方法，可在遷移過程中刪除來源工作負載。此方法涉及延長停機時間，通常用於小型非關鍵工作負載。

OI

請參閱作業整合。

OLA

請參閱作業層級協定。

線上遷移

一種遷移方法，無需離線即可將來源工作負載複製到目標系統。連接至工作負載的應用程式可在遷移期間繼續運作。此方法涉及零至最短停機時間，通常用於關鍵的生產工作負載。

OPCA

請參閱開放程序通訊-統一架構。

開放程序通訊-統一架構 (OPC-UA)

用於工業自動化的 machine-to-machine (M2M) 通訊協定。OPC-UA 提供數據加密，身份驗證和授權方案的互操作性標準。

操作水準協議 (OLA)

一份協議，闡明 IT 職能群組承諾向彼此提供的內容，以支援服務水準協議 (SLA)。

操作準備程度檢討 (ORR)

問題和相關最佳做法的檢查清單，可協助您瞭解、評估、預防或減少事件和可能的故障範圍。如需詳細資訊，請參閱 AWS Well-Architected 的架構中的作業準備檢閱 (ORR)。

操作技術

可與實體環境搭配使用的硬體和軟體系統，以控制工業作業、設備和基礎設施。在製造業中，整合 OT 和資訊技術 (IT) 系統是工業 4.0 轉型的關鍵焦點。

操作整合 (OI)

在雲端中將操作現代化的程序，其中包括準備程度規劃、自動化和整合。如需詳細資訊，請參閱操作整合指南。

組織追蹤

由建立的追蹤 AWS CloudTrail 記錄中組織 AWS 帳戶 中所有人的所有事件 AWS Organizations。在屬於組織的每個 AWS 帳戶 中建立此追蹤，它會跟蹤每個帳戶中的活動。如需詳細資訊，請參閱CloudTrail文件中的為組織建立追蹤。

組織變更管理 (OCM)

用於從人員、文化和領導力層面管理重大、顛覆性業務轉型的架構。OCM 透過加速變更採用、解決過渡問題，以及推動文化和組織變更，協助組織為新系統和策略做好準備，並轉移至新系統和策略。在 AWS 移轉策略中，這個架構稱為人員加速，因為雲端採用專案所需的變更速度。如需詳細資訊，請參閱 OCM 指南。

原始存取控制 (OAC)

在中 CloudFront，限制存取權限以保護 Amazon Simple Storage Service (Amazon S3) 內容的增強選項。OAC 支援所有 S3 儲存貯體 AWS 區域、伺服器端加密 AWS KMS (SSE-KMS)，以及 S3 儲存貯體的動態PUT和DELETE請求。

原始存取身分 (OAI)

在中 CloudFront，用於限制存取以保護 Amazon S3 內容的選項。當您使用 OAI 時， CloudFront 會建立 Amazon S3 可用來進行驗證的主體。經驗證的主體只能透過特定散發存取 S3 儲存 CloudFront 貯體中的內容。另請參閱 OAC，它可提供更精細且增強的存取控制。

ORR

請參閱作業整備檢閱。

OT

請參閱操作技術。

傳出 (輸出) VPC

在 AWS 多帳戶架構中，處理從應用程式內啟動的網路連線的 VPC。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

P

許可界限

附接至 IAM 主體的 IAM 管理政策，可設定使用者或角色擁有的最大許可。如需詳細資訊，請參閱 IAM 文件中的許可界限。

個人識別資訊 (PII)

直接查看或與其他相關數據配對時，可用於合理推斷個人身份的信息。PII 的範例包括姓名、地址和聯絡資訊。

PII

請參閱個人識別資訊。

手冊

一組預先定義的步驟，可擷取與遷移關聯的工作，例如在雲端中提供核心操作功能。手冊可以採用指令碼、自動化執行手冊或操作現代化環境所需的程序或步驟摘要的形式。

公司

請參閱可編程邏輯控制器

PLM

查看產品生命週期管理。

政策

可以定義權限 (請參閱以身分識別為基礎的策略)、指定存取條件 (請參閱以資源為基礎的策略) 或定義組織中所有帳戶的最大權限的物件 AWS Organizations (請參閱服務控制策略)。

混合持久性

根據資料存取模式和其他需求，獨立選擇微服務的資料儲存技術。如果您的微服務具有相同的資料儲存技術，則其可能會遇到實作挑戰或效能不佳。如果微服務使用最適合其需求的資料儲存，則可以更輕鬆地實作並達到更好的效能和可擴展性。如需詳細資訊，請參閱在微服務中啟用資料持久性。

組合評定

探索、分析應用程式組合並排定其優先順序以規劃遷移的程序。如需詳細資訊，請參閱評估遷移準備程度。

述詞

傳回`true`或的查詢條件`false`，通常位於子`WHERE`句中。

謂詞下推

一種資料庫查詢最佳化技術，可在傳輸前篩選查詢中的資料。這樣可減少必須從關聯式資料庫擷取和處理的資料量，並改善查詢效能。

預防性控制

旨在防止事件發生的安全控制。這些控制是第一道防線，可協助防止對網路的未經授權存取或不必要變更。如需詳細資訊，請參閱在 AWS上實作安全控制中的預防性控制。

委託人

中 AWS 可執行動作和存取資源的實體。此實體通常是 IAM 角色或使用者的根使用者。 AWS 帳戶如需詳細資訊，請參閱 IAM 文件中角色術語和概念中的主體。

隱私設計

一種系統工程方法，在整個工程過程中將隱私權納入考量。

私有託管區域

一種容器，它包含有關您希望 Amazon Route 53 如何回應一個或多個 VPC 內的域及其子域之 DNS 查詢的資訊。如需詳細資訊，請參閱 Route 53 文件中的使用私有託管區域。

主動控制

一種安全控制項，旨在防止部署不符合規範的資源。這些控制項會在資源佈建之前進行掃描。如果資源不符合控制項，則不會佈建該資源。如需詳細資訊，請參閱 AWS Control Tower 文件中的控制項參考指南，並參閱實作安全性控制中的主動控制 AWS。

產品生命週期管理 (PLM)

在產品的整個生命週期中管理資料和流程，從設計、開發、上市到成長與成熟度，再到下降和移除。

生產環境

請參閱環境。

可編程邏輯控制器 (PLC)

在製造業中，一台高度可靠且適應性強的計算機，可監控機器並自動化製造過程。

化名化

以預留位置值取代資料集中的個人識別碼的程序。化名化有助於保護個人隱私。假名化數據仍被認為是個人數據。

發布/訂閱 (發布/訂閱)

一種模式，可在微服務之間實現非同步通訊，以提高延展性和回應能力 例如，在微服務型 MES 中，微服務可以將事件訊息發佈到其他微服務可訂閱的通道。系統可以在不變更發佈服務的情況下新增微服務。

Q

查詢計劃

一系列步驟，如指示，用來存取 SQL 關聯式資料庫系統中的資料。

查詢計劃迴歸

在資料庫服務優化工具選擇的計畫比對資料庫環境進行指定的變更之前的計畫不太理想時。這可能因為對統計資料、限制條件、環境設定、查詢參數繫結的變更以及資料庫引擎的更新所導致。

R

拉齊矩陣

請參閱負責任，負責，諮詢，通知（RACI）。

勒索軟體

一種惡意軟體，旨在阻止對計算機系統或資料的存取，直到付款為止。

拉西矩陣

請參閱負責任，負責，諮詢，通知（RACI）。

RCAC

請參閱列與欄存取控制。

僅供讀取複本

用於唯讀用途的資料庫複本。您可以將查詢路由至僅供讀取複本以減少主資料庫的負載。

重新建築師

見 7 盧比

復原點目標 (RPO)

自上次資料復原點以來可接受的時間上限。這決定了最後一個恢復點和服務中斷之間可接受的數據丟失。

復原時間目標 (RTO)

服務中斷與恢復服務之間的最大可接受延遲。

重構

見 7 盧比

區域

地理區域中的 AWS 資源集合。每個都 AWS 區域 是隔離和獨立於其他的，以提供容錯能力，穩定性和彈性。如需詳細資訊，請參閱指定 AWS 區域 您的帳戶可以使用的項目。

迴歸

預測數值的 ML 技術。例如，為了解決「這房子會賣什麼價格？」的問題 ML 模型可以使用線性迴歸模型，根據已知的房屋事實 (例如，平方英尺) 來預測房屋的銷售價格。

重新主持

見 7 盧比

版本

在部署程序中，它是將變更提升至生產環境的動作。

重新定位

見 7 盧比

再平台

見 7 盧比

買回

見 7 盧比

彈性

應用程式抵抗或從中斷中復原的能力。在規劃備援時，高可用性和災難復原是常見的考量因素。AWS 雲端如需詳細資訊，請參閱AWS 雲端 復原力。

資源型政策

附接至資源的政策，例如 Amazon S3 儲存貯體、端點或加密金鑰。這種類型的政策會指定允許存取哪些主體、支援的動作以及必須滿足的任何其他條件。

負責者、當責者、事先諮詢者和事後告知者 (RACI) 矩陣

定義移轉活動和雲端作業所涉及之所有各方的角色與責任的矩陣。矩陣名稱衍生自矩陣中定義的責任型別：負責 (R)、負責 (A)、諮詢 (C) 及通知 (I)。支撐（S）類型是可選的。如果您包含支援，則該矩陣稱為 RASCI 矩陣，如果您將其排除，則稱為 R ACI 矩陣。

回應性控制

一種安全控制，旨在驅動不良事件或偏離安全基準的補救措施。如需詳細資訊，請參閱在 AWS上實作安全控制中的回應性控制。

保留

見 7 盧比

退休

見 7 盧比

旋轉

定期更新密碼以使攻擊者更難以存取認證的程序。

資料列與資料行存取控制 (RCAC)

使用已定義存取規則的基本、彈性 SQL 運算式。RCAC 由資料列權限和資料行遮罩所組成。

RPO

請參閱復原點目標。

RTO

請參閱復原時間目標。

執行手冊

執行特定任務所需的一組手動或自動程序。這些通常是為了簡化重複性操作或錯誤率較高的程序而建置。

S

SAML 2.0

許多身份提供者（IdPs）使用的開放標準。此功能可啟用聯合單一登入 (SSO)，因此使用者可以登入 AWS Management Console 或呼叫 AWS API 作業，而不必為組織中的每個人在 IAM 中建立使用者。如需有關以 SAML 2.0 為基礎的聯合詳細資訊，請參閱 IAM 文件中的關於以 SAML 2.0 為基礎的聯合。

斯卡達

請參閱監督控制和資料擷取。

SCP

請參閱服務控制策略。

秘密

您以加密形式儲存的機密或受限制資訊，例如密碼或使用者認證。 AWS Secrets Manager它由秘密值及其中繼資料組成。密碼值可以是二進位、單一字串或多個字串。如需詳細資訊，請參閱「Secrets Manager 碼中有什麼內容？ 在 Secrets Manager 文檔中。

安全控制

一種技術或管理防護機制，它可預防、偵測或降低威脅行為者利用安全漏洞的能力。安全性控制有四種主要類型：預防性、偵測、回應式和主動式。

安全強化

減少受攻擊面以使其更能抵抗攻擊的過程。這可能包括一些動作，例如移除不再需要的資源、實作授予最低權限的安全最佳實務、或停用組態檔案中不必要的功能。

安全資訊與事件管理 (SIEM) 系統

結合安全資訊管理 (SIM) 和安全事件管理 (SEM) 系統的工具與服務。SIEM 系統會收集、監控和分析來自伺服器、網路、裝置和其他來源的資料，以偵測威脅和安全漏洞，並產生提醒。

安全回應自動化

預先定義且程式化的動作，其設計用來自動回應或修復安全性事件。這些自動化作業可做為偵探或回應式安全控制項，協助您實作 AWS 安全性最佳實務。自動回應動作的範例包括修改 VPC 安全群組、修補 Amazon EC2 執行個體或輪換登入資料。

伺服器端加密

在其目的地的數據加密，通 AWS 服務 過接收它。

服務控制政策 (SCP)

為 AWS Organizations中的組織的所有帳戶提供集中控制許可的政策。SCP 會定義防護機制或設定管理員可委派給使用者或角色的動作限制。您可以使用 SCP 作為允許清單或拒絕清單，以指定允許或禁止哪些服務或動作。如需詳細資訊，請參閱 AWS Organizations 文件中的服務控制原則。

服務端點

的進入點的 URL AWS 服務。您可以使用端點，透過程式設計方式連接至目標服務。如需詳細資訊，請參閱 AWS 一般參考 中的 AWS 服務 端點。

服務水準協議 (SLA)

一份協議，闡明 IT 團隊承諾向客戶提供的服務，例如服務正常執行時間和效能。

服務等級指示器 (SLI)

對服務效能層面的測量，例如錯誤率、可用性或輸送量。

服務等級目標 (SLO)

代表服務狀況的目標測量結果，由服務層次指示器測量。

共同責任模式

描述您在雲端安全性和合規方面共享的責任的模型。 AWS AWS 負責雲端的安全性，而您則負責雲端的安全性。如需詳細資訊，請參閱共同責任模式。

暹

請參閱安全性資訊和事件管理系統。

單點故障 (SPF)

應用程式的單一重要元件發生故障，可能會中斷系統。

SLA

請參閱服務等級協議。

SLI

請參閱服務層級指示器。

SLO

請參閱服務等級目標。

split-and-seed 模型

擴展和加速現代化專案的模式。定義新功能和產品版本時，核心團隊會進行拆分以建立新的產品團隊。這有助於擴展組織的能力和服務，提高開發人員生產力，並支援快速創新。如需詳細資訊，請參閱中的應用程式現代化的階段化方法。 AWS 雲端

痙攣

請參閱單一故障點。

星型綱要

使用一個大型事實資料表來儲存交易或測量資料，並使用一或多個較小的維度表格來儲存資料屬性的資料庫組織結構。這種結構是專為在數據倉庫中使用或用於商業智能目的。

Strangler Fig 模式

一種現代化單一系統的方法，它會逐步重寫和取代系統功能，直到舊式系統停止使用為止。此模式源自無花果藤，它長成一棵馴化樹並最終戰勝且取代了其宿主。該模式由 Martin Fowler 引入，作為重寫單一系統時管理風險的方式。如需有關如何套用此模式的範例，請參閱使用容器和 Amazon API Gateway 逐步現代化舊版 Microsoft ASP.NET (ASMX) Web 服務。

子網

您 VPC 中的 IP 地址範圍。子網必須位於單一可用區域。

監督控制與資料擷取 (SCADA)

在製造業中，使用硬體與軟體來監控實體資產與生產作業的系統。

對稱加密

使用相同金鑰來加密及解密資料的加密演算法。

合成測試

以模擬使用者互動以偵測潛在問題或監控效能的方式測試系統。您可以使用 Amazon CloudWatch Synthetics 來創建這些測試。

T

標籤

作為組織 AWS 資源的中繼資料的索引鍵值配對。標籤可協助您管理、識別、組織、搜尋及篩選資源。如需詳細資訊，請參閱標記您的 AWS 資源。

目標變數

您嘗試在受監督的 ML 中預測的值。這也被稱為結果變數。例如，在製造設定中，目標變數可能是產品瑕疵。

任務清單

用於透過執行手冊追蹤進度的工具。任務清單包含執行手冊的概觀以及要完成的一般任務清單。對於每個一般任務，它包括所需的預估時間量、擁有者和進度。

測試環境

請參閱環境。

訓練

為 ML 模型提供資料以供學習。訓練資料必須包含正確答案。學習演算法會在訓練資料中尋找將輸入資料屬性映射至目標的模式 (您想要預測的答案)。它會輸出擷取這些模式的 ML 模型。可以使用 ML 模型，來預測您不知道的目標新資料。

傳輸閘道

可以用於互連 VPC 和內部部署網路的網路傳輸中樞。如需詳細資訊，請參閱 AWS Transit Gateway 文件中的傳輸閘道是什麼。

主幹型工作流程

這是一種方法，開發人員可在功能分支中本地建置和測試功能，然後將這些變更合併到主要分支中。然後，主要分支會依序建置到開發環境、生產前環境和生產環境中。

受信任的存取權

授與權限給您指定的服務，以代表您在組織內 AWS Organizations 及其帳戶中執行工作。受信任的服務會在需要該角色時，在每個帳戶中建立服務連結角色，以便為您執行管理工作。如需詳細資訊，請參閱 AWS Organizations 文件中的AWS Organizations 與其他 AWS 服務搭配使用。

調校

變更訓練程序的各個層面，以提高 ML 模型的準確性。例如，可以透過產生標籤集、新增標籤、然後在不同的設定下多次重複這些步驟來訓練 ML 模型，以優化模型。

雙比薩團隊

一個小 DevOps 團隊，你可以餵兩個比薩餅。雙披薩團隊規模可確保軟體開發中的最佳協作。

U

不確定性

這是一個概念，指的是不精確、不完整或未知的資訊，其可能會破壞預測性 ML 模型的可靠性。有兩種類型的不確定性：認知不確定性是由有限的、不完整的資料引起的，而隨機不確定性是由資料中固有的噪聲和隨機性引起的。如需詳細資訊，請參閱量化深度學習系統的不確定性指南。

無差別的任務

也稱為繁重工作，是創建和操作應用程序所必需的工作，但不能為最終用戶提供直接價值或提供競爭優勢。無差異化作業的範例包括採購、維護和容量規劃。

較高的環境

請參閱環境。

V

清空

一種資料庫維護操作，涉及增量更新後的清理工作，以回收儲存並提升效能。

版本控制

追蹤變更的程序和工具，例如儲存庫中原始程式碼的變更。

VPC 對等互連

兩個 VPC 之間的連線，可讓您使用私有 IP 地址路由流量。如需詳細資訊，請參閱 Amazon VPC 文件中的什麼是 VPC 對等互連。

漏洞

會危及系統安全性的軟體或硬體瑕疵。

W

暖快取

包含經常存取的目前相關資料的緩衝快取。資料庫執行個體可以從緩衝快取讀取，這比從主記憶體或磁碟讀取更快。

溫暖的數據

不常存取的資料。查詢此類資料時，通常可以接受中度緩慢的查詢。

視窗功能

一種 SQL 函數，可對以某種方式與當前記錄相關的一組行執行計算。視窗函數對於處理工作非常有用，例如計算移動平均值或根據目前列的相對位置存取列的值。

工作負載

提供商業價值的資源和程式碼集合，例如面向客戶的應用程式或後端流程。

工作串流

遷移專案中負責一組特定任務的功能群組。每個工作串流都是獨立的，但支援專案中的其他工作串流。例如，組合工作串流負責排定應用程式、波次規劃和收集遷移中繼資料的優先順序。組合工作串流將這些資產交付至遷移工作串流，然後再遷移伺服器和應用程式。

蠕蟲

看到寫一次，多讀。

WQF

請參閱 AWS 工作負載資格架構。

寫一次，多讀（WORM）

一種儲存模型，可單次寫入資料並防止資料遭到刪除或修改。授權用戶可以根據需要多次讀取數據，但無法更改數據。這種數據存儲基礎設施被認為是不可變的。

Z

零日漏洞

一種利用零時差漏洞的攻擊，通常是惡意軟件。

零時差漏洞

生產系統中未緩解的瑕疵或弱點。威脅參與者可以利用這種類型的漏洞攻擊系統。由於攻擊，開發人員經常意識到該漏洞。

殭屍應用程式

CPU 和記憶體平均使用率低於 5% 的應用程式。在遷移專案中，通常會淘汰這些應用程式。

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。