다음과 같은 환경에서 아파치 아이스버그 사용하기 AWS

AWS 규범적 지침

AWS 규범적 지침: 다음과 같은 환경에서 아파치 아이스버그 사용하기 AWS

Table of Contents

AWS에서 아파치 아이스버그 사용

Amazon Web Services (기고자)

2024년 4월 (문서 기록)

Apache Iceberg는 성능을 향상시키면서 테이블 관리를 간소화하는 오픈 소스 테이블 형식입니다. AWS Amazon EMR, Amazon Athena AWS Glue, Amazon Redshift와 같은 분석 서비스에는 아파치 아이스버그에 대한 기본 지원이 포함되어 있으므로 Amazon Simple Storage Service (Amazon S3) 를 기반으로 트랜잭션 데이터 레이크를 쉽게 구축할 수 있습니다. AWS

이 기술 가이드에서는 다양한 AWS 서비스환경에서 Apache Iceberg를 시작하는 방법에 대한 지침을 제공하고 비용 및 성능을 최적화하면서 규모에 맞게 Apache Iceberg를 실행하기 위한 모범 사례 및 권장 사항을 포함합니다. AWS

이 가이드는 Apache Iceberg를 빠르게 시작하려는 초보 사용자부터 기존 Apache Iceberg 워크로드를 최적화하고 조정하려는 고급 사용자에 이르기까지 Apache Iceberg를 사용하는 모든 사용자에게 적용됩니다. AWS AWS

이 가이드에서 다루는 내용은 다음과 같습니다.

모던 데이터 레이크

최신 데이터 레이크의 고급 사용 사례

데이터 레이크는 비용, 확장성 및 유연성 측면에서 데이터를 저장하는 데 가장 적합한 옵션 중 하나를 제공합니다. 데이터 레이크를 사용하면 저렴한 비용으로 대량의 정형 및 비정형 데이터를 보존하고, 이 데이터를 비즈니스 인텔리전스 보고부터 빅데이터 처리, 실시간 분석, 기계 학습, 제너레이티브 인공 지능 (AI) 에 이르기까지 다양한 유형의 분석 워크로드에 사용하여 더 나은 의사 결정을 내릴 수 있습니다.

이러한 이점에도 불구하고 데이터 레이크는 처음에는 데이터베이스와 유사한 기능을 제공하도록 설계되지 않았습니다. 데이터 레이크는 원자성, 일관성, 격리성, 내구성 (ACID) 처리 시맨틱을 지원하지 않습니다. 이러한 시맨틱은 다양한 기술을 사용하여 수백 또는 수천 명의 사용자에 대해 대규모로 데이터를 효과적으로 최적화하고 관리하는 데 필요할 수 있습니다. 데이터 레이크는 다음 기능에 대한 기본 지원을 제공하지 않습니다.

- 비즈니스의 데이터 변경에 따라 레코드 수준의 효율적인 업데이트 및 삭제 수행
- 테이블이 수백만 개의 파일과 수십만 개의 파티션으로 확장될 때 쿼리 성능 관리
- 여러 명의 동시 작성자와 독자 간의 데이터 일관성 보장
- 작업 도중에 쓰기 작업이 실패할 경우의 데이터 손상 방지
- 데이터 세트를 (부분적으로) 다시 작성하지 않고 시간이 지남에 따라 테이블 스키마를 발전시킵니다.

이러한 문제는 CDC (변경 데이터 캡처) 처리 등의 사용 사례나 개인 정보 보호, 데이터 삭제, 스트리밍 데이터 수집과 관련된 사용 사례에서 특히 많이 발생하며, 이로 인해 테이블이 최적화되지 않을 수 있습니다.

기존 Hive 형식 테이블을 사용하는 데이터 레이크는 전체 파일에 대한 쓰기 작업만 지원합니다. 따라서 업데이트 및 삭제를 구현하기 어렵고 시간과 비용이 많이 듭니다. 또한 데이터 무결성과 일관성을 보장하려면 ACID 준수 시스템에서 제공되는 동시성 제어 및 보장이 필요합니다.

이러한 문제를 극복하기 위해 Apache Iceberg는 데이터 레이크의 최적화 및 관리 오버헤드를 단순화하는 데이터베이스와 유사한 추가 기능을 제공하는 동시에 Amazon Simple Storage Service (Amazon S3) 와 같은 비용 효율적인 시스템의 스토리지를 계속 지원합니다.

아파치 아이스버그 소개

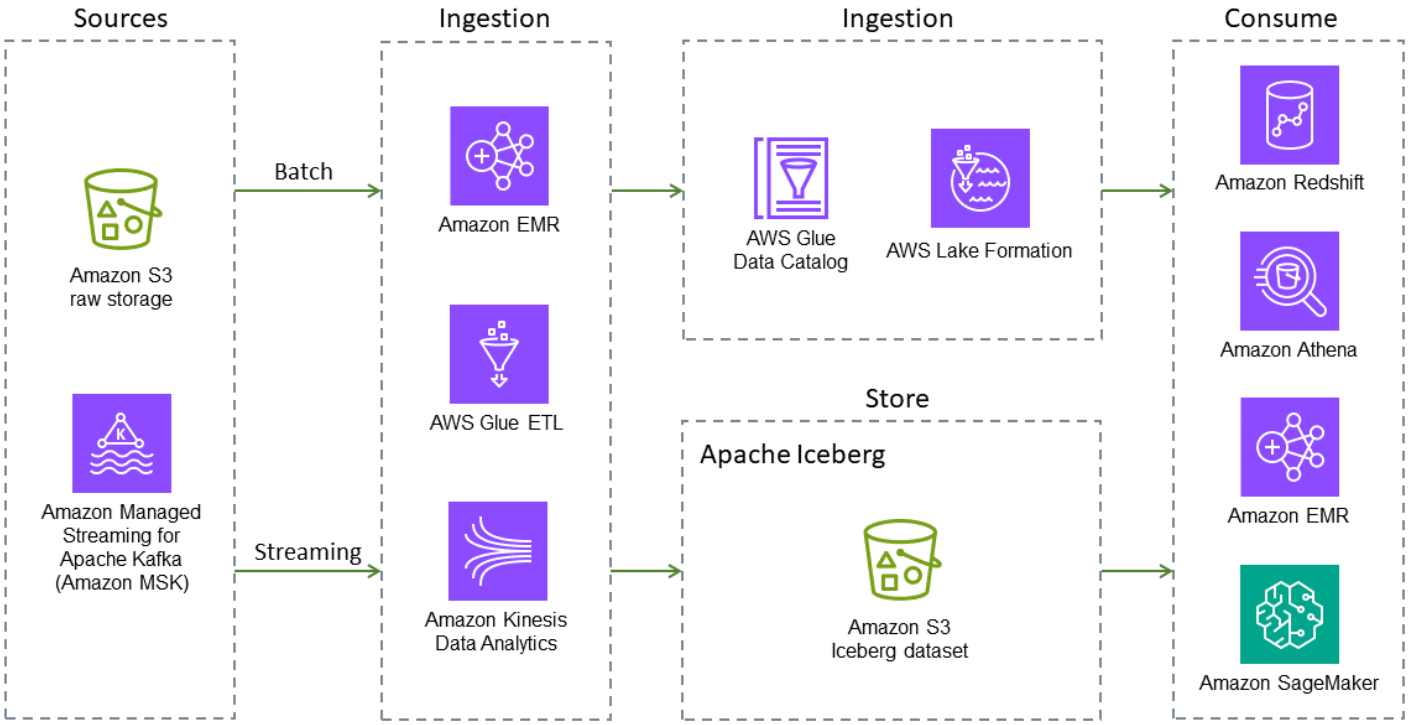
Apache Iceberg는 이전에는 데이터베이스 또는 데이터 웨어하우스에서만 사용할 수 있었던 기능을 데이터 레이크 테이블에 제공하는 오픈 소스 테이블 형식입니다. 확장성과 성능을 고려하여 설계되었으며 수백 기가바이트가 넘는 테이블을 관리하는 데 적합합니다. Iceberg 테이블의 주요 기능 중 일부는 다음과 같습니다.

- 삭제, 업데이트, 병합Iceberg는 데이터 레이크 테이블과 함께 사용할 데이터 웨어하우징용 표준 SQL 명령을 지원합니다.
- 빠른 스캔 계획 및 고급 필터링. Iceberg는 엔진에서 쿼리 계획 및 실행 속도를 높이는 데 사용할 수 있는 파티션 및 열 수준 통계와 같은 메타데이터를 저장합니다.
- 전체 스키마 진화. Iceberg는 부작용 없이 열 추가, 삭제, 업데이트 또는 이름 변경을 지원합니다.
- 파티션 진화. 데이터 볼륨 또는 쿼리 패턴 변경에 따라 테이블의 파티션 레이아웃을 업데이트할 수 있습니다. Iceberg는 테이블이 분할되는 열을 변경하거나 복합 파티션에 열을 추가하거나 복합 파티션에서 열을 제거하는 기능을 지원합니다.
- 숨겨진 파티셔닝. 이 기능은 불필요한 파티션을 자동으로 읽는 것을 방지합니다. 따라서 사용자는 테이블의 파티션 세부 정보를 이해하거나 쿼리에 추가 필터를 추가할 필요가 없습니다.
- 버전 롤백. 사용자는 트랜잭션 전 상태로 되돌아가 문제를 빠르게 수정할 수 있습니다.
- 시간 여행. 사용자는 테이블의 이전 특정 버전을 쿼리할 수 있습니다.
- 직렬화 가능한 격리. 테이블 변경은 원자적이므로 독자는 부분적 변경이나 커밋되지 않은 변경 내용을 확인할 수 없습니다.
- 동시 작성자. Iceberg는 낙관적 동시성을 사용하여 여러 거래가 성공할 수 있도록 합니다. 충돌이 발생할 경우 작성자 중 한 명이 트랜잭션을 다시 시도해야 합니다.
- 파일 형식을 엽니다. 아이스버그는 아파치 파켓, 아파치 아브로, 아파치 ORC 등 다양한 오픈 소스 파일 형식을 지원합니다.

요약하면 Iceberg 형식을 사용하는 데이터 레이크는 트랜잭션 일관성, 속도, 규모 및 스키마 발전의 이점을 누릴 수 있습니다. 이러한 기능 및 기타 Iceberg 기능에 대한 자세한 내용은 Apache Iceberg 설명서를 참조하십시오.

AWS 아파치 아이스버그 지원

아파치 아이스버그는 인기 있는 오픈 소스 데이터 처리 프레임워크와 Amazon EMR, AWS 서비스 아마존 아테나,아마존 Redshift 등에서 지원됩니다. AWS Glue 다음 다이어그램은 Iceberg를 기반으로 하는 데이터 레이크의 단순화된 참조 아키텍처를 보여줍니다.



다음은 네이티브 AWS 서비스 Iceberg 통합을 제공합니다. 간접적으로 또는 Iceberg 라이브러리를 패키징하여 Iceberg와 상호 작용할 수 AWS 서비스 있는 추가 기능도 있습니다.

- Amazon S3는 내구성, 가용성, 확장성, 보안, 규정 준수 및 감사 기능으로 인해 데이터 레이크를 구축하기에 가장 좋은 장소입니다. Iceberg는 Amazon S3와 원활하게 상호 작용하도록 설계 및 구축되었으며 Iceberg 설명서에 나열된 다양한 Amazon S3 기능을 지원합니다.
- Amazon EMR은 Apache Spark, Flink, Trino, Hive 등의 오픈 소스 프레임워크를 사용하여 페타바이트 규모의 데이터 처리, 대화형 분석 및 기계 학습을 지원하는 빅 데이터 솔루션입니다. Amazon EMR은 사용자 지정된 아마존 Elastic Compute Cloud (Amazon EC2) 클러스터, 아마존 엘라스틱 쿠버네티스 서비스 (Amazon EKS) 또는 아마존 EMR 서버리스에서 실행할 수 있습니다. AWS Outposts
- Amazon Athena는 오픈 소스 프레임워크를 기반으로 구축된 서버리스 대화형 분석 서비스입니다. 오픈 테이블 및 파일 형식을 지원하며, 데이터가 있는 곳에서 페타바이트 규모의 데이터를 분석할 수 있는 단순하고 유연한 방법을 제공합니다. Athena는 Iceberg에 대한 읽기, 시간 여행, 쓰기 및 DDL 쿼리를 기본적으로 지원하며 Iceberg AWS Glue Data Catalog 메타스토어에도 이를 사용합니다.
- Amazon Redshift는 클러스터 기반 및 서버리스 배포 옵션을 모두 지원하는 페타바이트 규모의 클라우드 데이터 웨어하우스입니다. Amazon Redshift Spectrum은 Amazon S3에 등록되어 있고 Amazon S3에 저장된 외부 테이블을 쿼리할 수 있습니다. AWS Glue Data Catalog Redshift Spectrum은 또한 아이스버그 스토리지 형식을 지원합니다.

- AWS Glue분석, 기계 학습 (ML) 및 애플리케이션 개발을 위해 여러 소스의 데이터를 더 쉽게 검색, 준비, 이동 및 통합할 수 있는 서버리스 데이터 통합 서비스입니다. AWS Glue 3.0 이상 버전은 데이터 레이크에 대한 Iceberg 프레임워크를 지원합니다. 를 AWS Glue 사용하여 Amazon S3의 Iceberg 테이블에서 읽기 및 쓰기 작업을 수행하거나 를 사용하여 Iceberg 테이블로 작업할 수 있습니다. AWS Glue Data Catalog삽입, 업데이트, Spark 쿼리, Spark 쓰기와 같은 추가 작업도 지원됩니다.
- AWS Glue Data CatalogIceberg 테이블을 지원하는 Hive 메타스토어 호환 데이터 카탈로그 서비스를 제공합니다.
- AWS Glue 크롤러에 Iceberg 테이블을 등록하기 위한 자동화를 제공합니다. AWS Glue Data Catalog
- Amazon은 Iceberg 형식을 사용하여 Amazon SageMaker 기능 스토어에 기능 세트를 저장할 수 있도록 SageMaker 지원합니다.
- AWS Lake FormationAthena 또는 Amazon Redshift에서 사용하는 Iceberg 테이블을 포함하여 데이터에 액세스할 수 있는 대략적이고 세분화된 액세스 제어 권한을 제공합니다. Iceberg 테이블의 권한 지원에 대해 자세히 알아보려면 Lake Formation 설명서를 참조하십시오.

AWS 에는 Iceberg를 지원하는 다양한 서비스가 있지만 이러한 서비스를 모두 다루는 것은 이 가이드의 범위를 벗어납니다. 다음 섹션에서는 Amazon EMR 및 Amazon Athena SQL의 Spark (배치 AWS Glue및 구조화된 스트리밍) 에 대해 다룹니다. 다음 섹션에서는 Athena SQL의 아이스버그 지원에 대한 간략한 설명을 제공합니다.

Amazon Athena SQL에서 아파치 아이스버그 테이블 시작하기

Amazon Athena는 아파치 아이스버그에 대한 기본 지원을 제공합니다. Athena 설명서의 시작하기 섹션에 자세히 설명된 서비스 사전 요구 사항을 설정하는 것 외에는 추가 단계나 구성 없이 Iceberg를 사용할 수 있습니다. 이 섹션에서는 Athena에서 테이블을 생성하는 방법을 간략하게 소개합니다. 자세한 내용은 이 가이드의 뒷부분에 있는 Athena SQL을 사용한 Apache Iceberg 테이블 사용을 참조하십시오.

다양한 엔진을 사용하여 Iceberg 테이블을 생성할 AWS 수 있습니다. 이러한 테이블은 여러 곳에서 원활하게 작동합니다. AWS 서비스 Athena SQL로 첫 번째 Iceberg 테이블을 만들려면 다음 상용구 코드를 사용할 수 있습니다.

```
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'
)
```

다음 섹션에서는 Athena에서 분할된 Iceberg 테이블과 분할되지 않은 Iceberg 테이블을 만드는 예를 제공합니다. 자세한 내용은 Athena 설명서에 자세히 설명된 Iceberg 구문을 참조하십시오.

파티션을 나누지 않은 테이블 만들기

다음 예제 명령문은 보일러플레이트 SQL 코드를 사용자 지정하여 Athena에 분할되지 않은 Iceberg 테이블을 생성합니다. Athena 콘솔의 쿼리 편집기에 이 명령문을 추가하여 테이블을 생성할 수 있습니다.

```
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'
)
```

쿼리 편집기 사용에 대한 step-by-step 지침은 Athena 설명서의 시작하기를 참조하십시오.

파티셔닝된 테이블 생성

다음 명령문은 Iceberg의 숨겨진 파티션 개념을 사용하여 날짜를 기준으로 파티션을 나눈 테이블을 만듭니다. day()변환을 통해 dd-mm-yyyy 형식을 사용하여 타임스탬프 열에서 일별 파티션을 추출합니다. Iceberg는 이 값을 데이터셋의 새 열로 저장하지 않습니다. 대신 데이터를 쓰거나 쿼리할 때 값이 즉석에서 파생됩니다.

```
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'
)
```

단일 CTAS 문으로 테이블 생성 및 데이터 로드

이전 섹션의 파티션을 나눈 예제와 파티션을 나누지 않은 예제에서 Iceberg 테이블은 빈 테이블로 생성됩니다. or 문을 사용하여 테이블에 데이터를 로드할 수 있습니다. INSERT MERGE 또는 CREATE TABLE AS SELECT (CTAS) 명령문을 사용하여 한 번에 데이터를 만들고 Iceberg 테이블에 로드할 수 있습니다.

CTAS는 Athena에서 테이블을 생성하고 단일 명령문으로 데이터를 로드하는 가장 좋은 방법입니다. 다음 예제는 CTAS를 사용하여 Athena의 기존 Hive/Parquet 테이블 (iceberg_ctas_table) 에서 Iceberg 테이블 () 을 생성하는 방법을 보여줍니다. 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
```

CTAS에 대한 자세한 내용은 Athena CTAS 설명서를 참조하십시오.

데이터 삽입, 업데이트 및 삭제

Athena는INSERT INTO, UPDATEMERGE INTO, 및 DELETE FRO M 문을 사용하여 Iceberg 테이블에 데이터를 쓰는 다양한 방법을 지원합니다.

참고: UPDATEMERGE INTO, 그리고 위치 삭제와 함께 이 merge-on-read 접근 방식을 DELETE FROM 사용하세요. 이 copy-on-write 접근 방식은 현재 Athena SQL에서 지원되지 않습니다.

예를 들어 다음 명령문은 Iceberg 테이블에 데이터를 INSERT INTO 추가하는 데 사용합니다.

```
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;
```

샘플 출력:

자세한 내용은 Athena 설명서를 참조하십시오.

아이스버그 테이블 쿼리

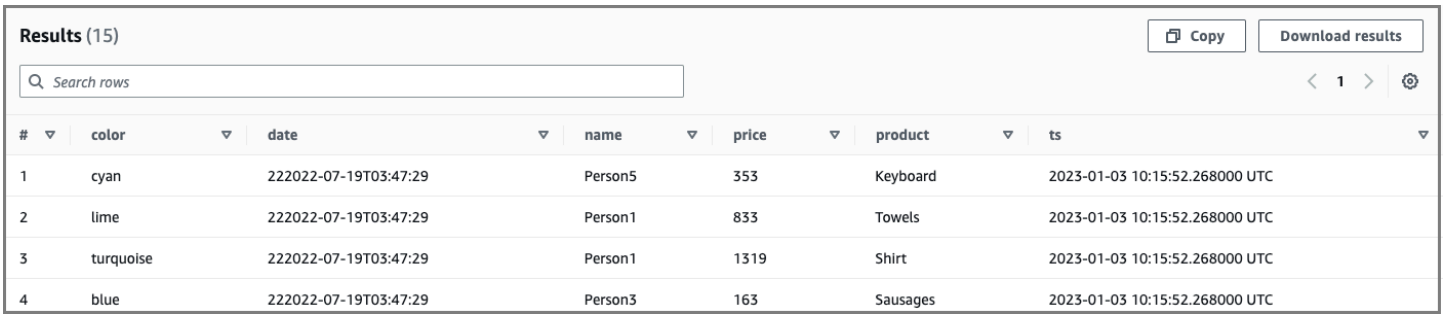
이전 예제에서 설명한 것처럼 Athena SQL을 사용하여 Iceberg 테이블에 대해 일반 SQL 쿼리를 실행할 수 있습니다.

Athena는 일반적인 쿼리 외에도 Iceberg 테이블에 대한 시간 여행 쿼리를 지원합니다. 앞서 설명했듯이 Iceberg 테이블에서 업데이트나 삭제를 통해 기존 레코드를 변경할 수 있으므로 시간 여행 쿼리를 사용하여 타임스탬프 또는 스냅샷 ID를 기반으로 테이블의 이전 버전을 찾아보는 것이 편리합니다.

예를 들어 다음 명령문은 색상 값을 업데이트한 다음 Person5 2023년 1월 4일의 이전 값을 표시합니다.

```
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'
```

샘플 출력:



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

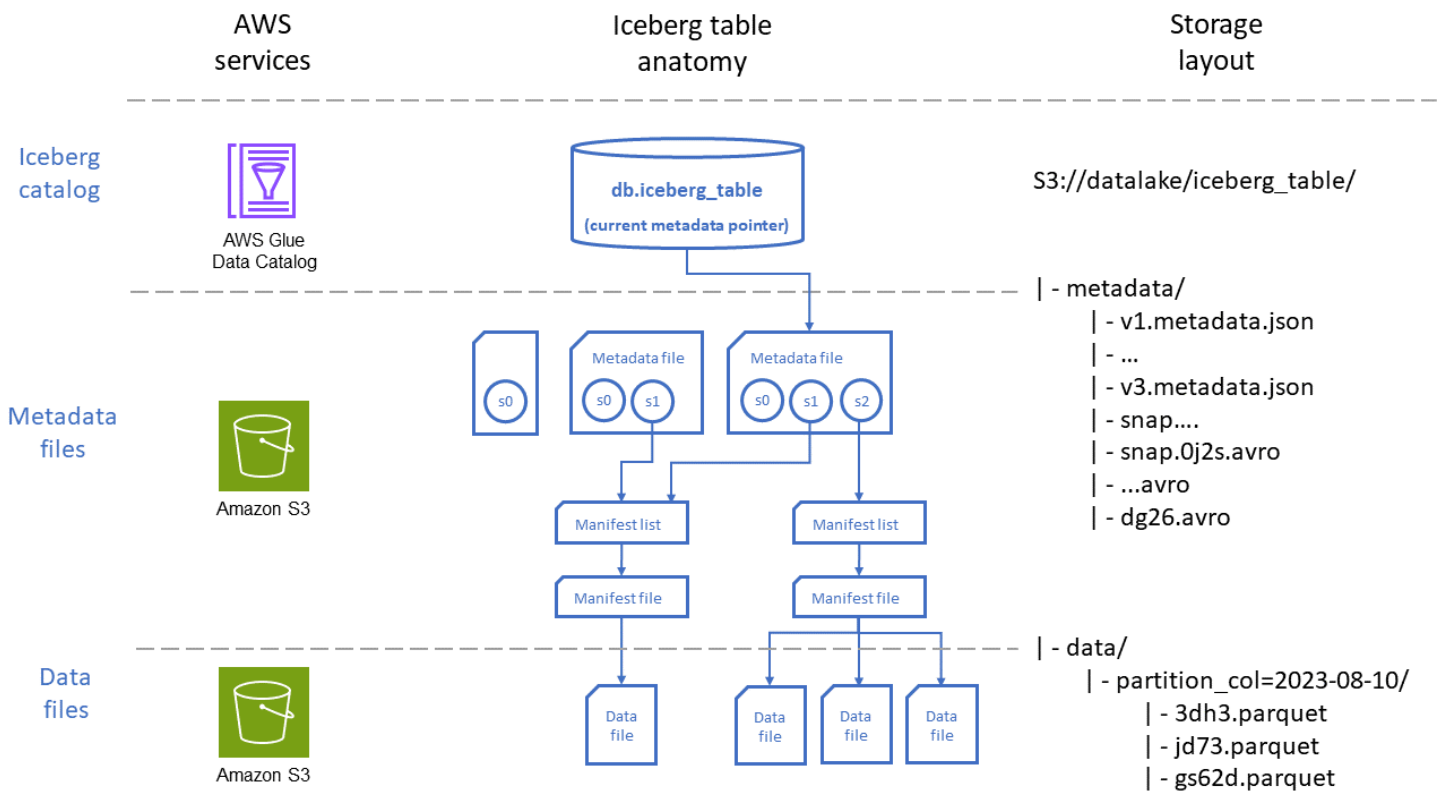
시간 여행 쿼리의 구문 및 추가 예는 Athena 설명서를 참조하십시오.

아이스버그 테이블 해부학

이제 Iceberg 테이블을 사용하는 기본 단계를 다루었으므로 Iceberg 테이블의 복잡한 세부 사항과 디자인에 대해 더 자세히 살펴보겠습니다.

이 가이드의 앞부분에서 설명한 기능을 활성화하기 위해 Iceberg는 데이터와 메타데이터 파일의 계층적 계층으로 설계되었습니다. 이러한 계층은 메타데이터를 지능적으로 관리하여 쿼리 계획 및 실행을 최적화합니다.

다음 다이어그램은 테이블을 저장하는 AWS 서비스 데 사용되는 위치와 Amazon S3의 파일 배치라는 두 가지 관점을 통해 Iceberg 테이블의 구성을 보여줍니다.



다이어그램에서 볼 수 있듯이 빙산 테이블은 세 가지 주요 레이어로 구성됩니다.

- Iceberg 카탈로그: Iceberg와 기본적으로 AWS Glue Data Catalog 통합되며 대부분의 사용 사례에서 실행되는 워크로드에 가장 적합한 옵션입니다. AWS Iceberg 테이블과 상호 작용하는 서비스 (예: Athena) 는 카탈로그를 사용하여 테이블의 현재 스냅샷 버전을 찾아 데이터를 읽거나 씁니다.
- 메타데이터 계층: 매니페스트 파일 및 매니페스트 목록 파일과 같은 메타데이터 파일은 테이블의 스키마, 파티션 전략, 데이터 파일의 위치와 같은 정보와 각 데이터 파일에 저장된 레코드의 최소 및 최대 범위와 같은 열 수준 통계를 추적합니다. 이러한 메타데이터 파일은 Amazon S3의 테이블 경로에 저장됩니다.
  - 매니페스트 파일에는 위치, 형식, 크기, 체크섬 및 기타 관련 정보를 포함하여 각 데이터 파일에 대한 기록이 들어 있습니다.
  - 매니페스트 목록은 매니페스트 파일의 색인을 제공합니다. 테이블의 매니페스트 파일 수가 늘어남에 따라 해당 정보를 더 작은 하위 섹션으로 나누면 쿼리로 스캔해야 하는 매니페스트 파일 수를 줄일 수 있습니다.
  - 메타데이터 파일에는 매니페스트 목록, 스키마, 파티션 메타데이터, 스냅샷 파일 및 테이블의 메타데이터를 관리하는 데 사용되는 기타 파일을 포함하여 전체 Iceberg 테이블에 대한 정보가 포함됩니다.

- 데이터 레이어: 이 레이어에는 쿼리를 실행할 데이터 레코드가 있는 파일이 포함됩니다. 이러한 파일은 아파치 파켓, 아파치 아브로, 아파치 ORC 등 다양한 형식으로 저장할 수 있습니다.
  - 데이터 파일에는 테이블의 데이터 레코드가 들어 있습니다.
  - 파일 삭제는 Iceberg 테이블에서 행 수준 삭제 및 업데이트 작업을 인코딩합니다. Iceberg에는 Iceberg 설명서에 설명된 대로 두 가지 유형의 삭제 파일이 있습니다. 이러한 파일은 모드를 사용한 조작으로 생성됩니다. merge-on-read

Amazon EMR에서 아파치 아이스버그와 함께 작업하기

Amazon EMR은 Apache Spark, Apache Hive, Flink 및 Trino와 같은 오픈 소스 프레임워크를 사용하여 클라우드에서 페타바이트 규모의 데이터 처리, 대화형 분석 및 기계 학습을 제공합니다.

> **Note**
>
> 이 가이드에서는 아파치 스파크를 예로 사용합니다.

Amazon EMR은 여러 배포 옵션을 지원합니다. 예를 들어, Amazon EC2의 Amazon EMR, Amazon EC2의 Amazon EMR, Amazon EKS의 Amazon EMR, 서버리스 및 Amazon EMR을 지원합니다. AWS Outposts워크로드에 맞는 배포 옵션을 선택하려면 Amazon EMR FAQ를 참조하십시오.

버전 및 기능 호환성

Amazon EMR 버전 6.5.0 이상 버전은 기본적으로 아파치 아이스버그를 지원합니다. 각 Amazon EMR 릴리스에 지원되는 Iceberg 버전 목록은 Amazon EMR 설명서의 Iceberg 출시 기록을 참조하십시오. 또한 Amazon EMR에서 Iceberg를 사용하기 위한 고려 사항 및 제한 사항을 검토하여 다양한 프레임워크의 Amazon EMR에서 지원되는 Iceberg 기능을 확인하십시오.

지원되는 최신 Iceberg 버전을 활용하려면 최신 Amazon EMR 버전을 사용하는 것이 좋습니다. 이 섹션의 코드 예제 및 구성은 Amazon EMR 릴리스 emr-6.9.0을 사용하고 있다고 가정합니다.

아이스버그로 Amazon EMR 클러스터 만들기

아이스버그가 설치된 상태에서 Amazon EC2에 Amazon EMR 클러스터를 생성하려면 Amazon EMR 설명서의 지침을 따르십시오.

특히 클러스터는 다음과 같은 분류로 구성되어야 합니다.

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

또한 Amazon EMR 6.6.0부터 시작하는 아이스버그 워크로드의 배포 옵션으로 Amazon EMR 서버리스 또는 Amazon EKS의 Amazon EMR을 사용하도록 선택할 수 있습니다.

Amazon EMR에서 아이스버그 애플리케이션 개발

Iceberg 애플리케이션용 Spark 코드를 개발하려면 Amazon EMR 클러스터에서 실행되는 완전 관리형 Jupyter 노트북을 위한 웹 기반 통합 개발 환경 (IDE) 인 Amazon EMR Studio를 사용할 수 있습니다.

Amazon EMR 스튜디오 노트북 사용

Amazon EMR 스튜디오 워크스페이스 노트북에서 Spark 애플리케이션을 대화형 방식으로 개발하고 해당 노트북을 Amazon EC2 클러스터의 Amazon EMR 또는 Amazon EKS 관리형 엔드포인트의 Amazon EMR에 연결할 수 있습니다. Amazon EC2에서 Amazon EMR용 EMR 스튜디오를 설정하고 Amazon EKS에서 Amazon EMR을 설정하는 방법에 대한 지침은 AWS 서비스 설명서를 참조하십시오.

EMR Studio에서 Iceberg를 사용하려면 다음 단계를 따르십시오.

1. Iceberg가 설치된 클러스터 사용에 나와 있는 지침에 따라 Iceberg를 활성화한 상태에서 Amazon EMR 클러스터를 시작합니다.
2. EMR 스튜디오를 설정합니다. 지침은 Amazon EMR 스튜디오 설정을 참조하십시오.
3. EMR Studio Workspace 노트북을 열고 노트북의 첫 번째 셀로 다음 코드를 실행하여 Iceberg를 사용하도록 Spark 세션을 구성합니다.

```
%%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"
    }
}
```

여기서 각 항목은 다음과 같습니다.

- `<catalog_name>`아이스버그 스파크 세션 카탈로그 이름입니다. 카탈로그 이름으로 바꾸고, 이 카탈로그와 관련된 모든 구성에서 참조를 변경해야 한다는 점을 잊지 마세요. 그런 다음 코드에서 다음과 같이 Spark 세션 카탈로그 이름을 포함하여 정규화된 테이블 이름을 사용하여 Iceberg 테이블을 참조해야 합니다.

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

- `<catalog_name>.warehouse`데이터 및 메타데이터를 저장하려는 Amazon S3 경로를 가리킵니다.
- 카탈로그를 an으로 AWS Glue Data Catalog만들려면 `<catalog_name>.catalog-impl` 로 설정합니다`org.apache.iceberg.aws.glue.GlueCatalog`. 이 키는 모든 사용자 정의 카탈로그 구현을 위한 구현 클래스를 가리키는 데 필요합니다. 이 가이드 뒷부분의 일반 모범 사례 섹션에서는 Iceberg가 지원하는 다양한 카탈로그에 대해 설명합니다.
- 높은 병렬성을 `<catalog_name>.io-impl` 위해 Amazon S3 멀티파트 업로드를 활용하려면 `org.apache.iceberg.aws.s3.S3FileIO` as를 사용하십시오.

4. 이제 다른 Spark 애플리케이션과 마찬가지로 노트북에서 Iceberg용 Spark 애플리케이션을 대화형 방식으로 개발할 수 있습니다.

Amazon EMR Studio를 사용하여 Apache Iceberg용 Spark를 구성하는 방법에 대한 자세한 내용은 Amazon EMR에서 Apache Iceberg를 사용하여 Apache Iceberg를 사용하여 고성능, ACID 규정을 준수하는 진화하는 데이터 레이크 구축하기 블로그 게시물을 참조하십시오.

Amazon EMR에서 아이스버그 작업 실행

Iceberg 워크로드용 Spark 애플리케이션 코드를 개발한 후에는 Iceberg를 지원하는 모든 Amazon EMR 배포 옵션에서 실행할 수 있습니다 (Amazon EMR FAQ 참조).

다른 Spark 작업과 마찬가지로, 단계를 추가하거나 대화형 방식으로 Spark 작업을 마스터 노드에 제출하여 Amazon EC2 기반 Amazon EMR 클러스터에 작업을 제출할 수 있습니다. Spark 작업을 실행하려면 다음 Amazon EMR 설명서 페이지를 참조하십시오.

- Amazon EMR on Amazon EC2 클러스터에 작업을 제출하기 위한 다양한 옵션에 대한 개요와 각 옵션에 대한 자세한 지침은 클러스터에 작업 제출을 참조하십시오.
- Amazon EKS의 Amazon EMR에 대해서는 다음을 사용하여 Spark 작업 실행을 참조하십시오. StartJobRun
- Amazon EMR 서버리스의 경우 작업 실행을 참조하십시오.

다음 섹션은 각 Amazon EMR 배포 옵션의 예를 제공합니다.

아마존 EC2 기반 아마존 EMR

다음 단계를 사용하여 아이스버그 스파크 작업을 제출할 수 있습니다.

1. 워크스테이션에 다음 콘텐츠가 `emr_step_iceberg.json` 포함된 파일을 생성하십시오.

```
[{
    "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. 굵게 강조 표시된 Iceberg 구성 옵션을 사용자 지정하여 특정 Spark 작업에 대한 구성 파일을 수정하십시오.
3. () 를 사용하여 단계를 제출하십시오. AWS Command Line Interface AWS CLI`emr_step_iceberg.json`파일이 있는 디렉터리에서 명령을 실행합니다.

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

Amazon EMR Serverless

다음을 사용하여 Amazon EMR 서버리스에 아이스버그 스파크 작업을 제출하려면 AWS CLI

1. 워크스테이션에 다음 콘텐츠가 `emr_serverless_iceberg.json` 포함된 파일을 생성하십시오.

```
{
    "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. 굵게 강조 표시된 Iceberg 구성 옵션을 사용자 지정하여 특정 Spark 작업에 대한 구성 파일을 수정하십시오.
3. 를 사용하여 작업을 제출하십시오. AWS CLIemr_serverless_iceberg.json파일이 있는 디렉터리에서 명령을 실행합니다.

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

EMR 스튜디오 콘솔을 사용하여 Amazon EMR 서버리스에 아이스버그 스파크 작업을 제출하려면:

1. Amazon EMR 서버리스 설명서의 지침을 따르십시오.
2. Job 구성의 경우 에 제공된 Spark용 Iceberg 구성을 사용하고 Iceberg에 대해 강조 표시된 필드를 사용자 정의하십시오. AWS CLI 자세한 지침은 Amazon EMR 설명서에서 EMR 서버리스와 함께 아파치 아이스버그 사용을 참조하십시오.

아마존 EKS 기반 아마존 EMR

다음을 사용하여 Amazon EKS의 Amazon EMR에 아이스버그 스파크 작업을 제출하려면: AWS CLI

1. 워크스테이션에 다음 콘텐츠가 emr_eks_iceberg.json 포함된 파일을 생성하십시오.

```
{
    "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. 굵게 강조 표시된 Iceberg 구성 옵션을 사용자 지정하여 Spark 작업의 구성 파일을 수정하십시오.
3. 를 사용하여 작업을 제출하십시오. AWS CLIemr_eks_iceberg.json파일이 있는 디렉터리에서 다음 명령을 실행합니다.

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

자세한 지침은 EKS용 Amazon EMR 설명서에서 EKS에서 Amazon EMR과 함께 아파치 아이스버그 사용하기 섹션을 참조하십시오.

아마존 EMR 모범 사례

이 섹션에서는 Iceberg 테이블에 대한 데이터 읽기 및 쓰기를 최적화하기 위해 Amazon EMR에서 Spark 작업을 조정하는 일반적인 지침을 제공합니다. Iceberg별 모범 사례는 이 가이드 뒷부분의 모범 사례 섹션을 참조하십시오.

- 최신 버전의 Amazon EMR 사용 — Amazon EMR은 Amazon EMR Spark 런타임과 함께 즉시 사용할 수 있는 Spark 최적화 기능을 제공합니다. AWS 새 릴리스가 출시될 때마다 Spark 런타임 엔진의 성능이 개선됩니다.

- Spark 워크로드를 위한 최적의 인프라 결정 — Spark 워크로드의 최적 성능을 보장하기 위해 작업 특성에 따라 다른 유형의 하드웨어가 필요할 수 있습니다. Amazon EMR은 모든 유형의 처리 요구 사항을 충족할 수 있도록 여러 인스턴스 유형 (예: 컴퓨팅 최적화, 메모리 최적화, 범용, 스토리지 최적화) 을 지원합니다. 새 워크로드를 온보딩할 때는 M5 또는 M6g와 같은 일반 인스턴스 유형으로 벤치마킹하는 것이 좋습니다. Ganglia와 Amazon의 운영 체제 (OS) 및 YARN 메트릭을 CloudWatch 모니터링하여 최대 부하 시 시스템 병목 현상 (CPU, 메모리, 스토리지 및 I/O) 을 파악하고 적절한 하드웨어를 선택합니다.
- 조정 `spark.sql.shuffle.partitions` — `spark.sql.shuffle.partitions` 속성을 클러스터의 총 가상 코어 (vCore) 수 또는 해당 값의 배수 (일반적으로 총 vCore 수의 1~2배) 로 설정합니다. 이 설정은 해시 및 범위 파티셔닝을 쓰기 분산 모드로 사용할 때 Spark의 병렬 처리에 영향을 줍니다. 데이터를 정리하기 위해 쓰기 전에 셔플을 요청하므로 파티션 정렬이 보장됩니다.
- 관리형 규모 조정 활성화 - 거의 모든 사용 사례에서 관리형 규모 조정 및 동적 할당을 활성화하는 것이 좋습니다. 하지만 예측 가능한 패턴을 가진 워크로드가 있는 경우 자동 조정 및 동적 할당을 비활성화하는 것이 좋습니다. 관리형 크기 조정이 활성화된 경우 스팟 인스턴스를 사용하여 비용을 절감하는 것이 좋습니다. 코어 또는 마스터 노드 대신 스팟 인스턴스를 태스크 노드에 사용하십시오. 스팟 인스턴스를 사용할 때는 플릿당 여러 인스턴스 유형이 있는 인스턴스 플릿을 사용하여 스팟 가용성을 보장하십시오.
- 가능하면 브로드캐스트 조인 사용 — 브로드캐스트 (맵사이드) 조인이 가장 최적의 조인입니다. 단, 테이블 중 하나가 가장 작은 노드의 메모리 (MB 단위) 에 들어갈 만큼 작고 equi (=) 조인을 수행하는 경우에 한합니다. 전체 외부 조인을 제외한 모든 조인 유형이 지원됩니다. 브로드캐스트 조인은 메모리의 모든 작업자 노드에서 작은 테이블을 해시 테이블로 브로드캐스트합니다. 스몰 테이블이 브로드캐스트된 후에는 테이블을 변경할 수 없습니다. 해시 테이블은 Java Virtual Machine (JVM) 에 로컬로 저장되므로 해시 조인을 사용하여 조인 조건에 따라 대형 테이블과 쉽게 병합할 수 있습니다. 브로드캐스트 조인은 셔플 오버헤드가 최소화되므로 높은 성능을 제공합니다.
- 가비지 컬렉터 조정 — 가비지 컬렉션 (GC) 주기가 느린 경우 성능 향상을 위해 기본 Parallel 가비지 컬렉터에서 G1GC로 전환하는 것을 고려해 보세요. GC 성능을 최적화하기 위해 GC 파라미터를 미세 조정할 수 있습니다. GC 성능을 추적하려면 Spark UI를 사용하여 GC 성능을 모니터링할 수 있습니다. 이상적으로는 GC 시간이 전체 작업 런타임의 1% 미만이어야 합니다.

에서 아파치 아이스버그와 함께 작업하기 AWS Glue

AWS Glue분석, 기계 학습 (ML) 및 애플리케이션 개발을 위해 여러 소스의 데이터를 더 쉽게 검색, 준비, 이동 및 통합할 수 있는 서버리스 데이터 통합 서비스입니다. 의 핵심 기능 중 AWS Glue 하나는 ETL (추출, 변환, 로드) 작업을 간단하고 비용 효율적인 방식으로 수행할 수 있다는 것입니다. 이를 통해 데이터를 분류하고, 정리하고, 보강하고, 다양한 데이터 저장소와 데이터 스트림 간에 안정적으로 이동할 수 있습니다.

AWS Glue 작업은 Apache Spark 또는 Python 런타임을 사용하여 변환 로직을 정의하는 스크립트를 캡슐화합니다. AWS Glue 작업은 배치 모드와 스트리밍 모드 모두에서 실행할 수 있습니다.

에서 AWS Glue Iceberg 작업을 생성할 때 버전에 따라 네이티브 Iceberg 통합 또는 사용자 지정 Iceberg 버전을 사용하여 Iceberg 종속성을 작업에 연결할 수 있습니다. AWS Glue

네이티브 아이스버그 통합 사용

AWS Glue 버전 3.0과 4.0은 스파크용 아파치 아이스버그, 아파치 후디, 리눅스 파운데이션 델타 레이크와 같은 트랜잭션 데이터 레이크 형식을 기본적으로 지원합니다. AWS Glue 이 통합 기능은 에서 이러한 프레임워크를 사용하기 시작하는 데 필요한 구성 단계를 단순화합니다. AWS Glue

AWS Glue 작업에 대한 Iceberg 지원을 활성화하려면 작업을 설정합니다. 작업에 대한 Job details 탭을 선택하고 고급 속성에서 AWS Glue Job parameters (작업 매개 변수) 로 `--datalake-formats` 스크롤한 다음 키와 값을 로 설정합니다. `iceberg`

노트북을 사용하여 작업을 작성하는 경우 다음과 같이 `%%configure` 마법을 사용하여 첫 번째 노트북 셀에서 매개변수를 구성할 수 있습니다.

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

사용자 지정 Iceberg 버전 사용

경우에 따라 해당 작업에 사용할 Iceberg 버전을 계속 관리하면서 원하는 속도에 맞게 업그레이드해야 할 수도 있습니다. 예를 들어, 최신 버전으로 업그레이드하면 새로운 기능과 성능 향상을 이용할 수 있

습니다. 특정 Iceberg 버전을 함께 AWS Glue사용하려면 사용자 지정 커넥터 또는 자체 JAR 파일을 사용할 수 있습니다.

커스텀 커넥터 사용

AWS Glue 커넥터를 지원합니다. 커넥터는 데이터 저장소에 액세스하는 데 도움이 되는 선택적 코드 패키지입니다 AWS Glue Studio. 에서 AWS Marketplace커넥터를 구독하거나 사용자 지정 커넥터를 만들 수 있습니다.

> **Note**
>
> AWS Marketplace 에 대한 Apache Iceberg 커넥터를 제공합니다. AWS Glue하지만 Iceberg 버전에 대한 제어를 유지하려면 대신 사용자 지정 커넥터를 사용하는 것이 좋습니다.

예를 들어 Iceberg 버전 0.13.1용 고객 커넥터를 생성하려면 다음 단계를 따르십시오.

1. 파일 `iceberg-spark-runtime-3.1_2.12-0.13.1.jarbundle-2.17.161.jar`, 을 (를) Amazon S3 `url-connection-client-2.17.161.jar` 버킷에 업로드합니다. 해당 Apache Maven 리포지토리에서 이러한 파일을 다운로드할 수 있습니다.
2. AWS Glue Studio 콘솔에서 사용자 지정 Spark 커넥터를 생성합니다.
   a. 탐색 창에서 데이터 연결을 선택합니다. (이전 내비게이션을 사용하는 경우 커넥터, 사용자 지정 커넥터 생성을 선택합니다.)
   b. 커넥터 상자에서 사용자 지정 커넥터 만들기를 선택합니다.
   c. 사용자 지정 커넥터 만들기 페이지에서:
      - Amazon S3에 있는 JAR 파일의 경로를 지정합니다.
      - 커넥터 이름을 입력합니다.
      - 커넥터 유형으로 Spark를 선택합니다.
      - 클래스 이름에는 연산자와 함께 Spark 데이터 소스를 로드할 때 사용하는 정규화된 데이터 소스 클래스 이름 (또는 해당 별칭) 을 지정합니다. `format`
      - (선택 사항) 커넥터에 대한 설명을 제공하십시오.
3. [커넥터 생성(Create connector)]을 선택합니다.

에서 AWS Glue커넥터로 작업할 때는 커넥터에 대한 연결을 생성해야 합니다. 연결에는 특정 데이터 저장소에 연결하는 데 필요한 속성이 포함되어 있습니다. ETL 작업에서 데이터 원본 및 데이터 대상과의 연결을 사용합니다. 커넥터와 연결은 함께 작동하여 데이터 스토어에 쉽게 액세스할 수 있습니다.

생성한 사용자 지정 Iceberg 커넥터를 사용하여 연결을 만들려면:

1. AWS Glue Studio 콘솔에서 사용자 지정 Iceberg 커넥터를 선택합니다.
2. 안내에 따라 VPC 및 작업에 필요한 기타 네트워크 구성과 같은 세부 정보를 제공한 다음 Create connection (연결 생성) 을 선택합니다.

이제 AWS Glue ETL 작업에서 연결을 사용할 수 있습니다. 작업을 생성하는 방법에 따라 작업에 연결을 연결하는 다양한 방법이 있습니다.

- 를 사용하여 AWS Glue Studio시각적 작업을 만드는 경우 데이터 원본 속성 — 커넥터 탭의 연결 목록에서 연결을 선택할 수 있습니다.
- 노트북에서 작업을 개발하는 경우 `%connections` 마법을 사용하여 연결 이름을 설정하십시오.

```
%glue_version 3.0

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

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

- 스크립트 편집기를 사용하여 작업을 작성하는 경우 [작업 세부 정보] 탭의 [고급 속성], [추가 네트워크 연결] 에서 연결을 지정합니다.

이 섹션의 절차에 대한 자세한 내용은 AWS Glue 설명서의 커넥터 및 연결 AWS Glue Studio사용을 참조하십시오.

자체 JAR 파일 가져오기

AWS Glue에서는 커넥터를 사용하지 않고도 Iceberg와 함께 작업할 수 있습니다. 이 방법은 Iceberg 버전에 대한 제어를 유지하고 신속하게 업데이트하려는 경우에 유용합니다. 이 옵션을 사용하려면 필요한 Iceberg JAR 파일을 선택한 S3 버킷에 업로드하고 작업에서 파일을 참조해야 AWS Glue 합니다. 예를 들어, Iceberg 1.0.0을 사용하는 경우 필수 JAR 파일은`iceberg-`

spark-runtime-3.0_2.12-1.0.0.jar,, 입니다. url-connection-client-2.15.40.jar bundle-2.15.40.jar 작업의 --user-jars-first 매개변수를 로 설정하여 클래스 경로에 있는 추가 JAR 파일의 우선 순위를 지정할 수도 있습니다. true

아이스버그의 스파크 컨피그레이션은 AWS Glue

이 섹션에서는 Iceberg 데이터세트의 AWS Glue ETL 작업을 작성하는 데 필요한 Spark 구성에 대해 설명합니다. 모든 Spark 구성 키 및 값을 쉼표로 구분된 목록과 함께 --conf Spark 키와 함께 사용하여 이러한 구성을 설정할 수 있습니다. 노트북이나 AWS Glue Studio 콘솔의 Job parameters 섹션에서 %%configure 마법을 사용할 수 있습니다.

```
%glue_version 3.0

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

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

다음 속성을 사용하여 Spark 세션을 구성하십시오.

- <catalog_name>아이스버그 스파크 세션 카탈로그 이름입니다. 카탈로그 이름으로 바꾸고, 이 카탈로그와 관련된 모든 구성에서 참조를 변경해야 한다는 점을 잊지 마세요. 그런 다음 코드에서 다음과 같이 Spark 세션 카탈로그 이름을 포함하여 정규화된 테이블 이름을 사용하여 Iceberg 테이블을 참조해야 합니다. <catalog_name>.<database_name>.<table_name>
- <catalog_name>.<warehouse>데이터 및 메타데이터를 저장하려는 Amazon S3 경로를 가리킵니다.
- 카탈로그를 an으로 AWS Glue Data Catalog만들려면 <catalog_name>.catalog-impl 로 설정합니다org.apache.iceberg.aws.glue.GlueCatalog. 이 키는 모든 사용자 정의 카탈로그 구현을 위한 구현 클래스를 가리키는 데 필요합니다. Iceberg에서 지원하는 카탈로그는 이 가이드 뒷부분의 일반 모범 사례 ??? 섹션을 참조하십시오.
- 높은 병렬성을 <catalog_name>.io-impl 위해 Amazon S3 멀티파트 업로드를 활용하려면 org.apache.iceberg.aws.s3.S3FileIO as를 사용하십시오.

예를 들어glue_iceberg, 라는 카탈로그가 있는 경우 다음과 같이 여러 --conf 키를 사용하여 작업을 구성할 수 있습니다.

```
%%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
}
```

또는 다음과 같이 코드를 사용하여 위의 구성을 Spark 스크립트에 추가할 수 있습니다.

```
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()
```

작업 모범 AWS Glue 사례

이 섹션에서는 Iceberg 테이블에 대한 데이터 읽기 및 쓰기를 AWS Glue 최적화하기 위해 Spark 작업을 조정하는 일반적인 지침을 제공합니다. Iceberg별 모범 사례는 이 가이드 뒷부분의 모범 사례 섹션을 참조하십시오.

- 최신 버전의 AWS Glue를 사용하고 가능하면 업그레이드하십시오. 새 버전은 성능 향상, 시작 시간 단축 및 새로운 기능을 AWS Glue 제공합니다. 또한 최신 Iceberg 버전에 필요할 수 있는 최신 Spark 버전도 지원합니다. 사용 가능한 AWS Glue 버전 및 지원되는 Spark 버전 목록은 설명서를 참조하십시오.AWS Glue
- AWS Glue 작업 메모리 최적화 - AWS 블로그 게시물 메모리 관리 최적화의 권장 사항을 따르십시오. AWS Glue

- AWS Glue Auto Scaling 사용 — Auto Scaling을 활성화하면 작업량에 따라 AWS Glue 작업자 수를 동적으로 AWS Glue 자동으로 조정합니다. 이렇게 하면 AWS Glue 작업량이 적고 작업자가 유휴 상태일 때 작업자 수를 줄이므로 AWS Glue 최대 부하 시 작업 비용을 줄이는 데 도움이 됩니다. AWS Glue Auto Scaling을 사용하려면 AWS Glue 작업을 확장할 수 있는 최대 작업자 수를 지정해야 합니다. 자세한 내용은AWS GlueAWS Glue 설명서의 Auto Scaling 사용을 참조하십시오.
- 사용자 지정 커넥터를 사용하거나 라이브러리 종속성을 추가하세요. Iceberg를 시작하려면 Iceberg를 AWS Glue 기본적으로 통합하는 것이 가장 좋습니다. 하지만 프로덕션 워크로드의 경우 Iceberg 버전을 완벽하게 제어하려면 사용자 지정 컨테이너를 사용하거나 라이브러리 종속성을 추가 (이 가이드의 앞부분에서 설명) 하는 것이 좋습니다. 이 접근 방식을 통해 최신 Iceberg 기능을 활용하고 작업 성능을 개선할 수 있습니다. AWS Glue
- 모니터링 및 디버깅을 위한 Spark UI 활성화 - Spark UI를 사용하여 Spark 작업의 여러 단계를 방향성 비순환 그래프 (DAG) 로 시각화하고 작업을 자세히 모니터링하여 Iceberg 작업을 검사할 수도 있습니다. AWS Glue Spark UI는 Iceberg 작업의 문제를 해결하고 최적화할 수 있는 효과적인 방법을 제공합니다. 예를 들어 셔플이 크거나 디스크 유출이 발생하는 병목 단계를 식별하여 튜닝 기회를 식별할 수 있습니다. 자세한 내용은 설명서에서 Apache Spark 웹 UI를 사용한 작업 모니터링을 참조하십시오. AWS Glue

Apache Spark를 사용하여 아파치 아이스버그 테이블 작업하기

이 섹션에서는 Apache Spark를 사용하여 Iceberg 테이블과 상호 작용하는 방법을 간략하게 설명합니다. Amazon EMR 또는 에서 실행할 수 있는 상용구 코드를 예로 들 수 있습니다. AWS Glue

참고: Iceberg 테이블과 상호 작용하기 위한 기본 인터페이스는 SQL이므로 대부분의 예제는 Spark SQL을 API와 결합합니다. DataFrames

아이스버그 테이블 생성 및 작성

Spark SQL과 Spark를 사용하여 Iceberg 테이블에 데이터를 생성하고 DataFrames 추가할 수 있습니다.

스파크 SQL 사용

아이스버그 데이터세트를 작성하려면 및 와 같은 표준 Spark SQL 문을 사용하세요. `CREATE TABLE` `INSERT INTO`

파티션을 나누지 않은 테이블

다음은 Spark SQL을 사용하여 파티션을 나누지 않은 Iceberg 테이블을 만드는 예시입니다.

```
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')
""")
```

분할되지 않은 테이블에 데이터를 삽입하려면 표준 명령문을 사용하세요. `INSERT INTO`

```
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
""")
```

분할된 테이블

다음은 Spark SQL을 사용하여 파티션을 나눈 Iceberg 테이블을 만드는 예제입니다.

```
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')
""")
```

Spark SQL을 사용하여 분할된 Iceberg 테이블에 데이터를 삽입하려면 글로벌 정렬을 수행한 다음 데이터를 작성합니다.

```
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
""")
```

API 사용 DataFrames

Iceberg 데이터세트를 작성하려면 API를 사용할 수 있습니다. `DataFrameWriterV2`

Iceberg 테이블을 만들고 여기에 데이터를 쓰려면 `df.writeTo( t)` 함수를 사용하세요. 테이블이 있으면 `.append()` 함수를 사용하십시오. 그렇지 않은 경우 다음 예제에서 use `.create()`. `.createOrReplace()` 를 사용하세요. 이 예제는 다음과 같은 변형입니다`CREATE OR REPLACE TABLE AS SELECT`. `.create()`

파티션을 나누지 않은 테이블

API를 사용하여 파티션을 나누지 않은 Iceberg 테이블을 만들고 채우려면: `DataFrameWriterV2`

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

API를 사용하여 기존의 분할되지 않은 Iceberg 테이블에 데이터를 삽입하려면: `DataFrameWriterV2`

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

분할된 테이블

`DataFrameWriterV2`API를 사용하여 분할된 Iceberg 테이블을 만들고 채우려면 로컬 정렬을 사용하여 데이터를 수집할 수 있습니다.

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

`DataFrameWriterV2`API를 사용하여 분할된 Iceberg 테이블에 데이터를 삽입하려면 글로벌 정렬을 사용하여 데이터를 수집할 수 있습니다.

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

Iceberg 테이블의 데이터 업데이트

다음 예제는 Iceberg 테이블의 데이터를 업데이트하는 방법을 보여줍니다. 이 예제에서는 열에 짝수인 모든 행을 수정합니다`c_customer_sk`.

```
spark.sql(f"""
UPDATE {CATALOG_NAME}.{db.name}.{table.name}
```

```
SET c_email_address = 'even_row'
WHERE c_customer_sk % 2 == 0
""")
```

이 작업은 기본 copy-on-write 전략을 사용하므로 영향을 받는 모든 데이터 파일을 다시 작성합니다.

Iceberg 테이블의 데이터 업데이트

데이터를 업데이트하는 것은 단일 트랜잭션에서 새 데이터 기록을 삽입하고 기존 데이터 기록을 업데이트하는 것을 말합니다. Iceberg 테이블에 데이터를 삽입하려면 명령문을 사용합니다. SQL `MERGE INTO`

다음 예제에서는 테이블 내에 있는 테이블{UPSERT_TABLE_NAME} 의 내용을 보여줍니다. {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 *
""")
```

- 에 있는 고객 기록이 동일한 {TABLE_NAME} 기록과 함께 {UPSERT_TABLE_NAME} 이미 존재하는 경우c_customer_id, {UPSERT_TABLE_NAME} 레코드 c_email_address 값이 기존 값보다 우선합니다 (업데이트 작업).
- 에 있는 고객 기록이 {UPSERT_TABLE_NAME} 없는 경우 해당 {UPSERT_TABLE_NAME} 기록이 {TABLE_NAME} (삽입 작업) 에 추가됩니다. {TABLE_NAME}

Iceberg 테이블의 데이터 삭제

Iceberg 테이블에서 데이터를 삭제하려면 `DELETE FROM` 표현식을 사용하고 삭제할 행과 일치하는 필터를 지정하십시오.

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

필터가 전체 파티션과 일치하면 Iceberg는 메타데이터만 삭제하고 데이터 파일은 그대로 둡니다. 그렇지 않으면 영향을 받은 데이터 파일만 다시 씁니다.

삭제 메서드는 WHERE 조항의 영향을 받는 데이터 파일을 가져와서 삭제된 레코드 없이 데이터 파일의 사본을 만듭니다. 그런 다음 새 데이터 파일을 가리키는 새 테이블 스냅숏을 만듭니다. 따라서 삭제된 레코드는 테이블의 이전 스냅샷에 계속 남아 있습니다. 예를 들어 테이블의 이전 스냅샷을 검색하면 방금 삭제한 데이터가 표시됩니다. 정리를 위해 관련 데이터 파일이 있는 불필요한 오래된 스냅샷을 제거하는 방법에 대한 자세한 내용은 이 안내서의 뒷부분에 나오는 압축을 사용하여 파일 유지 관리 섹션을 참조하십시오.

데이터 읽기

Spark SQL과 Spark를 모두 사용하여 Spark에 있는 Iceberg 테이블의 최신 상태를 읽을 수 있습니다. DataFrames

스파크 SQL을 사용한 예시:

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

DataFrames API를 사용한 예시:

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

시간 여행 사용

Iceberg 테이블에서 쓰기 작업 (삽입, 업데이트, 업로드, 삭제) 을 수행할 때마다 새 스냅샷이 생성됩니다. 그런 다음 이 스냅샷을 시간 여행에 사용할 수 있습니다. 즉, 시간을 거슬러 올라가 과거의 테이블 상태를 확인할 수 있습니다.

값을 사용하고 snapshot-id 시간을 지정하여 테이블의 스냅샷 기록을 검색하는 방법에 대한 자세한 내용은 이 가이드 뒷부분에 있는 메타데이터 액세스 섹션을 참조하십시오.

다음 시간 여행 쿼리는 특정 snapshot-id 조건을 기반으로 테이블 상태를 표시합니다.

스파크 SQL 사용:

```
spark.sql(f"""
```

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

DataFrames API 사용:

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

다음 시간 여행 쿼리는 특정 타임스탬프 이전에 생성된 마지막 스냅샷을 기반으로 테이블의 상태를 밀리초 () `as-of-timestamp` 단위로 표시합니다.

Spark SQL 사용:

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

DataFrames API 사용:

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

증분 쿼리 사용

또한 Iceberg 스냅샷을 사용하여 첨부된 데이터를 점진적으로 읽을 수 있습니다.

참고: 현재 이 작업은 스냅샷에서 데이터를 읽는 것을 지원합니다. `append` `replace`, `overwrite` 또는 등의 작업에서 데이터를 가져오는 것은 지원되지 않습니다. `delete` 또한 Spark SQL 구문에서는 증분 읽기 작업이 지원되지 않습니다.

다음 예제는 스냅샷 `start-snapshot-id` (제외) 과 (포함) 사이의 Iceberg 테이블에 추가된 모든 레코드를 검색합니다. `end-snapshot-id`

```
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}")
)
```

메타데이터 액세스

Iceberg는 SQL을 통해 메타데이터에 대한 액세스를 제공합니다. 네임스페이스를 쿼리하여 지정된 테이블 (`<table_name>`) 의 메타데이터에 액세스할 수 있습니다. `<table_name>.<metadata_table>` 메타데이터 테이블의 전체 목록은 Iceberg 문서의 테이블 검사를 참조하십시오.

다음 예제는 Iceberg 테이블의 커밋 (변경) 기록을 보여주는 Iceberg 기록 메타데이터 테이블에 액세스하는 방법을 보여줍니다.

Amazon EMR 스튜디오 노트북에서 Spark SQL (`%%sql`마법과 함께) 사용하기:

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

API 사용 DataFrames :

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

샘플 출력:

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

Amazon Athena SQL을 사용하여 Apache Iceberg 테이블 작업하기

Amazon Athena는 Apache Iceberg를 기본적으로 지원하며 추가 단계나 구성이 필요하지 않습니다. 이 섹션에서는 지원되는 기능에 대한 자세한 개요와 Athena를 사용하여 Iceberg 테이블과 상호 작용하는 데 필요한 고급 지침을 제공합니다.

버전 및 기능 호환성

> **Note**
>
> 다음 섹션에서는 Athena 엔진 버전 3을 사용하고 있다고 가정합니다.

아이스버그 테이블 사양 지원

Apache Iceberg 테이블 사양은 Iceberg 테이블의 작동 방식을 지정합니다. Athena는 테이블 형식 버전 2를 지원하므로 콘솔, CLI 또는 SDK로 생성하는 모든 Iceberg 테이블은 기본적으로 해당 버전을 사용합니다.

Amazon EMR의 Apache Spark와 같은 다른 엔진으로 만든 Iceberg 테이블을 사용하는 경우 테이블 속성을 사용하여 테이블 형식 버전을 설정해야 합니다. AWS Glue 참조로, 이 가이드 앞부분의 Iceberg 테이블 생성 및 작성 섹션을 참조하십시오.

아이스버그 기능 지원

Athena를 사용하여 Iceberg 테이블에서 읽고 쓸 수 있습니다. `UPDATE`, `MERGE INTO`, 및 `DELETE FROM` 문을 사용하여 데이터를 변경하는 경우 Athena는 merge-on-read 모드만 지원합니다. 이 속성은 변경할 수 없습니다. 에서 데이터를 업데이트하거나 삭제하려면 Amazon EMR의 Apache Spark 또는 같은 다른 엔진을 사용해야 합니다. copy-on-write AWS Glue다음 표에는 Athena의 Iceberg 기능 지원이 요약되어 있습니다.

		DDL 지원		DML 지원		AWS Lake Formation 보안용 (선택 사항)
	테이블 형식	테이블 생성	스키마 진화	데이터 읽기	데이터 쓰기	행/열 액세스 제어
Amazon Athena	버전 2	✓	✓	✓	X C opy-on-write	✓
					✓ M erge-on-read	✓

> **Note**
>
> Athena는 증분 쿼리를 지원하지 않습니다.

아이스버그 테이블 사용하기

Athena에서 Iceberg를 빠르게 사용하는 방법을 알아보려면 이 가이드 앞부분에 있는 Athena SQL의 Iceberg 테이블 시작하기 섹션을 참조하십시오.

다음 표에는 제한 사항 및 권장 사항이 나와 있습니다.

시나리오	제한	권장 사항
테이블 DDL 생성	다른 엔진으로 만든 빙산 테이블은 Athena에서 노출되지 않는 속성을 가질 수 있습니다. 이러한 테이블의 경우 DDL을 생성할 수 없습니다.	테이블을 만든 엔진의 해당 명령문 (예: Spark의 `SHOW CREATE TABLE` 명령문) 을 사용하십시오.
아이스버그 테이블에 기록된 객체의 임의 Amazon S3 접두사	Athena로 만든 Iceberg 테이블에는 기본적으로 이 속성이 활	이 동작을 비활성화하고 Iceberg 테이블 속성을 완전히 제어하려면 Amazon EMR의

시나리오	제한	권장 사항
	성화되어 있습니다`write.object-storage.enabled` .	Spark 또는 같은 다른 엔진으로 Iceberg 테이블을 생성하십시오. AWS Glue
증분 쿼리	Athena에서는 현재 지원되지 않습니다.	증분 쿼리를 사용하여 증분 데이터 수집 파이프라인을 활성화하려면 Amazon EMR의 Spark 또는 을 사용하십시오. AWS Glue

기존 테이블을 Iceberg로 마이그레이션하기

현재 Athena 또는 AWS Glue 테이블 (Hive 테이블이라고도 함) 을 Iceberg 형식으로 마이그레이션하려면 인플레이스 또는 전체 데이터 마이그레이션을 사용할 수 있습니다.

- 인플레이스 마이그레이션은 기존 데이터 파일 위에 Iceberg의 메타데이터 파일을 생성하는 프로세스입니다.
- 전체 데이터 마이그레이션은 Iceberg 메타데이터 레이어를 생성하고 원본 테이블의 기존 데이터 파일을 새 Iceberg 테이블에 다시 씁니다.

다음 섹션에서는 테이블을 마이그레이션하는 데 사용할 수 있는 API에 대한 개요와 마이그레이션 전략 선택을 위한 지침을 제공합니다. 이 두 가지 전략에 대한 자세한 내용은 Iceberg 설명서의 테이블 마이그레이션 섹션을 참조하십시오.

인플레이스 마이그레이션

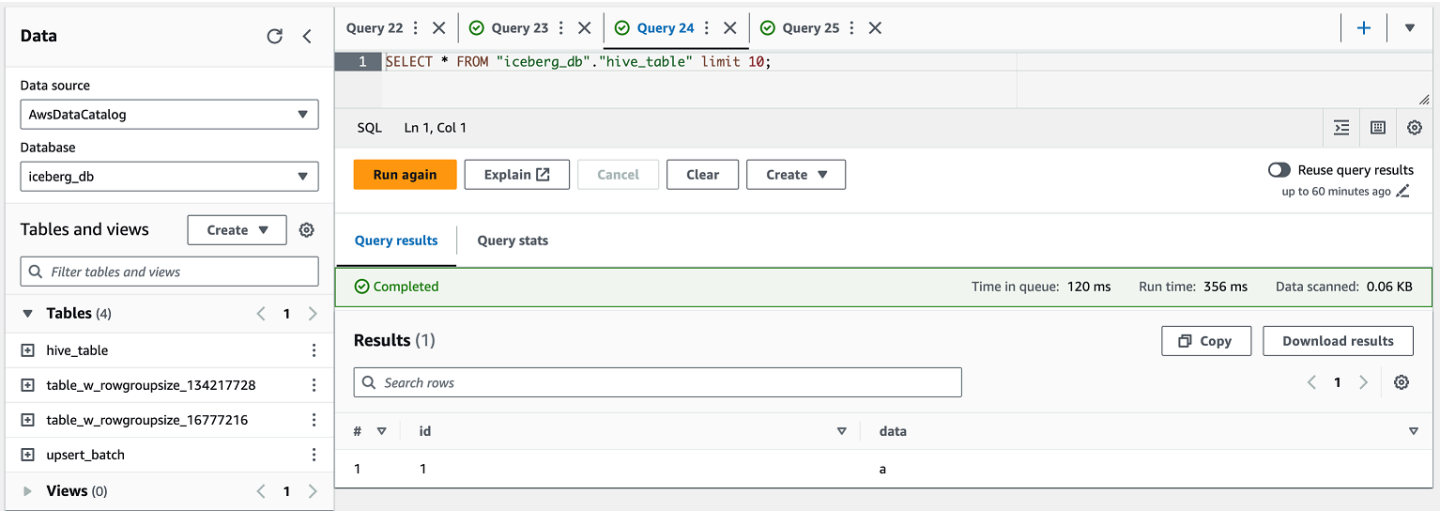
인플레이스 마이그레이션을 사용하면 모든 데이터 파일을 다시 작성할 필요가 없습니다. 대신 Iceberg 메타데이터 파일이 생성되어 기존 데이터 파일에 연결됩니다. Iceberg는 인플레이스 마이그레이션을 구현하기 위한 세 가지 옵션을 제공합니다.

- 이 `snapshot` 절차를 사용하는 방법은 스냅샷 테이블 및 스파크 절차: Iceberg 설명서의 스냅샷 섹션에 설명되어 있습니다.
- `add_files`Iceberg 설명서의 파일 추가 및 Spark 프로시저: add_files 섹션에 설명된 대로 절차를 사용하십시오.

- `migrateIceberg` 설명서의 테이블 및 스파크 마이그레이션 절차: 마이그레이션 섹션에 설명된 대로 절차를 사용하십시오.

현재 마이그레이션 절차는 Hive 메타스토어에서만 AWS Glue Data Catalog작동하므로 직접 사용할 수 없습니다. `snapshot`또는 `add_files` 대신 `migrate` 절차를 사용해야 하는 경우 Hive 메타스토어 (HMS) 와 함께 임시 Amazon EMR 클러스터를 사용할 수 있습니다. 이 접근 방식을 사용하려면 Iceberg 버전 1.2 이상이 필요합니다.

다음과 같은 Hive 테이블을 만들고 싶다고 가정해 보겠습니다.



Athena 콘솔에서 다음 코드를 실행하여 이 Hive 테이블을 생성할 수 있습니다.

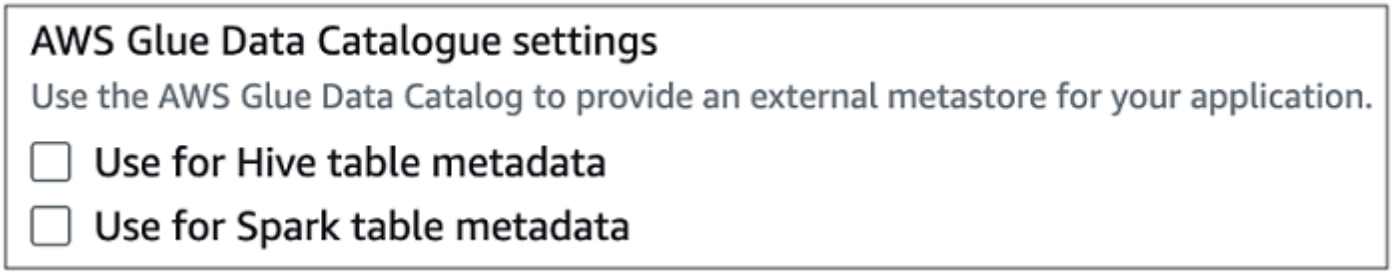
```
CREATE EXTERNAL TABLE 'hive_table'(
  'id' bigint,
  'data' string)
USING parquet
LOCATION
  's3://datalake-xxxx/aws_workshop/iceberg_db/hive_table'

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

Hive 테이블이 분할된 경우 파티션 명령문을 포함하고 Hive 요구 사항에 따라 파티션을 추가하십시오.

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

단계:

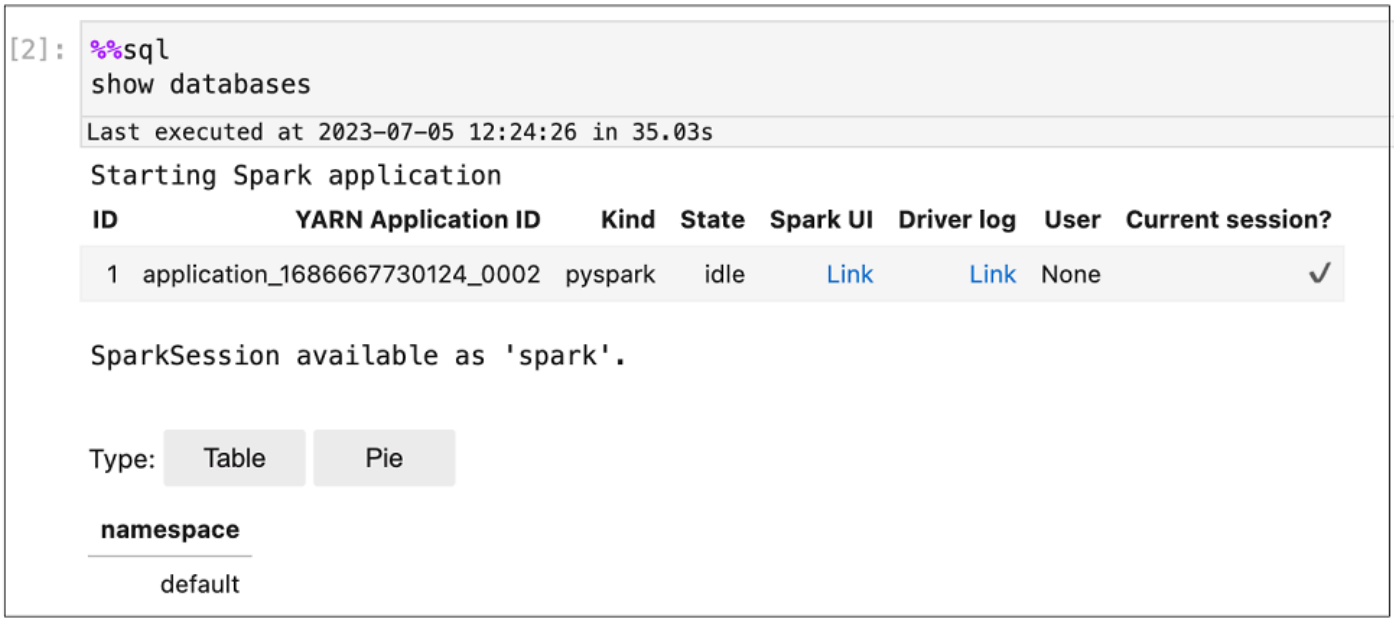
1. AWS Glue Data Catalog 통합을 활성화하지 않고 Amazon EMR 클러스터를 생성합니다. 즉, Hive 또는 Spark 테이블 메타데이터의 확인란을 선택하지 마십시오. 이 해결 방법을 위해 클러스터에서 사용할 수 있는 네이티브 Hive 메타스토어 (HMS) 를 사용하기 때문입니다.



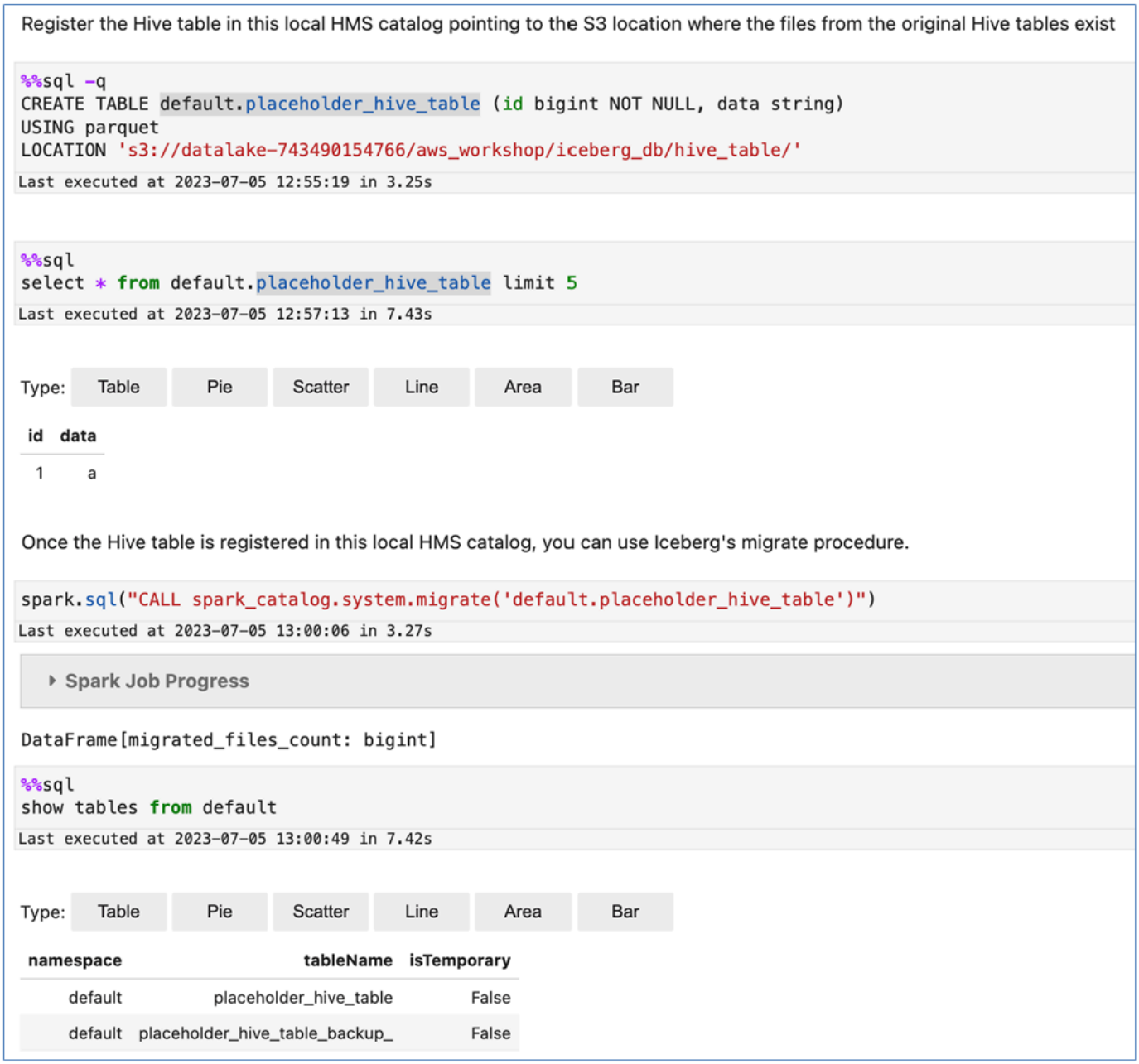
2. Iceberg Hive 카탈로그 구현을 사용하도록 Spark 세션을 구성하십시오.

```
"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. 또는 를 AWS Glue Data Catalog `show databases` 실행하여 Amazon EMR 클러스터가 에 연결되어 있지 않은지 확인합니다. `show tables`

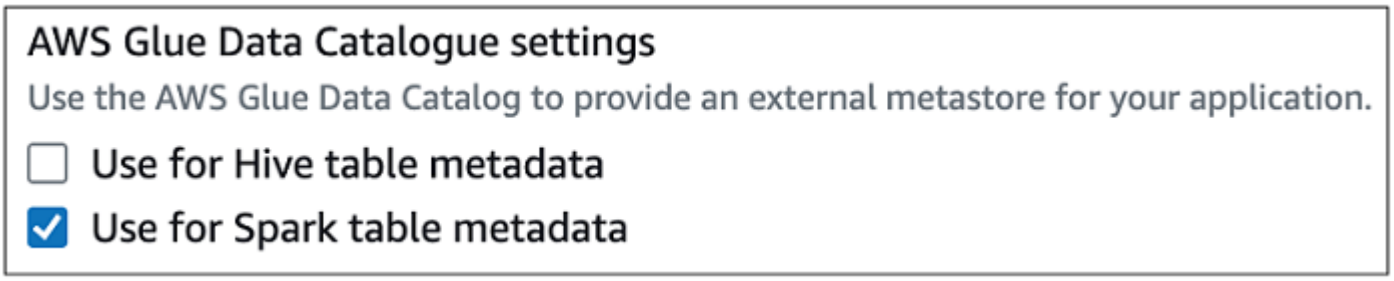


4. Amazon EMR 클러스터의 하이브 메타스토어에 Hive 테이블을 등록한 다음 Iceberg 절차를 사용하십시오. `migrate`



이 절차는 Hive 테이블과 동일한 위치에 Iceberg 메타데이터 파일을 생성합니다.

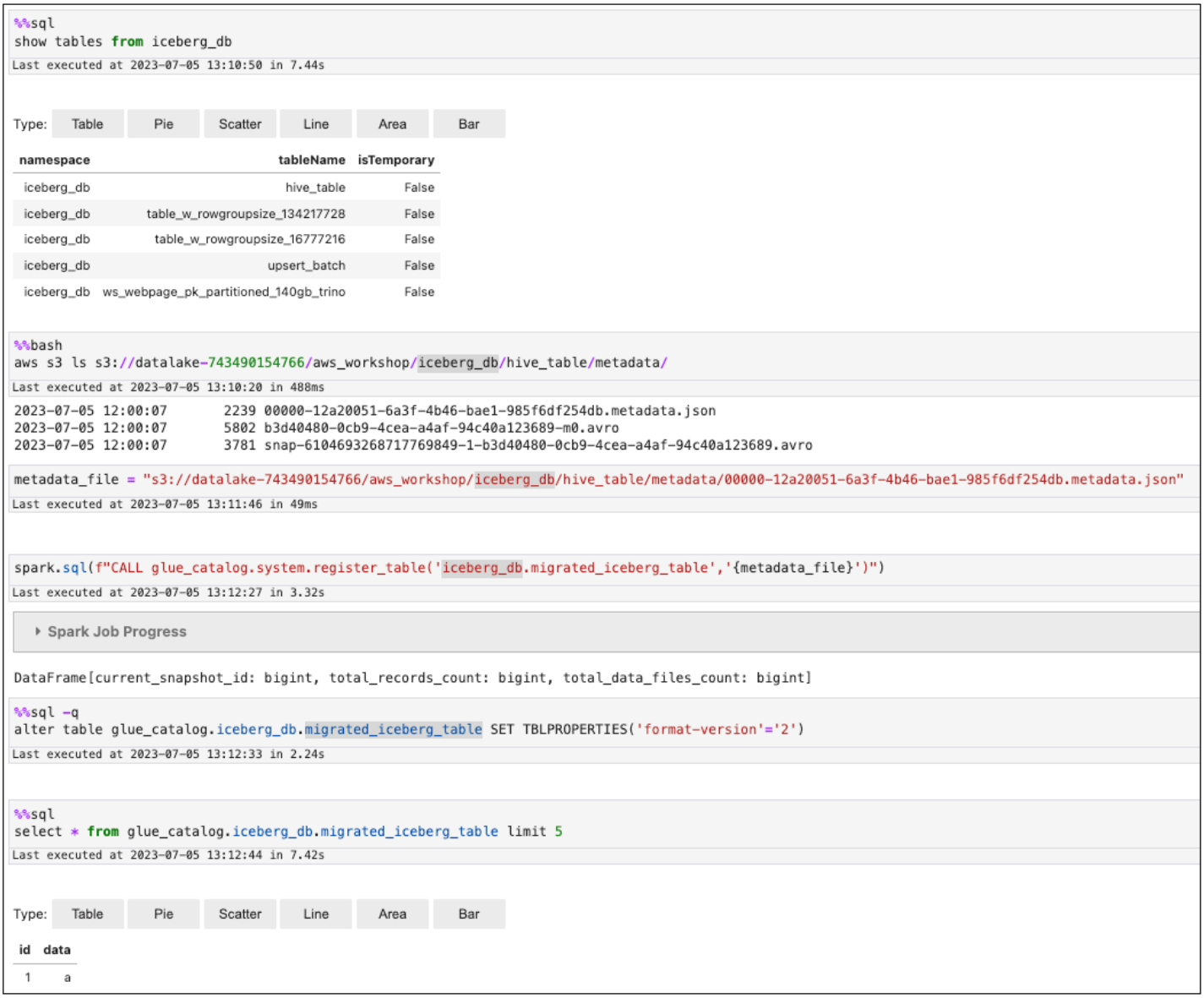
5. 마이그레이션된 Iceberg 테이블을 에 등록하십시오. AWS Glue Data Catalog
6. AWS Glue Data Catalog 통합이 활성화된 Amazon EMR 클러스터로 다시 전환하십시오.



7. Spark 세션에서 다음 Iceberg 구성을 사용하세요.

```
"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",
```

이제 Amazon EMR 또는 Athena에서 이 테이블을 쿼리할 수 있습니다. AWS Glue



전체 데이터 마이그레이션

전체 데이터 마이그레이션은 데이터 파일과 메타데이터를 다시 생성합니다. 이 접근 방식은 인플레이스 마이그레이션에 비해 시간이 더 오래 걸리고 추가 컴퓨팅 리소스가 필요합니다. 그러나 이 옵션은 테이블 품질을 개선하는 데 도움이 됩니다. 즉, 데이터를 검증하고, 스키마와 파티션을 변경하고, 데이터를 재배치하는 등의 작업을 수행할 수 있습니다. 전체 데이터 마이그레이션을 구현하려면 다음 옵션 중 하나를 사용하십시오.

- 아마존 EMR의 스파크나 Athena에서 CREATE TABLE ... AS SELECT (CTAS) 명령문을 사용하십시오. AWS Glue and 절을 사용하여 새 Iceberg 테이블의 파티션 사양과 테이블 속성을 설정할 수

있습니다. `PARTITIONED BY TBLPROPERTIES` 단순히 원본 테이블에서 새 테이블을 상속하는 대신 필요에 따라 새 테이블의 스키마와 파티셔닝을 세밀하게 조정할 수 있습니다.

- Amazon EMR에서 Spark를 사용하거나 AWS Glue (Iceberg 설명서의 테이블 생성 참조) 소스 테이블에서 데이터를 읽고 새 Iceberg 테이블로 작성합니다.

마이그레이션 전략 선택

최상의 마이그레이션 전략을 선택하려면 다음 표의 질문을 고려해 보십시오.

질문	권장 사항
데이터 파일 형식은 무엇입니까 (예: CSV 또는 Apache Parquet)?	• 테이블 파일 형식이 Parquet, ORC 또는 Avro 인 경우 내부 마이그레이션을 고려해 보십시오. • CSV, JSON 등과 같은 다른 형식의 경우 전체 데이터 마이그레이션을 사용하십시오.
테이블 스키마를 업데이트하거나 통합하시겠습니까?	• Iceberg의 기본 기능을 사용하여 테이블 스키마를 발전시키고 싶다면 인플레이스 마이그레이션을 고려해 보세요. 예를 들어, 마이그레이션 후에 열 이름을 바꿀 수 있습니다. (스키마는 Iceberg 메타데이터 레이어에서 변경할 수 있습니다.) • 데이터 파일에서 전체 열을 삭제하려면 전체 데이터 마이그레이션을 사용하는 것이 좋습니다.
파티션 전략을 변경하면 테이블에 도움이 될까요?	• Iceberg의 파티셔닝 방식이 요구 사항을 충족하는 경우 (예: 기존 파티션을 그대로 유지하면서 새 파티션 레이아웃을 사용하여 새 데이터를 저장하는 경우) 내부 마이그레이션을 고려해 보세요.

질문	권장 사항
	• 테이블에 숨겨진 파티션을 사용하려면 전체 데이터 마이그레이션을 고려해 보세요. 숨겨진 파티션에 대한 자세한 내용은 모범 사례 섹션을 참조하십시오.
정렬 순서 전략을 추가하거나 변경하면 테이블에 도움이 될까요?	• 데이터의 정렬 순서를 추가하거나 변경하려면 데이터세트를 다시 작성해야 합니다. 이 경우 전체 데이터 마이그레이션을 사용해 보세요.
	• 모든 테이블 파티션을 다시 작성하는 데 엄청난 비용이 드는 대형 테이블의 경우, 내부 마이그레이션을 사용하고 가장 자주 액세스하는 파티션에 대해 압축 (정렬이 활성화된 상태)을 실행하는 것이 좋습니다.
테이블에 작은 파일이 많나요?	• 작은 파일을 큰 파일로 병합하려면 데이터세트를 다시 작성해야 합니다. 이 경우 전체 데이터 마이그레이션을 사용해 보세요.
	• 모든 테이블 파티션을 다시 작성하는 데 엄청난 비용이 드는 대형 테이블의 경우, 내부 마이그레이션을 사용하고 가장 자주 액세스하는 파티션에 대해 압축 (정렬이 활성화된 상태)을 실행하는 것이 좋습니다.

Apache Iceberg 워크로드 최적화 모범 사례

Iceberg는 데이터 레이크 관리를 단순화하고 워크로드 성능을 향상시키도록 설계된 테이블 형식입니다. 사용 사례마다 비용, 읽기 성능, 쓰기 성능 또는 데이터 보존과 같은 다양한 측면에서 우선 순위가 정해질 수 있으므로 Iceberg는 이러한 장단점을 관리할 수 있는 구성 옵션을 제공합니다. 이 섹션에서는 요구 사항에 맞게 Iceberg 워크로드를 최적화하고 미세 조정하는 방법에 대한 통찰력을 제공합니다.

주제

일반 모범 사례

사용 사례와 상관없이 Apache Iceberg를 사용하는 경우 다음과 AWS같은 일반적인 모범 사례를 따르는 것이 좋습니다.

- Iceberg 형식 버전 2를 사용하세요.

  Athena는 기본적으로 아이스버그 형식 버전 2를 사용합니다.

  Amazon AWS Glue EMR에서 Spark를 사용하거나 Iceberg 테이블을 생성할 때는 Iceberg 설명서에 설명된 대로 형식 버전을 지정하십시오.
- 를 데이터 AWS Glue Data Catalog 카탈로그로 사용하십시오.

  Athena는 AWS Glue Data Catalog 기본적으로 를 사용합니다.

  Amazon EMR에서 Spark를 AWS Glue 사용하거나 Iceberg를 사용하는 경우, Spark 세션에 다음 구성을 추가하여 AWS Glue 데이터 카탈로그를 사용하십시오. 자세한 내용은 이 가이드 앞부분의 AWS Glue의 Iceberg용 Spark 구성 섹션을 참조하십시오.

```
"spark.sql.catalog.<your_catalog_name>.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog"
```

- AWS Glue Data Catalog as Lock Manager를 사용하십시오.

  Athena는 Iceberg AWS Glue Data Catalog 테이블의 경우 기본적으로 를 잠금 관리자로 사용합니다.

  Amazon AWS Glue EMR에서 Spark를 사용하거나 Iceberg와 함께 작업하는 경우 Spark 세션 구성을 잠금 관리자로 사용하도록 AWS Glue Data Catalog 구성해야 합니다. 자세한 내용은 Iceberg 설명서의 낙관적 잠금을 참조하십시오.
- Z표준 (ZSTD) 압축을 사용하세요.

  Iceberg의 기본 압축 코덱은 gzip이며, 테이블 속성을 사용하여 수정할 수 있습니다. `write.<file_type>.compression-codec` Athena는 이미 ZSTD를 Iceberg 테이블의 기본 압축 코덱으로 사용하고 있습니다.

  일반적으로 ZSTD 압축 코덱은 GZIP과 Snappy 사이의 균형을 유지하고 압축률을 손상시키지 않으면서 우수한 읽기/쓰기 성능을 제공하므로 사용하는 것이 좋습니다. 또한 필요에 맞게 압축 수준을 조정할 수 있습니다. 자세한 내용은 Athena 설명서의 Athena의 ZSTD 압축 수준을 참조하십시오.

  Snappy는 전반적으로 최상의 읽기 및 쓰기 성능을 제공하지만 GZIP 및 ZSTD보다 압축률이 낮을 수 있습니다. 성능의 우선 순위를 정하는 경우 (Amazon S3에 더 큰 데이터 볼륨을 저장하는 것을 의미하더라도) Snappy가 최적의 선택일 수 있습니다.

읽기 성능 최적화

이 섹션에서는 엔진과 관계없이 읽기 성능을 최적화하기 위해 조정할 수 있는 테이블 속성에 대해 설명합니다.

분할

Hive 테이블과 마찬가지로 Iceberg는 불필요한 메타데이터 파일 및 데이터 파일을 읽지 않도록 파티션을 인덱싱의 기본 계층으로 사용합니다. 열 통계는 쿼리 계획을 더욱 개선하기 위한 인덱싱의 보조 계층으로도 고려되어 전체 실행 시간을 단축합니다.

데이터 파티셔닝

Iceberg 테이블을 쿼리할 때 스캔되는 데이터의 양을 줄이려면 예상 읽기 패턴에 맞는 균형 잡힌 파티션 전략을 선택하세요.

- 쿼리에 자주 사용되는 열을 식별하세요. 이러한 파티셔닝 후보들은 이상적인 파티셔닝 후보입니다. 예를 들어, 일반적으로 특정 날짜의 데이터를 쿼리하는 경우 파티션 열의 자연스러운 예로 날짜 열을 들 수 있습니다.
- 파티션이 너무 많이 생성되지 않도록 하려면 카디널리티가 낮은 파티션 열을 선택하십시오. 파티션이 너무 많으면 테이블의 파일 수가 증가하여 쿼리 성능에 부정적인 영향을 미칠 수 있습니다. 일반적으로 "너무 많은 파티션"은 대부분의 파티션에 있는 데이터 크기가 에서 설정한 값의 2-5배 미만인 시나리오로 정의할 수 있습니다. `target-file-size-bytes`

> **Note**
>
> 일반적으로 카디널리티가 높은 열 (예: 수천 개의 값을 포함할 수 있는 `id` 열) 에서 필터를 사용하여 쿼리하는 경우 다음 섹션에서 설명하는 대로 Iceberg의 숨겨진 파티션 기능을 버킷 변환과 함께 사용하십시오.

숨겨진 파티셔닝 사용하기

쿼리가 일반적으로 테이블 열의 파생 객체를 기준으로 필터링되는 경우 파티션으로 사용할 새 열을 명시적으로 생성하는 대신 숨겨진 파티션을 사용하세요. 이 기능에 대한 자세한 내용은 Iceberg 설명서를 참조하십시오.

예를 들어 타임스탬프 열이 있는 데이터세트 (예:`2023-01-01 09:00:00`) 에서 파싱된 날짜로 새 열을 만드는 대신 (예:`2023-01-01`) 파티션 변환을 사용하여 타임스탬프에서 날짜 부분을 추출하고 이러한 파티션을 즉시 생성할 수 있습니다.

숨겨진 파티셔닝의 가장 일반적인 사용 사례는 다음과 같습니다.

- 데이터에 타임스탬프 열이 있는 날짜 또는 시간을 기준으로 파티셔닝합니다. Iceberg는 타임스탬프의 날짜 또는 시간 부분을 추출하기 위한 여러 변환을 제공합니다.
- 열의 해시 함수로 파티셔닝하는 경우, 파티셔닝 열의 카디널리티가 높아 파티션 수가 너무 많아질 수 있습니다. Iceberg의 버킷 변환은 파티션 열에 해시 함수를 사용하여 여러 파티션 값을 더 적은 수의 숨겨진 (버킷) 파티션으로 그룹화합니다.

사용 가능한 모든 파티션 변환에 대한 개요는 Iceberg 설명서의 파티션 변환을 참조하십시오.

숨겨진 파티셔닝에 사용되는 열은 및 같은 일반 SQL 함수를 사용하여 쿼리 조건자의 일부가 될 수 있습니다. `year()` `month()` 조건자를 및 와 같은 연산자와 조합할 수도 있습니다. `BETWEEN` `AND`

> **Note**
>
> Iceberg는 다른 데이터 유형을 생성하는 함수 (예:) 에 대해서는 파티션 프루닝을 수행할 수 없습니다. `substring(event_time, 1, 10) = '2022-01-01'`

파티션 진화 사용

기존 파티션 전략이 최적이 아닐 때는 Iceberg의 파티션 진화를 사용하세요. 예를 들어, 시간당 파티션이 너무 작아서 (각각 몇 메가바이트에 불과함) 선택한 경우 일별 또는 월별 파티션으로 전환하는 것을 고려해 보세요.

처음에는 테이블에 가장 적합한 파티션 전략이 확실하지 않고 더 많은 통찰력을 얻으면서 파티션 전략을 구체화하고 싶을 때 이 방법을 사용할 수 있습니다. 파티션 진화의 또 다른 효과적인 용도는 데이터 볼륨이 변하고 시간이 지남에 따라 현재의 파티션 전략의 효과가 떨어지는 경우입니다.

파티션을 발전시키는 방법에 대한 지침은 Iceberg 설명서의 ALTER TABLE SQL 확장을 참조하십시오.

파일 크기 조정

쿼리 성능을 최적화하려면 테이블에 있는 작은 파일 수를 최소화해야 합니다. 쿼리 성능을 높이려면 일반적으로 Parquet 및 ORC 파일을 100MB 이상으로 유지하는 것이 좋습니다.

파일 크기는 Iceberg 테이블의 쿼리 계획에도 영향을 줍니다. 테이블의 파일 수가 늘어날수록 메타데이터 파일의 크기도 커집니다. 메타데이터 파일이 크면 쿼리 계획 속도가 느려질 수 있습니다. 따라서 테이블 크기가 커지면 파일 크기를 늘려 메타데이터가 기하급수적으로 확장되는 것을 방지하십시오.

다음 모범 사례를 사용하여 Iceberg 테이블에서 적절한 크기의 파일을 생성하십시오.

대상 파일 및 행 그룹 크기를 설정합니다.

Iceberg는 데이터 파일 레이아웃 조정을 위한 다음과 같은 주요 구성 매개변수를 제공합니다. 이 파라미터를 사용하여 대상 파일 크기와 행 그룹 또는 스트라이크 크기를 설정하는 것이 좋습니다.

파라미터	기본값	Comment
`write.target-file-size-bytes`	512MB	이 매개변수는 Iceberg가 생성할 최대 파일 크기를 지정합니다. 하지만 특정 파일은 이 제한

파라미터	기본값	Comment
		보다 작은 크기로 기록될 수 있습니다.
`write.parquet.row-group-size-bytes`	128MB	Parquet과 ORC 모두 데이터를 청크 단위로 저장하므로 일부 작업에서는 엔진이 전체 파일을 읽지 않아도 됩니다.
`write.orc.stripe-size-bytes`	64메가바이트	
`write.distribution-mode`	없음, 아이스버그 버전 1.1 이하인 경우 해시, 아이스버그 버전 1.2부터 시작	Iceberg는 Spark에 스토리지에 쓰기 전에 작업 간에 데이터를 정렬하도록 요청합니다.

- 예상 테이블 크기에 따라 다음 일반 가이드라인을 따르세요.
  - 소형 테이블 (최대 몇 기가바이트) - 대상 파일 크기를 128MB로 줄이십시오. 또한 행 그룹 또는 스트라이프 크기를 줄이십시오 (예: 8MB 또는 16MB).
  - 중대형 테이블 (몇 기가바이트에서 수백 기가바이트까지) — 이러한 테이블은 기본값을 사용하는 것이 좋습니다. 매우 선별적인 쿼리인 경우 행 그룹 또는 스트라이프 크기를 조정하십시오 (예: 16MB).
  - 매우 큰 테이블 (수백 기가바이트 또는 테라바이트) - 대상 파일 크기를 1024MB 이상으로 늘리고 쿼리에서 일반적으로 많은 양의 데이터를 가져오는 경우 행 그룹이나 스트라이프 크기를 늘리는 것이 좋습니다.

- Iceberg 테이블에 쓰는 Spark 애플리케이션이 적절한 크기의 파일을 생성하도록 하려면 속성을 또는 로 설정하십시오. `write.distribution-mode` `hash` `range` 이러한 모드 간의 차이점에 대한 자세한 설명은 Iceberg 설명서의 분산 모드 작성을 참조하십시오.

다음은 일반적인 지침입니다. 테스트를 실행하여 특정 테이블 및 워크로드에 가장 적합한 값을 식별하는 것이 좋습니다.

정기적으로 압축을 실행하세요.

위 표의 구성은 쓰기 작업에서 생성할 수 있는 최대 파일 크기를 설정하지만 파일이 해당 크기를 가질 것이라고 보장하지는 않습니다. 파일 크기가 적절한지 확인하려면 정기적으로 압축을 실행하여 작은 파일을 큰 파일로 합치십시오. 압축 실행에 대한 자세한 지침은 이 가이드 뒷부분에 있는 Iceberg 압축을 참조하십시오.

컬럼 통계 최적화

Iceberg는 열 통계를 사용하여 파일 프루닝을 수행하는데, 이는 쿼리로 스캔되는 데이터의 양을 줄임으로써 쿼리 성능을 향상시킵니다. 열 통계를 활용하려면 Iceberg가 쿼리 필터에서 자주 사용되는 모든 열에 대한 통계를 수집하도록 하세요.

기본적으로 Iceberg는 테이블 속성에 정의된 대로 각 테이블의 처음 100개 열에 대한 통계만 수집합니다. `write.metadata.metrics.max-inferred-column-defaults` 테이블에 100개가 넘는 열이 있고 쿼리가 처음 100개 열을 제외한 열을 자주 참조하는 경우 (예: 132열을 필터링하는 쿼리가 있을 수 있음) Iceberg가 해당 열에 대한 통계를 수집하는지 확인하세요. 이를 위한 두 가지 옵션이 있습니다.

- Iceberg 테이블을 생성할 때는 쿼리에 필요한 열이 에서 설정한 열 범위 `write.metadata.metrics.max-inferred-column-defaults` (기본값 100) 에 속하도록 열을 재정렬하십시오.

  참고: 100개 열에 대한 통계가 필요하지 않은 경우 `write.metadata.metrics.max-inferred-column-defaults` 구성을 원하는 값 (예: 20) 으로 조정하고 쿼리를 읽고 쓰는 데 필요한 열이 데이터세트 왼쪽의 처음 20개 열에 포함되도록 열을 재정렬할 수 있습니다.
- 쿼리 필터에 몇 개의 열만 사용하는 경우 다음 예와 같이 지표 수집을 위한 전체 속성을 비활성화하고 통계를 수집할 개별 열을 선택적으로 선택할 수 있습니다.

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

참고: 열 통계는 해당 열을 기준으로 데이터를 정렬할 때 가장 효과적입니다. 자세한 내용은 이 가이드 뒷부분에 있는 정렬 순서 설정 섹션을 참조하십시오.

적절한 업데이트 전략을 선택하세요.

사용 사례에 따라 느린 쓰기 작업이 허용되는 경우 copy-on-write 전략을 사용하여 읽기 성능을 최적화하십시오. 이는 Iceberg에서 사용하는 기본 전략입니다.

C는 opy-on-write 파일을 읽기 최적화된 방식으로 스토리지에 직접 쓰기 때문에 읽기 성능이 향상됩니다. 그러나 에 비해 각 쓰기 작업은 시간이 더 오래 걸리고 더 많은 컴퓨팅 리소스를 소비합니다. merge-on-read 따라서 일반적으로 읽기 지연 시간과 쓰기 지연 시간 간의 균형을 맞출 수 있습니다. 일반적으로 대부분의 업데이트가 동일한 테이블 파티션에 배치되는 사용 사례 (예: 일일 배치 로드) 에 적합합니다. copy-on-write

C opy-on-write 구성 (`write.update.mode`,`write.delete.mode`, 및`write.merge.mode`) 은 테이블 수준에서 설정하거나 애플리케이션 측에서 독립적으로 설정할 수 있습니다.

ZSTD 압축 사용

table 속성을 사용하여 Iceberg에서 사용하는 압축 코덱을 수정할 수 있습니다. `write.<file_type>.compression-codec` 테이블의 전반적인 성능을 향상시키려면 ZSTD 압축 코덱을 사용하는 것이 좋습니다.

기본적으로 Iceberg 버전 1.3 및 이전 버전에서는 ZSTD에 비해 읽기/쓰기 성능이 느린 GZIP 압축을 사용합니다.

참고: 일부 엔진은 다른 기본값을 사용할 수 있습니다. Athena 또는 Amazon EMR 버전 7.x로 만든 아이스버그 테이블의 경우가 이에 해당합니다.

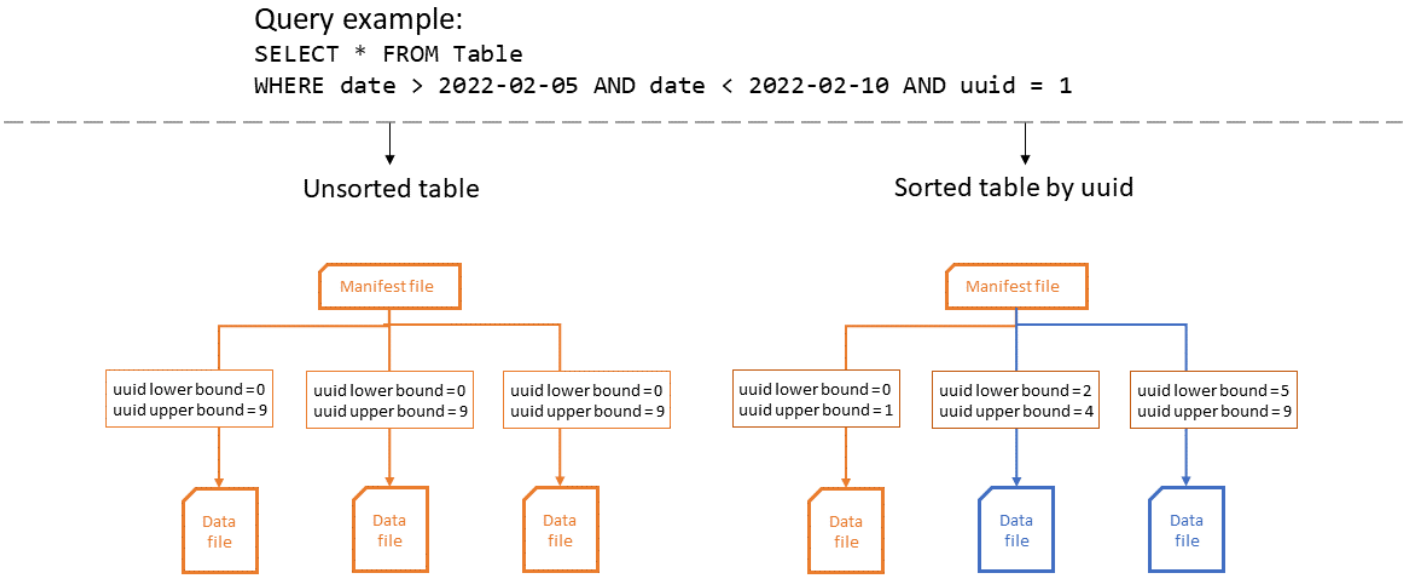
정렬 순서를 설정합니다.

Iceberg 테이블의 읽기 성능을 개선하려면 쿼리 필터에서 자주 사용되는 하나 이상의 열을 기준으로 테이블을 정렬하는 것이 좋습니다. 정렬을 Iceberg의 열 통계와 결합하면 파일 프루닝의 효율성이 크게 향상되어 읽기 작업 속도가 빨라질 수 있습니다. 또한 정렬은 쿼리 필터의 정렬 열을 사용하는 쿼리에 대한 Amazon S3 요청 수를 줄입니다.

Spark로 데이터 정의 언어 (DDL) 문을 실행하여 테이블 수준에서 계층적 정렬 순서를 설정할 수 있습니다. 사용 가능한 옵션은 Iceberg 설명서를 참조하십시오. 정렬 순서를 설정한 후 작성자는 이 정렬을 Iceberg 테이블의 후속 데이터 쓰기 작업에 적용합니다.

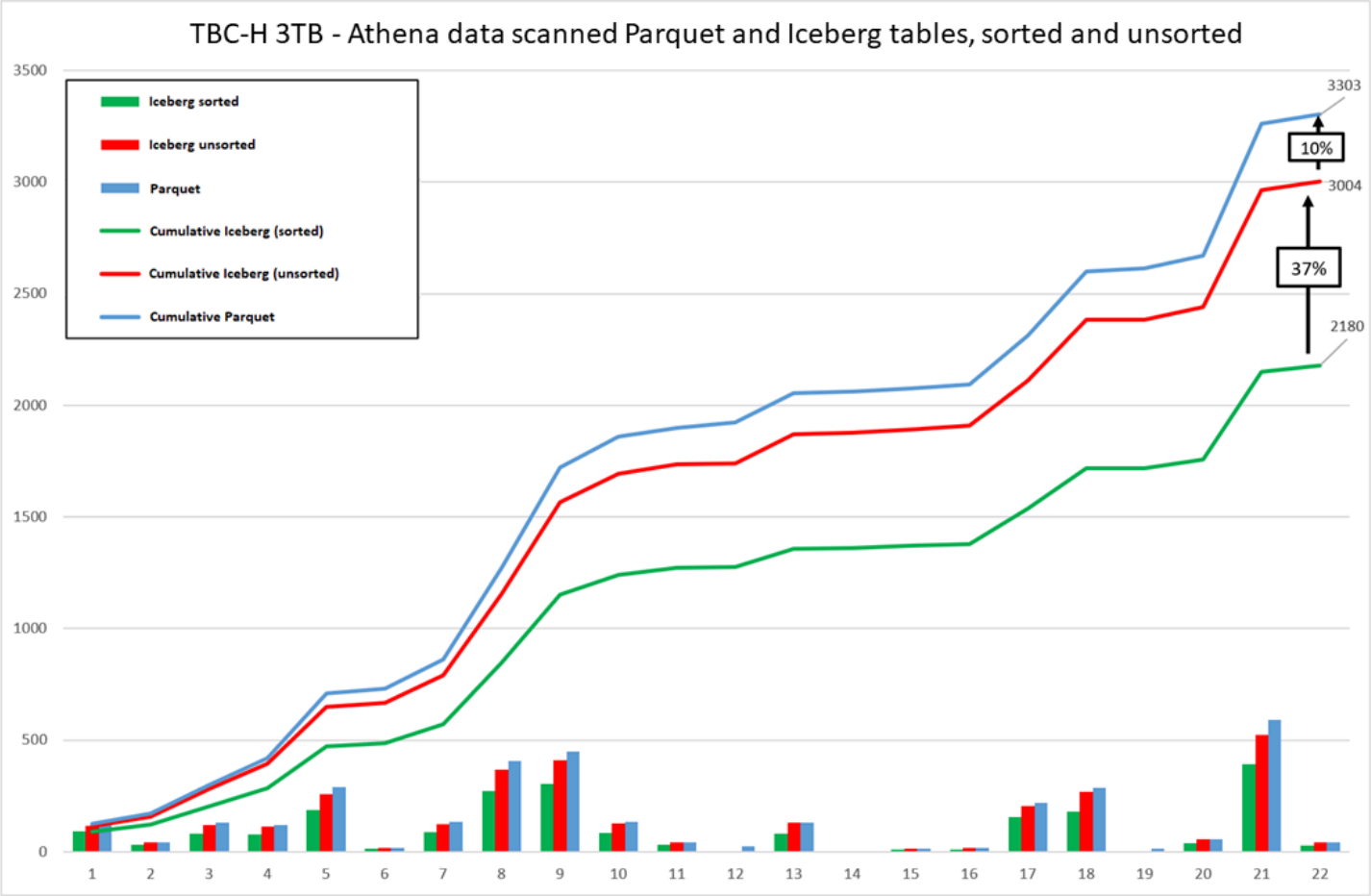
예를 들어, 대부분의 쿼리가 필터링 기준인 날짜 (`yyyy-mm-dd`) 로 분할된 테이블에서 DDL 옵션을 사용하여 Spark가 범위가 겹치지 않는 파일을 `Write Distributed By Partition Locally Ordered` 쓰도록 할 수 있습니다. `uuid`

다음 다이어그램은 테이블을 정렬할 때 열 통계의 효율성이 어떻게 향상되는지를 보여줍니다. 이 예제에서 정렬된 테이블은 단일 파일만 열면 되므로 Iceberg의 파티션과 파일을 최대한 활용할 수 있습니다. 정렬되지 않은 테이블에서는 임의의 데이터 파일에 임의의 데이터가 `uuid` 존재할 수 있으므로 쿼리는 모든 데이터 파일을 열어야 합니다.

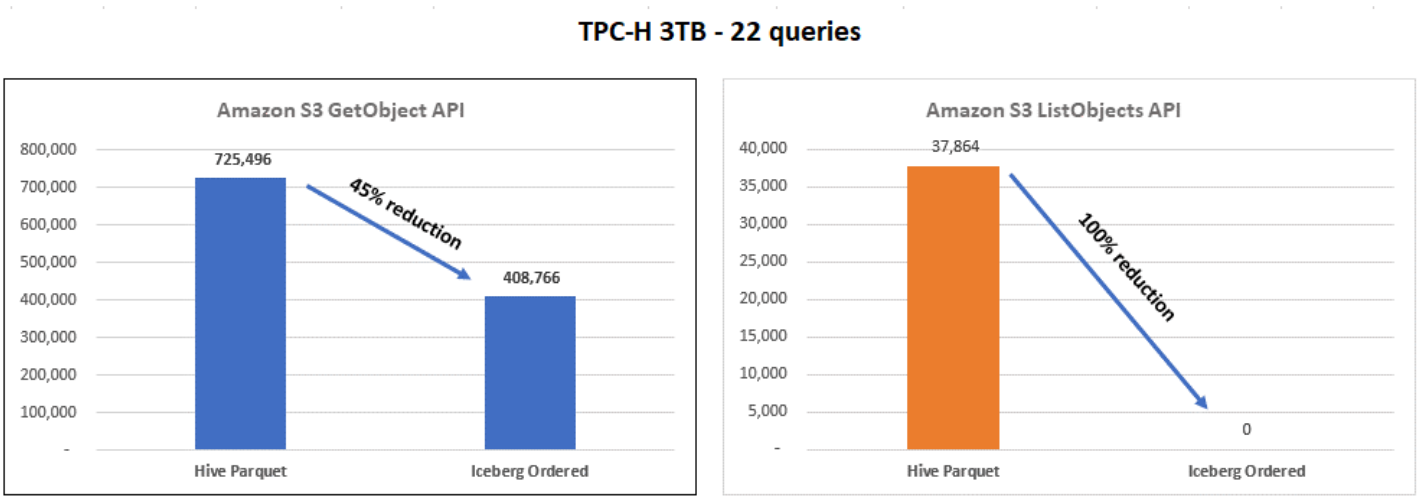


정렬 순서를 변경해도 기존 데이터 파일에는 영향을 주지 않습니다. Iceberg 압축을 사용하여 해당 항목에 정렬 순서를 적용할 수 있습니다.

다음 그래프에서 볼 수 있듯이 Iceberg 정렬 테이블을 사용하면 워크로드 비용을 줄일 수 있습니다.



이 그래프는 Hive (Parquet) 테이블에 대한 TPC-H 벤치마크를 실행한 결과를 Iceberg의 정렬 테이블과 비교하여 요약합니다. 하지만 다른 데이터셋이나 워크로드에서는 결과가 다를 수 있습니다.



쓰기 성능 최적화

이 섹션에서는 엔진과 관계없이 Iceberg 테이블의 쓰기 성능을 최적화하기 위해 조정할 수 있는 테이블 속성에 대해 설명합니다.

테이블 배포 모드를 설정하세요.

Iceberg는 쓰기 데이터가 Spark 태스크에 분산되는 방식을 정의하는 다양한 쓰기 분산 모드를 제공합니다. 사용 가능한 모드에 대한 개요는 Iceberg 설명서의 쓰기 분산 모드를 참조하십시오.

특히 스트리밍 워크로드에서 쓰기 속도를 우선시하는 사용 사례의 경우 로 설정하십시오. `write.distribution-mode` none 이렇게 하면 Iceberg에서 추가 Spark 셔플링을 요청하지 않고 Spark 태스크에서 사용할 수 있게 되는 대로 데이터가 기록됩니다. 이 모드는 Spark 구조적 스트리밍 애플리케이션에 특히 적합합니다.

> **Note**
>
> 쓰기 분산 모드를 로 설정하면 작은 파일이 많이 생성되어 읽기 성능이 저하되는 none 경향이 있습니다. 쿼리 성능을 위해 이러한 작은 파일을 적절한 크기의 파일로 통합하려면 정기적으로 압축하는 것이 좋습니다.

적절한 업데이트 전략을 선택하세요.

사용 사례에 따라 최신 데이터에 대한 느린 읽기 작업이 허용되는 경우 merge-on-read 전략을 사용하여 쓰기 성능을 최적화하십시오.

를 사용하면 merge-on-read Iceberg는 업데이트와 삭제를 별도의 작은 파일로 스토리지에 기록합니다. 테이블을 읽을 때 독자는 이러한 변경 내용을 기본 파일에 병합하여 데이터의 최신 보기를 반환해야 합니다. 이로 인해 읽기 작업의 성능이 저하되지만 업데이트 및 삭제 쓰기 속도는 빨라집니다. 일반적으로 업데이트가 있는 워크로드 또는 업데이트가 거의 없는 작업을 여러 테이블 파티션에 분산하여 스트리밍하는 데 적합합니다. merge-on-read

merge-on-read 구성 (`write.update.mode`write.delete.mode, 및`write.merge.mode`) 을 테이블 수준에서 설정하거나 애플리케이션 측에서 독립적으로 설정할 수 있습니다.

를 merge-on-read 사용하려면 정기적으로 압축을 실행하여 시간이 지나도 읽기 성능이 저하되지 않도록 해야 합니다. 압축은 업데이트 및 삭제를 기존 데이터 파일과 조정하여 새로운 데이터 파일 세트를 생성하므로 읽기 측면에서 발생하는 성능 저하를 방지합니다. 기본적으로 Iceberg의 압축은 `delete-`

file-threshold 속성의 기본값을 더 작은 값으로 변경하지 않는 한 삭제 파일을 병합하지 않습니다 (Iceberg 설명서 참조). 압축에 대한 자세한 내용은 이 가이드의 뒷부분에 있는 Iceberg 압축 섹션을 참조하십시오.

적절한 파일 형식을 선택하세요.

Iceberg는 파켓, ORC, 아브로 형식의 데이터 쓰기를 지원합니다. 파켓이 기본 형식입니다. Parquet 및 ORC는 뛰어난 읽기 성능을 제공하지만 일반적으로 쓰기 속도가 느린 열 형식 형식입니다. 이는 읽기와 쓰기 성능 간의 일반적인 균형을 나타냅니다.

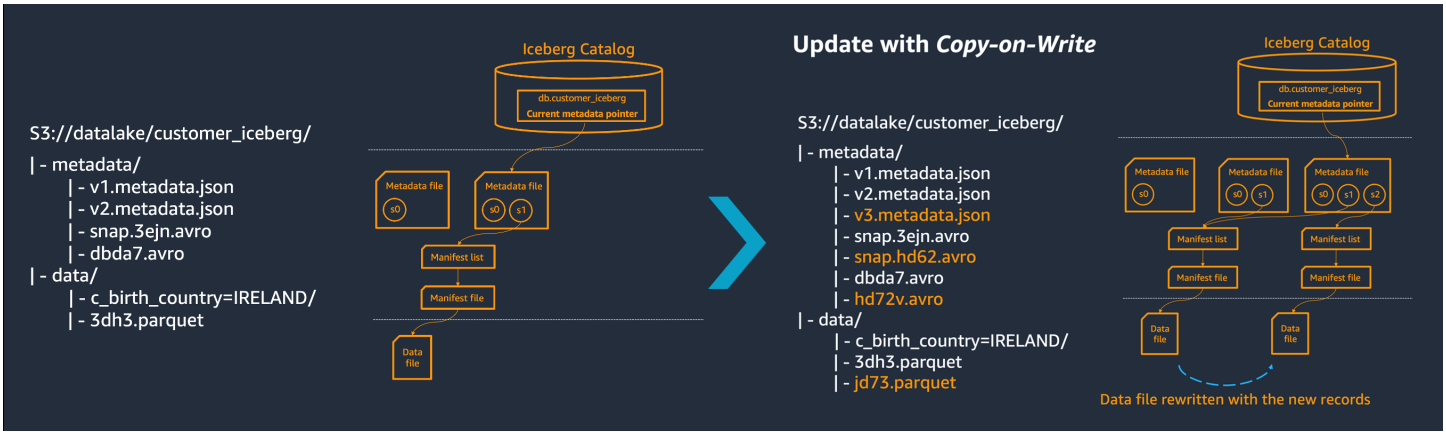
스트리밍 워크로드와 같은 사용 사례에서 쓰기 속도가 중요한 경우 작성기 옵션에서 로 설정하여 Avro 형식으로 작성하는 write-format 것을 Avro 고려해 보십시오. Avro는 행 기반 형식이므로 읽기 성능은 느리지만 쓰기 시간은 더 빠릅니다.

읽기 성능을 개선하려면 일반 압축을 실행하여 작은 Avro 파일을 더 큰 Parquet 파일로 병합하고 변환하십시오. 압축 프로세스의 결과는 테이블 설정에 따라 달라집니다. write.format.default Iceberg의 기본 형식은 Parquet이므로 Avro로 작성한 다음 압축을 실행하면 Iceberg는 Avro 파일을 Parquet 파일로 변환합니다. 다음은 그 예입니다.

```
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()
```

스토리지 최적화

다음 다이어그램과 같이 Iceberg 테이블에서 데이터를 업데이트하거나 삭제하면 데이터 사본 수가 늘어납니다. 압축을 실행할 때도 마찬가지입니다. 압축을 실행하면 Amazon S3의 데이터 사본 수가 늘어납니다. 이는 Iceberg가 모든 테이블의 기반이 되는 파일을 변경할 수 없는 것으로 취급하기 때문입니다.



이 섹션의 모범 사례에 따라 스토리지 비용을 관리하세요.

S3 인텔리전트 계층화를 활성화합니다.

Amazon S3 Intelligent-Tiering 스토리지 클래스를 사용하면 액세스 패턴이 변경될 때 데이터를 가장 비용 효율적인 액세스 티어로 자동으로 이동합니다. 이 옵션은 운영 오버헤드나 성능에 미치는 영향이 없습니다.

참고: Iceberg 테이블을 사용한 S3 지능형 계층화에서는 선택적 계층 (예: 아카이브 액세스 및 딥 아카이브 액세스) 을 사용하지 마십시오. 데이터를 보관하려면 다음 섹션의 지침을 참조하십시오.

또한 Amazon S3 수명 주기 규칙을 사용하여 S3 스탠다드-IA 또는 S3 One Zone-IA와 같은 다른 Amazon S3 스토리지 클래스로 객체를 이동하기 위한 자체 규칙을 설정할 수 있습니다 (Amazon S3 설명서의 지원되는 전환 및 관련 제약 참조).

과거 스냅샷을 보관 또는 삭제합니다.

Iceberg 테이블에 커밋된 모든 트랜잭션 (삽입, 업데이트, 병합, 압축) 에 대해 테이블의 새 버전 또는 스냅샷이 생성됩니다. 시간이 지남에 따라 Amazon S3의 버전 수와 메타데이터 파일 수가 누적됩니다.

스냅샷 격리, 테이블 롤백, 시간 여행 쿼리와 같은 기능을 사용하려면 테이블의 스냅샷을 보관해야 합니다. 하지만 보관하는 버전 수에 따라 스토리지 비용도 증가합니다.

다음 표에는 데이터 보존 요구 사항에 따라 비용을 관리하기 위해 구현할 수 있는 설계 패턴이 설명되어 있습니다.

디자인 패턴	솔루션	사용 사례
이전 스냅샷 삭제	• Athena의 VACUUM 문을 사용하여 이전 스냅샷을 제거합니다. 이 작업에는 컴퓨팅 비용이 발생하지 않습니다. • 또는 Amazon AWS Glue EMR에서 Spark를 사용하거나 스냅샷을 제거할 수 있습니다. 자세한 내용은 Iceberg 설명서의 expire_snapshots를 참조하십시오.	이 접근 방식은 더 이상 필요하지 않은 스냅샷을 삭제하여 스토리지 비용을 절감합니다. 데이터 보존 요구 사항에 따라 보존해야 하는 스냅샷의 수 또는 보존 기간을 구성할 수 있습니다. 이 옵션은 스냅샷을 영구 삭제합니다. 만료된 스냅샷으로 롤백하거나 시간 여행을 할 수 없습니다.
특정 스냅샷에 대한 보존 정책을 설정합니다.	1. 태그를 사용하여 특정 스냅샷을 표시하고 Iceberg에서 보존 정책을 정의할 수 있습니다. 자세한 내용은 Iceberg 설명서의 히스토리 태그를 참조하십시오. 예를 들어 Amazon EMR의 Spark에서 다음 SQL 문을 사용하여 매월 하나의 스냅샷을 1년 동안 보존할 수 있습니다. `ALTER TABLE glue_catalog.db.table CREATE TAG 'EOM-01' AS OF VERSION 30 RETAIN 365 DAYS` 2. Amazon EMR에서 Spark를 AWS Glue 사용하거나 태그	이 패턴은 과거 특정 시점의 테이블 상태를 표시해야 하는 비즈니스 또는 법적 요구 사항을 준수하는 데 유용합니다. 태그가 지정된 특정 스냅샷에 보존 정책을 적용하면 생성된 다른 (태그가 지정되지 않은) 스냅샷을 제거할 수 있습니다. 이렇게 하면 생성된 모든 스냅샷을 보존하지 않고도 데이터 보존 요구 사항을 충족할 수 있습니다.

디자인 패턴	솔루션	사용 사례
	가 지정되지 않은 나머지 중간 스냅샷을 제거합니다.	

디자인 패턴	솔루션	사용 사례
오래된 스냅샷을 보관하세요	1. Amazon S3 태그를 사용하여 Spark로 객체를 표시할 수 있습니다. (Amazon S3 태그는 Iceberg 태그와 다릅니다. 자세한 내용은 Iceberg 설명서를 참조하십시오.) 예: `spark.sql.catalog.my_catalog.s3.delete-enabled=false and \ spark.sql.catalog.my_catalog.s3.delete.tags.my_key=to_archive` 2. Amazon EMR에서 Spark를 AWS Glue 사용하거나 스냅샷을 제거할 수 있습니다. 예제의 설정을 사용하면 이 프로시저는 객체에 태그를 지정하고 Amazon S3에서 객체를 삭제하는 대신 Iceberg 테이블 메타데이터에서 분리합니다. 3. S3 수명 주기 규칙을 사용하여 태그가 지정된 객체를 S3 Glacier 스토리지 `to_archive` 클래스 중 하나로 전환할 수 있습니다. 4. 보관된 데이터를 쿼리하려면: • 보관된 객체를 복원합니다.	이 패턴을 사용하면 모든 테이블 버전과 스냅샷을 저렴한 비용으로 유지할 수 있습니다. 먼저 해당 버전을 새 테이블로 복원하지 않으면 시간 여행을 하거나 보관된 스냅샷으로 롤백할 수 없습니다. 이는 일반적으로 감사 목적으로 사용할 수 있습니다. 이 접근 방식을 이전 설계 패턴과 결합하여 특정 스냅샷에 대한 보존 정책을 설정할 수 있습니다.

디자인 패턴	솔루션	사용 사례
	• Iceberg의 register_table 프로시저를 사용하여 스냅샷을 카탈로그의 테이블로 등록할 수 있습니다. 자세한 지침은 Amazon S3 데이터 레이크에 빌드되는 Apache Iceberg 테이블의 운영 효율성 향상 AWS 블로그 게시물을 참조하십시오.	

고립된 파일 삭제

특정 상황에서는 거래를 커밋하기 전에 Iceberg 애플리케이션이 실패할 수 있습니다. 그러면 Amazon S3에 데이터 파일이 남습니다. 커밋이 없었기 때문에 이러한 파일은 어떤 테이블과도 연결되지 않으므로 비동기적으로 정리해야 할 수도 있습니다.

이러한 삭제를 처리하려면 Amazon Athena의 VACUUM 문을 사용할 수 있습니다. 이 명령문은 스냅샷을 제거하고 분리된 파일도 삭제합니다. Athena는 이 작업의 컴퓨팅 비용을 청구하지 않기 때문에 매우 비용 효율적입니다. 또한 명령문을 사용할 때 추가 작업을 예약할 필요가 없습니다. `VACUUM`

또는 Amazon EMR에서 Spark를 사용하거나 절차를 실행할 `remove_orphan_files` 수 AWS Glue 있습니다. 이 작업에는 컴퓨팅 비용이 발생하며 개별적으로 일정을 잡아야 합니다. 자세한 내용은 Iceberg 설명서를 참조하십시오.

압축을 사용하여 테이블 유지 관리하기

Iceberg에는 테이블에 데이터를 쓴 후 테이블 유지 관리 작업을 수행할 수 있는 기능이 포함되어 있습니다. 일부 유지 관리 작업은 메타데이터 파일을 간소화하는 데 중점을 두는 반면, 쿼리 엔진이 사용자 요청에 응답하는 데 필요한 정보를 효율적으로 찾을 수 있도록 파일 내 데이터 클러스터링 방식을 개선하는 작업도 있습니다. 이 섹션에서는 압축 관련 최적화에 중점을 둡니다.

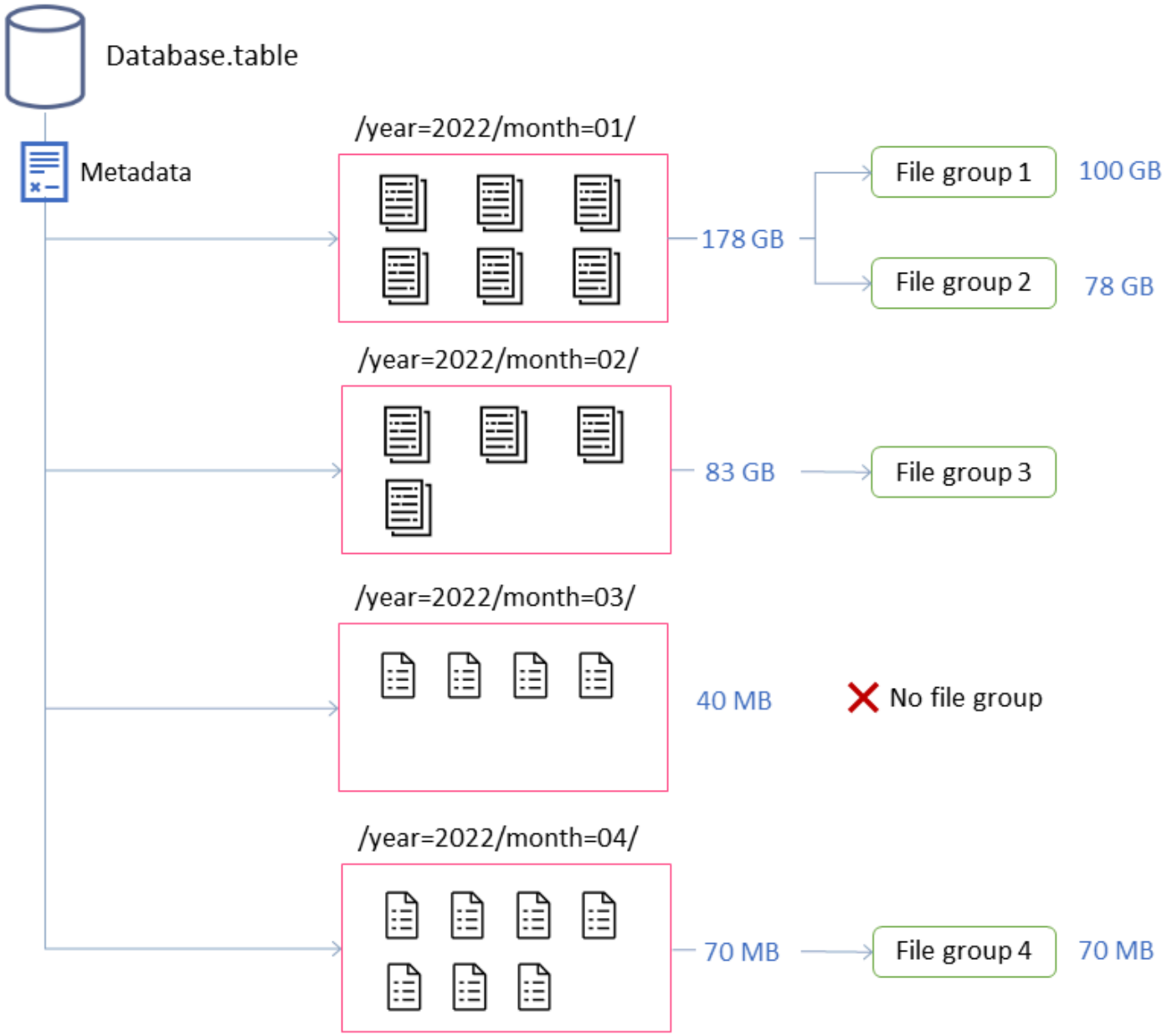
아이스버그 컴팩션

Iceberg에서는 압축을 사용하여 다음 네 가지 작업을 수행할 수 있습니다.

- 작은 파일을 일반적으로 크기가 100MB가 넘는 큰 파일로 결합합니다. 이 기법을 빈 패킹이라고 합니다.
- 삭제 파일을 데이터 파일과 병합 삭제 파일은 이 접근 방식을 사용하는 업데이트 또는 삭제를 통해 생성됩니다. merge-on-read
- 쿼리 패턴에 따라 데이터를 (재) 정렬합니다. 정렬 순서 없이 또는 쓰기 및 업데이트에 적합한 정렬 순서로 데이터를 쓸 수 있습니다.
- 공간을 채우는 곡선을 사용하여 데이터를 클러스터링하여 고유한 쿼리 패턴, 특히 z-순서 정렬에 맞게 최적화합니다.

에서는 AWS Amazon Athena를 통해 또는 Amazon EMR에서 Spark를 사용하여 Iceberg에 대한 테이블 압축 및 유지 관리 작업을 실행할 수 있습니다. AWS Glue

rewrite_data_files 프로시저를 사용하여 압축을 실행하는 경우 여러 개의 노브를 조정하여 압축 동작을 제어할 수 있습니다. 다음 다이어그램은 빈 패킹의 기본 동작을 보여줍니다. 계층 정렬과 Z-순서 정렬 구현은 빈 패킹 인터페이스의 확장이며 유사한 방식으로 작동하기 때문에 빈 패킹 압축을 이해하는 것이 중요합니다. 주된 차이점은 데이터를 정렬하거나 클러스터링하는 데 필요한 추가 단계입니다.



이 예제에서 Iceberg 테이블은 네 개의 파티션으로 구성되어 있습니다. 파티션마다 크기가 다르고 파일 수도 다릅니다. 압축을 실행하기 위해 Spark 애플리케이션을 시작하면 애플리케이션은 처리할 총 4개의 파일 그룹을 만듭니다. 파일 그룹은 단일 Spark 작업으로 처리될 파일 컬렉션을 나타내는 Iceberg 추상화입니다. 즉, 압축을 실행하는 Spark 애플리케이션은 데이터를 처리하는 Spark 작업 4개를 생성합니다.

압축 동작 조정

다음 주요 속성은 압축을 위해 데이터 파일을 선택하는 방법을 제어합니다.

- MAX_FILE_GROUP_SIZE_BYTES는 단일 파일 그룹 (스파크 작업) 에 대한 데이터 제한을 기본적으로 100GB로 설정합니다. 이 속성은 파티션이 없는 테이블이나 수백 기가바이트에 달하는 파티션이 있는 테이블에 특히 중요합니다. 이 제한을 설정하면 작업을 세분화하여 작업을 계획하고 진행하면서 클러스터의 리소스 고갈을 방지할 수 있습니다.

  참고: 각 파일 그룹은 개별적으로 정렬됩니다. 따라서 파티션 수준 정렬을 수행하려면 파티션 크기와 일치하도록 이 제한을 조정해야 합니다.

- MIN_FILE_SIZE_BYTES 또는 MIN_FILE_SIZE_DEFAULT_RATION의 기본값은 테이블 수준에서 설정된 대상 파일 크기의 75% 입니다. 예를 들어, 테이블의 대상 크기가 512MB인 경우 384MB보다 작은 파일은 압축될 파일 세트에 포함됩니다.

- MAX_FILE_SIZE_BYTES 또는 MAX_FILE_SIZE_DEFAULT_RATIO의 기본값은 대상 파일 크기의 180퍼센트입니다. 최소 파일 크기를 설정하는 두 속성과 마찬가지로 이러한 속성은 압축 작업에 사용할 후보 파일을 식별하는 데 사용됩니다.

- MIN_INPUT_FILES는 테이블 파티션 크기가 대상 파일 크기보다 작을 경우 압축할 최소 파일 수를 지정합니다. 이 속성의 값은 파일 수를 기준으로 파일을 압축할 가치가 있는지 여부를 결정하는 데 사용됩니다 (기본값은 5).

- DELETE_FILE_THRESHOLD는 압축에 포함되기 전에 파일에 대한 최소 삭제 작업 수를 지정합니다. 달리 지정하지 않는 한 압축은 삭제 파일을 데이터 파일과 결합하지 않습니다. 이 기능을 사용하려면 이 속성을 사용하여 임계값을 설정해야 합니다. 이 임계값은 개별 데이터 파일에만 적용되므로 이 임계값을 3으로 설정하면 해당 데이터 파일을 참조하는 삭제 파일이 세 개 이상 있는 경우에만 데이터 파일이 다시 작성됩니다.

이러한 속성을 통해 이전 다이어그램의 파일 그룹 구성을 파악할 수 있습니다.

예를 들어 레이블이 지정된 파티션은 최대 크기 제한인 100GB를 초과하므로 파일 그룹이 두 개 `month=01` 포함되어 있습니다. 반면 `month=02` 파티션은 크기가 100GB 미만이므로 단일 파일 그룹을 포함합니다. `month=03`파티션이 파일 5개라는 기본 최소 입력 파일 요구 사항을 충족하지 않습니다. 따라서 압축되지 않습니다. 마지막으로, `month=04` 파티션에 원하는 크기의 파일 하나를 만들 수 있는 데이터가 충분하지 않더라도 파티션에 작은 파일이 5개 이상 포함되어 있기 때문에 파일이 압축됩니다.

Amazon AWS Glue EMR에서 실행되는 Spark 또는 에 대해 이러한 파라미터를 설정할 수 있습니다. Amazon Athena의 경우 접두사로 `optimize_` 시작하는 테이블 속성을 사용하여 유사한 속성을 관리할 수 있습니다.

Amazon EMR에서 Spark를 사용하여 컴팩션을 실행하거나 AWS Glue

이 섹션에서는 Iceberg의 압축 유틸리티를 실행하기 위해 Spark 클러스터의 크기를 적절하게 조정하는 방법을 설명합니다. 다음 예시에서는 Amazon EMR 서버리스를 사용하지만, Amazon EC2의 Amazon EMR 또는 Amazon EKS에서도 동일한 방법을 사용할 수 있습니다. AWS Glue

파일 그룹과 Spark 작업 간의 상관 관계를 활용하여 클러스터 리소스를 계획할 수 있습니다. 파일 그룹당 최대 크기가 100GB인 것을 고려하여 파일 그룹을 순차적으로 처리하려면 다음과 같은 Spark 속성을 설정하면 됩니다.

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

압축 속도를 높이려면 병렬로 압축되는 파일 그룹 수를 늘려 수평으로 확장할 수 있습니다. 수동 또는 동적 조정을 사용하여 Amazon EMR을 확장할 수도 있습니다.

- 수동 조정 (예: 4배)
  - `MAX_CONCURRENT_FILE_GROUP_REWRITES= 4` (우리 계수)
  - `spark.executor.instances= 5` (예제에 사용된 값) x `4` (우리 계수) = `20`
  - `spark.dynamicAllocation.enabled = FALSE`
- 동적 스케일링
  - `spark.dynamicAllocation.enabled= TRUE` (기본값, 조치 필요 없음)
  - `MAX_CONCURRENT_FILE_GROUP_REWRITES = N` (이 값을 기본값인 100에 맞춥니다. 예제의 실행자 구성에 따라 20으로 `spark.dynamicAllocation.maxExecutors` 설정할 수 있습니다.) `N`

  다음은 클러스터 크기를 조정하는 데 도움이 되는 지침입니다. 하지만 워크로드에 가장 적합한 설정을 찾으려면 Spark 작업의 성능도 모니터링해야 합니다.

Amazon Athena로 컴팩션 실행하기

Athena는 OPTIMIZE 선언문을 통해 Iceberg의 압축 유틸리티 구현을 관리형 기능으로 제공합니다. 이 명령문을 사용하면 인프라를 평가할 필요 없이 컴팩션을 실행할 수 있습니다.

이 명령문은 빈 패킹 알고리즘을 사용하여 작은 파일을 큰 파일로 그룹화하고 삭제 파일을 기존 데이터 파일에 병합합니다. 계층적 정렬 또는 z-순서 정렬을 사용하여 데이터를 클러스터링하려면 Amazon EMR의 Spark 또는 을 사용하십시오. AWS Glue

`OPTIMIZE`명령문에 테이블 속성을 전달하여 테이블 생성 시 또는 테이블 생성 후 `CREATE TABLE` 명령문을 사용하여 명령문의 기본 동작을 변경할 수 있습니다. `ALTER TABLE` 기본값은 Athena 설명서를 참조하십시오.

컴팩션 실행을 위한 권장 사항

사용 사례	권장 사항
일정에 따라 쓰레기통 포장 압축 실행	• 테이블에 몇 개의 작은 파일이 들어 있는지 모르는 경우 Athena의 `OPTIMIZE` 명령문을 사용하십시오. Athena의 가격 책정 모델은 스캔한 데이터를 기반으로 하므로 압축할 파일이 없는 경우 이러한 작업과 관련된 비용이 발생하지 않습니다. Athena 테이블에서 타임아웃이 발생하지 않도록 하려면 베이스를 기준으로 실행하십시오. `OPTIMIZE` per-table-partition • 대용량의 작은 파일이 압축될 것으로 예상되는 경우 Amazon EMR을 사용하거나 동적 크기 조정과 AWS Glue 함께 사용하십시오.
이벤트를 기반으로 빈 패킹 압축 실행	• 대용량의 작은 파일이 압축될 것으로 예상되는 경우 Amazon EMR을 사용하거나 동적 크기 조정과 AWS Glue 함께 사용하십시오.
컴팩션을 실행하여 데이터를 정렬합니다.	• 정렬은 비용이 많이 들고 데이터를 AWS Glue 디스크로 넘겨야 할 수도 있으므로 Amazon EMR을 사용하거나 사용하십시오.
압축을 실행하여 z-순서 정렬을 사용하여 데이터를 클러스터링합니다.	• Amazon EMR을 사용하거나 AWS Glue, z-order 정렬은 비용이 많이 들고 데이터를 디스크로 넘겨야 할 수도 있으므로 Amazon EMR을 사용하십시오.

사용 사례	권장 사항
늦게 도착하는 데이터로 인해 다른 애플리케이션에서 업데이트할 수 있는 파티션에서 컴팩션을 실행합니다.	• Amazon EMR을 사용하거나. AWS Glue아이스버그 PARTIAL_PROGRESS_ENABLED 속성을 활성화하십시오. 이 옵션을 사용하면 Iceberg는 압축 출력을 여러 커밋으로 분할합니다. 충돌이 발생하는 경우 (즉, 압축이 실행되는 동안 데이터 파일이 업데이트되는 경우) 이 설정은 영향을 받는 파일이 포함된 커밋으로 제한하여 재시도 비용을 줄여줍니다. 그렇지 않으면 모든 파일을 다시 압축해야 할 수도 있습니다.

Amazon S3에서 아이스버그 워크로드 사용

이 섹션에서는 Iceberg와 Amazon S3의 상호 작용을 최적화하는 데 사용할 수 있는 Iceberg 속성에 대해 설명합니다.

핫 파티셔닝 방지 (HTTP 503 오류)

Amazon S3에서 실행되는 일부 데이터 레이크 애플리케이션은 수백만 또는 수십억 개의 객체를 처리하고 페타바이트 규모의 데이터를 처리합니다. 이로 인해 많은 양의 트래픽을 수신하는 접두사가 생길 수 있으며, 이러한 접두사는 일반적으로 HTTP 503 (서비스를 사용할 수 없음) 오류를 통해 감지됩니다. 이 문제를 방지하려면 다음 Iceberg 속성을 사용하세요.

- Iceberg가 대용량 파일을 `write.distribution-mode` `range` 쓰도록 `hash` 설정하거나 설정하여 Amazon S3 요청 수를 줄입니다. 이 구성이 선호되며 대부분의 경우를 해결해야 합니다.
- 워크로드의 엄청난 양의 데이터로 인해 503 오류가 계속 발생하는 경우 Iceberg에서 설정할 `write.object-storage.enabled` 수 있습니다. `true` 그러면 Iceberg가 객체 이름을 해시하고 여러 개의 무작위 Amazon S3 접두사에 부하를 분산하도록 지시합니다.

이러한 속성에 대한 자세한 내용은 Iceberg 설명서의 속성 쓰기를 참조하십시오.

Iceberg 유지 관리 작업을 사용하여 사용하지 않는 데이터를 공개하세요.

Iceberg 테이블을 관리하려면 Iceberg 코어 API, Iceberg 클라이언트 (예: Spark) 또는 Amazon Athena와 같은 관리형 서비스를 사용할 수 있습니다. Amazon S3에서 오래된 파일이나 사용하지 않는 파일을

삭제하려면 Iceberg 네이티브 API만 사용하여 스냅샷을 제거하고, 오래된 메타데이터 파일을 제거하고, 분리된 파일을 삭제하는 것이 좋습니다.

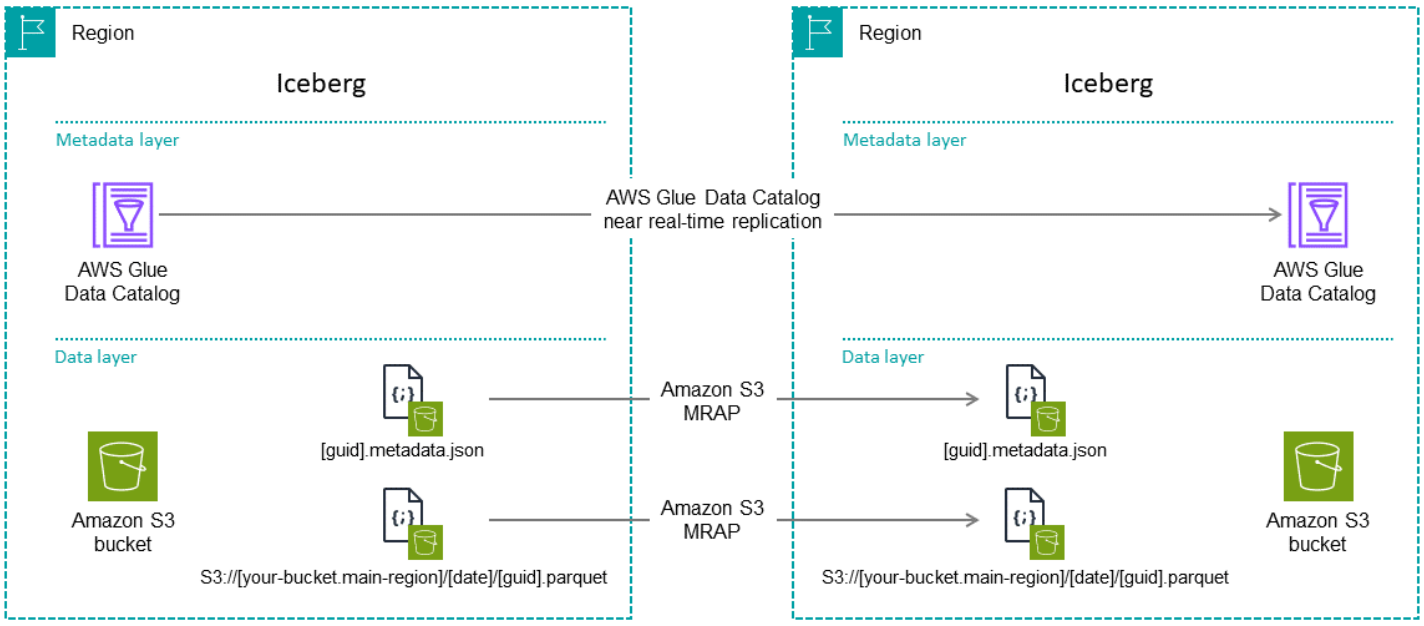
Boto3, Amazon S3 SDK 또는 AWS Command Line Interface (AWS CLI) 를 통해 Amazon S3 API를 사용하거나 다른 비빙산 메서드를 사용하여 Iceberg 테이블의 Amazon S3 파일을 덮어쓰거나 제거하면 테이블이 손상되고 쿼리가 실패합니다.

데이터를 여러 곳에 복제하십시오. AWS 리전

Amazon S3에 Iceberg 테이블을 저장하면 지역 간 복제 (CRR) 및 다중 지역 액세스 포인트 (MRAP) 와 같은 Amazon S3에 내장된 기능을 사용하여 여러 AWS 지역에 데이터를 복제할 수 있습니다. MRAP는 애플리케이션이 여러 곳에 위치한 S3 버킷에 액세스할 수 있는 글로벌 엔드포인트를 제공합니다. AWS 리전 Iceberg는 상대 경로를 지원하지 않지만 MRAP를 사용하여 버킷을 액세스 포인트에 매핑하여 Amazon S3 작업을 수행할 수 있습니다. 또한 MRAP는 Amazon S3 지역 간 복제 프로세스와도 원활하게 통합되므로 최대 15분의 지연이 발생합니다. 데이터와 메타데이터 파일을 모두 복제해야 합니다.

> **Important**
>
> 현재 아이스버그와 MRAP의 통합은 Apache Spark에서만 작동합니다. AWS 리전보조로 페일오버해야 하는 경우 사용자 쿼리를 장애 조치 지역의 Spark SQL 환경 (예: Amazon EMR) 으로 리디렉션할 계획을 세워야 합니다.

CRR 및 MRAP 기능은 다음 다이어그램에 나와 있는 것처럼 Iceberg 테이블을 위한 지역 간 복제 솔루션을 구축하는 데 도움이 됩니다.



이 지역 간 복제 아키텍처를 설정하려면:

1. MRAP 위치를 사용하여 테이블을 생성합니다. 이렇게 하면 Iceberg 메타데이터 파일이 물리적 버킷 위치 대신 MRAP 위치를 가리키게 됩니다.
2. Amazon S3 MRAP를 사용하여 아이스버그 파일을 복제합니다. MRAP는 15분의 서비스 수준 계약(SLA) 으로 데이터 복제를 지원합니다. Iceberg는 복제 중에 읽기 작업으로 인해 불일치가 발생하는 것을 방지합니다.
3. 보조 지역에서 테이블을 사용할 수 있도록 하세요. AWS Glue Data Catalog 다음 두 가지 옵션 중에서 선택할 수 있습니다.
   - 복제를 사용하여 AWS Glue Data Catalog Iceberg 테이블 메타데이터를 복제하기 위한 파이프라인을 설정하세요. 이 유틸리티는 GitHub Glue 카탈로그 및 Lake Formation 권한 복제 리포지토리에서 사용할 수 있습니다. 이 이벤트 기반 메커니즘은 이벤트 로그를 기반으로 대상 지역의 테이블을 복제합니다.
   - 페일오버가 필요한 경우 보조 리전에 테이블을 등록하십시오. 이 옵션의 경우 이전 유틸리티 또는 Iceberg register_table 프로시저를 사용하여 최신 파일을 가리킬 수 있습니다. `metadata.json`

아파치 아이스버그 워크로드 모니터링

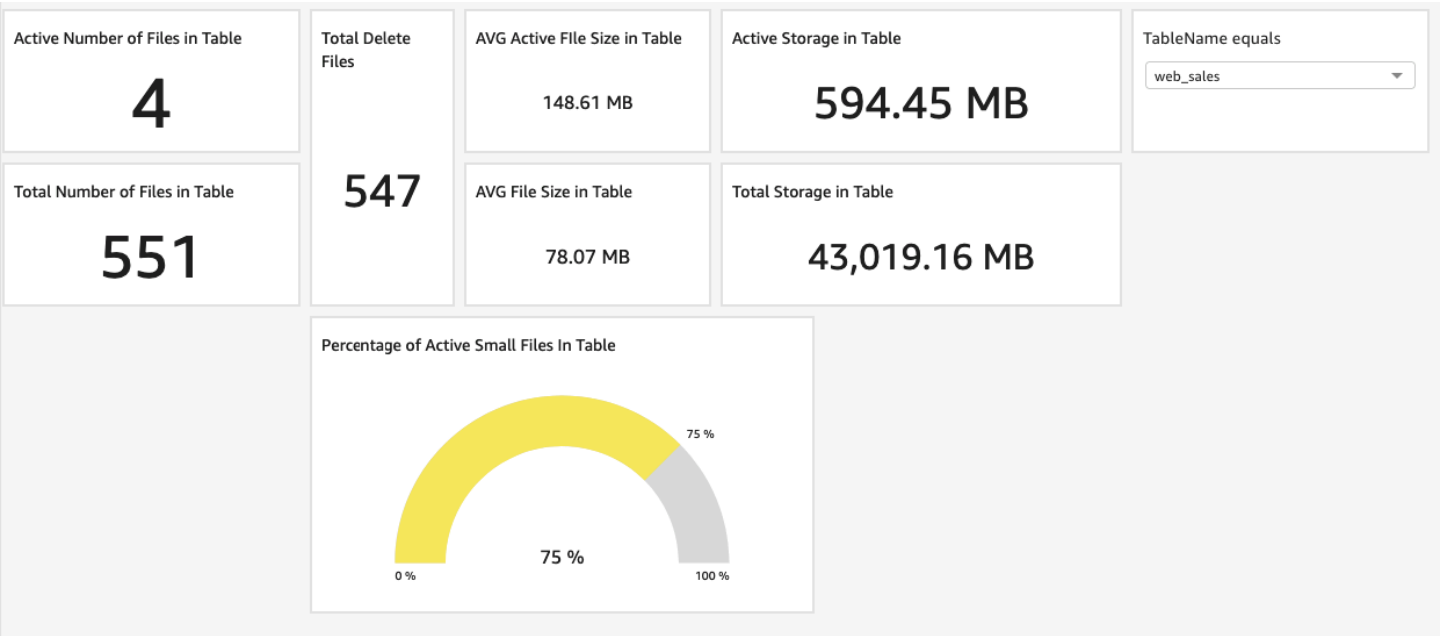
Iceberg 워크로드를 모니터링하려면 메타데이터 테이블을 분석하거나 메트릭 리포터를 사용하는 두 가지 옵션이 있습니다. 지표 리포터는 Iceberg 버전 1.2에 도입되었으며 REST 및 JDBC 카탈로그에서만 사용할 수 있습니다.

를 사용하는 AWS Glue Data Catalog경우 Iceberg가 공개하는 메타데이터 테이블 위에 모니터링을 설정하여 Iceberg 테이블의 상태를 파악할 수 있습니다.

모니터링은 성능 관리 및 문제 해결에 매우 중요합니다. 예를 들어 Iceberg 테이블의 파티션이 일정 비율의 작은 파일에 도달하면 워크로드가 압축 작업을 시작하여 파일을 더 큰 파일로 통합할 수 있습니다. 이렇게 하면 쿼리 속도가 허용 수준 이상으로 느려지는 것을 방지할 수 있습니다.

테이블 수준 모니터링

다음 화면은 Amazon에서 만든 테이블 모니터링 대시보드를 보여줍니다 QuickSight. 이 대시보드는 Spark SQL을 사용하여 Iceberg 메타데이터 테이블을 쿼리하고 활성 파일 수 및 총 스토리지와 같은 세부 지표를 캡처합니다. 그런 다음 이 정보는 운영 목적으로 AWS Glue 테이블에 저장됩니다. 마지막으로, 다음 그림과 같이 Amazon Athena를 사용하여 QuickSight 대시보드를 생성합니다. 이 정보는 시스템의 특정 문제를 식별하고 해결하는 데 도움이 됩니다.



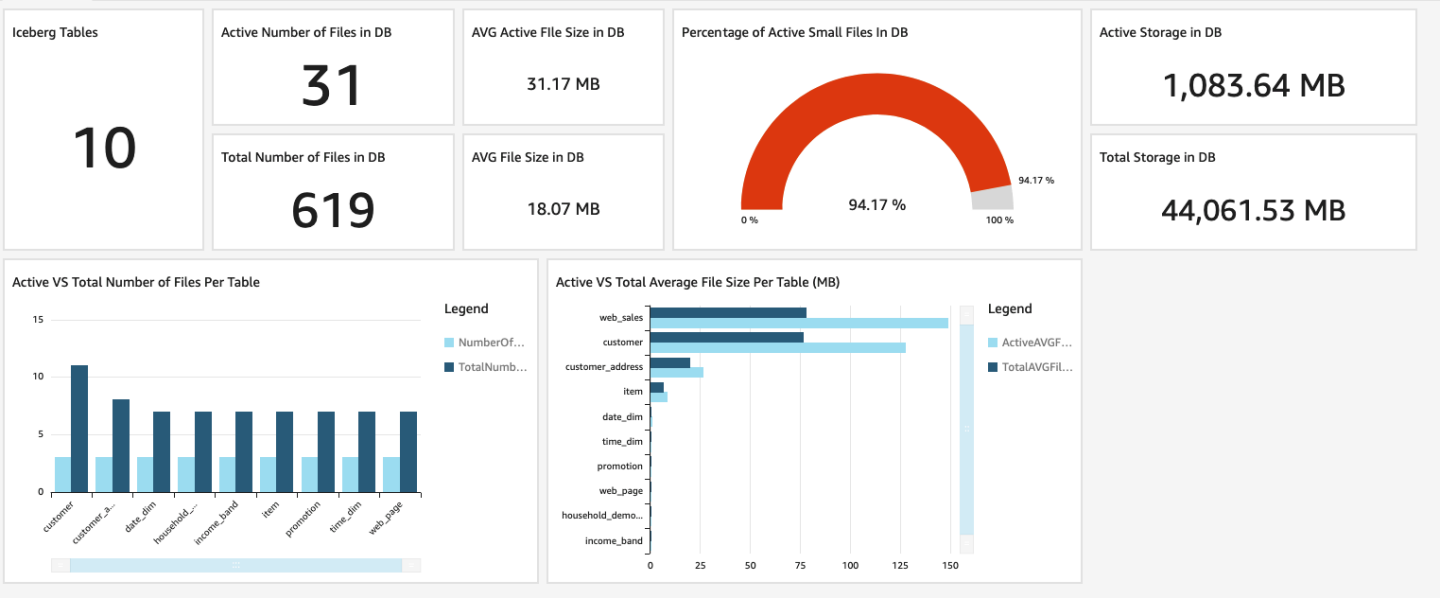
예제 QuickSight 대시보드는 Iceberg 테이블에 대한 다음과 같은 핵심 성과 지표 (KPI) 를 수집합니다.

KPI	설명	Query
파일 개수	Iceberg 테이블의 파일 수 (모든 스냅샷의 경우)	`select count(*) from <catalog.database.table_name>.all_files`
활성 파일 수	Iceberg 테이블의 마지막 스냅샷에 있는 활성 파일 수	`select count(*) from <catalog.database.table_name>.files`
평균 파일 크기	Iceberg 테이블에 있는 모든 파일의 평균 파일 크기 (메가바이트)	`select avg(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`
평균 활성 파일 크기	Iceberg 테이블에 있는 활성 파일의 평균 파일 크기 (메가바이트)	`select avg(file_size_in_bytes)/1000000 from <catalog.database.table_name>.files`
작은 파일의 비율	활성 파일 중 크기가 100MB 미만인 파일의 비율	`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`
총 스토리지 크기	분리된 파일과 Amazon S3 객체 버전을 제외한 테이블에 있는 모든 파일의 총 크기 (활성화된 경우)	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`

KPI	설명	Query
총 활성 스토리지 크기	지정된 테이블의 현재 스냅샷에 있는 모든 파일의 총 크기	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.files`

데이터베이스 수준 모니터링

다음 예는 Iceberg 테이블 컬렉션의 데이터베이스 수준 KPI에 대한 개요를 제공하기 QuickSight 위해 생성된 모니터링 대시보드를 보여줍니다.



이 대시보드는 다음과 같은 KPI를 수집합니다.

KPI	설명	Query
파일 개수	Iceberg 데이터베이스의 파일 수 (모든 스냅샷의 경우)	이 대시보드는 이전 섹션에 제공된 테이블 수준의 쿼리를 사용하고 결과를 통합합니다.
활성 파일 수	Iceberg 데이터베이스의 활성 파일 수 (Iceberg 테이블의 마지막 스냅샷 기준)	

KPI	설명	Query
평균 파일 크기	Iceberg 데이터베이스에 있는 모든 파일의 평균 파일 크기 (메가바이트)	
평균 활성 파일 크기	Iceberg 데이터베이스의 모든 활성 파일에 대한 평균 파일 크기 (메가바이트)	
작은 파일의 비율	Iceberg 데이터베이스에서 100MB 미만의 활성 파일 비율	
총 스토리지 크기	데이터베이스에 있는 모든 파일의 총 크기 (분리된 파일 및 Amazon S3 객체 버전 제외) (활성화된 경우)	
총 활성 스토리지 크기	데이터베이스에 있는 모든 테이블의 현재 스냅샷에 있는 모든 파일의 총 크기	

예방 유지 관리

이전 섹션에서 설명한 모니터링 기능을 설정하면 사후 대응이 아닌 예방 차원에서 테이블 유지 관리에 접근할 수 있습니다. 예를 들어, 테이블 수준 및 데이터베이스 수준 지표를 사용하여 다음과 같은 작업을 예약할 수 있습니다.

- 테이블이 N개의 작은 파일에 도달하면 빈 패킹 압축을 사용하여 작은 파일을 그룹화할 수 있습니다.
- 테이블이 지정된 파티션에서 N개의 삭제 파일에 도달하면 빈 패킹 압축을 사용하여 삭제 파일을 병합할 수 있습니다.
- 총 스토리지가 활성 스토리지보다 X배 많으면 스냅샷을 제거하여 이미 압축된 작은 파일을 제거합니다.

아파치 아이스버그의 거버넌스 및 액세스 제어 켜기 AWS

Apache Iceberg는 와 AWS Lake Formation 통합되어 데이터 거버넌스를 단순화합니다. 이 통합을 통해 데이터 레이크 관리자는 Iceberg 테이블에 셀 수준의 액세스 권한을 할당할 수 있습니다. Amazon Athena를 사용하여 Iceberg 테이블을 쿼리하는 예제는 Amazon Athena를 사용하여 Apache Iceberg 테이블과 AWS Lake Formation상호 작용하고 사용하는 교차 계정 세분화된 권한을 사용하는 AWS 블로그 게시물을 참조하십시오. AWS Lake Formation

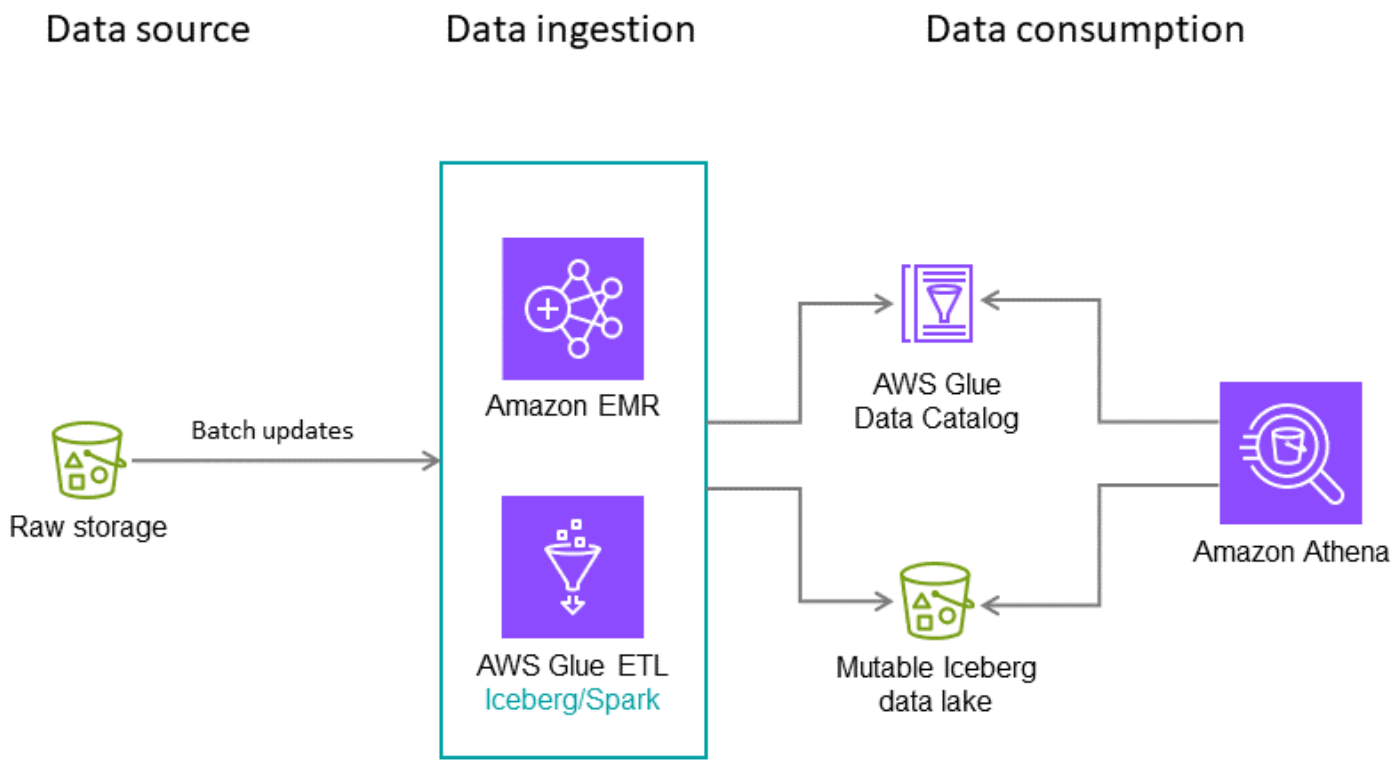
아파치 아이스버그의 참조 아키텍처 on AWS

이 섹션에서는 배치 통합, 일괄 처리 및 스트리밍 데이터 통합을 결합한 데이터 레이크와 같은 다양한 사용 사례에서 모범 사례를 적용하는 방법에 대한 예를 제공합니다.

야간 일괄 처리

이 가상의 사용 사례에서 Iceberg 테이블에서 매일 밤 단위로 신용 카드 거래를 처리한다고 가정해 보겠습니다. 각 배치에는 대상 테이블에 병합해야 하는 증분 업데이트만 포함됩니다. 일 년에 몇 번씩 전체 기록 데이터가 수신됩니다. 이 시나리오에서는 다음과 같은 아키텍처 및 구성을 사용하는 것이 좋습니다.

참고: 이는 예시일 뿐입니다. 최적의 구성은 데이터 및 요구 사항에 따라 달라집니다.



권장 사항:

- Apache Spark 태스크는 128MB 청크 단위로 데이터를 처리하기 때문에 파일 크기: 128MB입니다.
- 쓰기 유형:. copy-on-write 이 가이드의 앞부분에서 자세히 설명했듯이 이 접근 방식은 데이터를 읽기 최적화된 방식으로 작성하는 데 도움이 됩니다.

- 파티션 변수: 년/월/일. 가상의 사용 사례에서는 최근 데이터를 가장 자주 쿼리하지만 지난 2년간의 데이터에 대해 전체 테이블 스캔을 실행하는 경우도 있습니다. 파티셔닝의 목표는 사용 사례의 요구 사항에 따라 빠른 읽기 작업을 유도하는 것입니다.
- 정렬 순서: 타임스탬프
- 데이터 카탈로그: AWS Glue Data Catalog

일괄 처리와 거의 실시간 수집을 결합한 데이터 레이크

Amazon S3에 데이터 레이크를 프로비저닝하여 계정 및 지역 간에 배치 및 스트리밍 데이터를 공유할 수 있습니다. 아키텍처 다이어그램과 세부 정보는 AWS 블로그 게시물 Apache Iceberg를 사용한 트랜잭션 데이터 레이크 구축 및 Amazon Athena를 사용한 계정 간 데이터 공유를 참조하십시오. AWS Glue AWS Lake Formation

리소스

- (설명서) 의 아이스버그 프레임워크 사용 AWS GlueAWS Glue
- 아이스버그 (아마존 EMR 설명서)
- 아파치 아이스버그 테이블 사용 (Amazon Athena 설명서)
- Amazon S3 설명서
- 아마존 QuickSight 설명서
- Glue 카탈로그 및 Lake Formation 권한 복제 (GitHub 리포지토리)
- 아파치 아이스버그 문서
- 아파치 스파크 문서

기여자

이 안내서를 작성, AWS 공동 작성 및 검토한 사람은 다음과 같습니다.

기여자

- 카를로스 로드리게스, 빅데이터 솔루션 아키텍트
- 임티아즈 (타즈) 세이드, 솔루션스 아키텍트 테크 리더, 애널리틱스
- 샤나 스키퍼스, 솔루션스 아키텍트, 빅 데이터
- 프라샨트 싱, 소프트웨어 개발 엔지니어, Amazon EMR
- 스테파노 산도나, 솔루션스 아키텍트, 빅 데이터
- Arun A K, 솔루션스 아키텍트, 빅데이터 및 ETL
- 프란시스코 모릴로, 솔루션스 아키텍트, 스트리밍
- 수단 필립스, 애널리틱스 아키텍트, Amazon EMR
- 세르칸 가라오글루, 솔루션스 아키텍트
- 요나탄 돌란, 애널리틱스 스페셜리스트
- 가이 바카르, 솔루션스 아키텍트
- 소피아 질버만, 솔루션스 아키텍트, 스트리밍
- 이스마일 마클로프, 솔루션스 아키텍트, 애널리틱스
- 댄 스테어, 스페셜리스트 솔루션 아키텍트
- 삭티 미슈라, 솔루션스 아키텍트

리뷰어

- 릭 시어스, 아마존 EMR 제너럴 매니저
- 린다 오코너, 아마존 EMR 스페셜리스트
- 이안 메이어스, 아마존 EMR 디렉터
- 비니타 아난스, 아마존 EMR 제품 관리 디렉터
- 제이슨 버코위츠, 프로덕트 매니저, AWS Lake Formation
- 마헤시 미슈라, 아마존 Redshift 프로덕트 매니저
- 블라디미르 즐라트킨, 빅데이터 솔루션 아키텍처 담당 매니저
- 카르틱 프라바카르, 애널리틱스 아키텍트, Amazon EMR

- 잭 예, 소프트웨어 개발 엔지니어, Amazon EMR
- 비제이 제인, 프로덕트 매니저
- 아누프리티 워레이드, 아마존 S3 프로덕트 매니저
- 몰리 브라운, 제너럴 매니저, AWS Lake Formation
- 아짓 탄데일, 솔루션스 아키텍트, 데이터
- 그웬 첸, 제품 마케팅 매니저

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
최초 게시	—	2024년 4월 30일

AWS 규범적 지침 용어집

다음은 규범적 지침에서 제공하는 AWS 전략, 가이드 및 패턴에서 일반적으로 사용되는 용어입니다. 용어집 항목을 제안하려면 용어집 끝에 있는 피드백 제공 링크를 사용하십시오.

숫자

7가지 전략

애플리케이션을 클라우드로 이전하기 위한 7가지 일반적인 마이그레이션 전략 이러한 전략은 Gartner가 2011년에 파악한 5가지 전략을 기반으로 하며 다음으로 구성됩니다.

- 리팩터링/리아키텍트 - 클라우드 네이티브 기능을 최대한 활용하여 애플리케이션을 이동하고 해당 아키텍처를 수정함으로써 민첩성, 성능 및 확장성을 개선합니다. 여기에는 일반적으로 운영 체제와 데이터베이스 이식이 포함됩니다. 예: 온프레미스 Oracle 데이터베이스를 Amazon Aurora PostgreSQL 호환 에디션으로 마이그레이션하십시오.
- 리플랫포밍(리프트 앤드 리셰이프) - 애플리케이션을 클라우드로 이동하고 일정 수준의 최적화를 도입하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 오라클용 Amazon RDS (Amazon RDS) 로 마이그레이션합니다. AWS 클라우드
- 재구매(드롭 앤드 숍) - 일반적으로 기존 라이선스에서 SaaS 모델로 전환하여 다른 제품으로 전환합니다. 예: 고객 관계 관리 (CRM) 시스템을 Salesforce.com으로 마이그레이션하십시오.
- 리호스팅(리프트 앤드 시프트) - 애플리케이션을 변경하지 않고 클라우드로 이동하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 의 EC2 인스턴스에서 Oracle로 마이그레이션합니다. AWS 클라우드
- 재배치(하이퍼바이저 수준의 리프트 앤 시프트) - 새 하드웨어를 구매하거나, 애플리케이션을 다시 작성하거나, 기존 운영을 수정하지 않고도 인프라를 클라우드로 이동합니다. 온프레미스 플랫폼에서 동일한 플랫폼의 클라우드 서비스로 서버를 마이그레이션합니다. 예: Microsoft Hyper-V 애플리케이션을 다음으로 마이그레이션하십시오. AWS
- 유지(보관) - 소스 환경에 애플리케이션을 유지합니다. 대규모 리팩터링이 필요하고 해당 작업을 나중으로 연기하려는 애플리케이션과 비즈니스 차원에서 마이그레이션할 이유가 없어 유지하려는 레거시 애플리케이션이 여기에 포함될 수 있습니다.
- 사용 중지 - 소스 환경에서 더 이상 필요하지 않은 애플리케이션을 폐기하거나 제거합니다.

A

ABAC

속성 기반 액세스 제어를 참조하십시오.

추상화된 서비스

관리형 서비스를 참조하십시오.

산

원자성, 일관성, 격리성, 내구성을 참조하십시오.

능동-능동 마이그레이션

양방향 복제 도구 또는 이중 쓰기 작업을 사용하여 소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되고, 두 데이터베이스 모두 마이그레이션 중 연결 애플리케이션의 트랜잭션을 처리하는 데이터베이스 마이그레이션 방법입니다. 이 방법은 일회성 전환이 필요한 대신 소규모의 제어된 배치로 마이그레이션을 지원합니다. 더 유연하지만 액티브-패시브 마이그레이션보다 더 많은 작업이 필요합니다.

능동-수동 마이그레이션

소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되지만 소스 데이터베이스만 연결 애플리케이션의 트랜잭션을 처리하고 데이터는 대상 데이터베이스로 복제되는 데이터베이스 마이그레이션 방법입니다. 대상 데이터베이스는 마이그레이션 중 어떤 트랜잭션도 허용하지 않습니다.

집계 함수

행 그룹에서 연산을 수행하고 그룹에 대한 단일 반환값을 계산하는 SQL 함수입니다. 집계 함수의 예로는 및 등이 `SUM` 있습니다. `MAX`

AI

인공 지능을 참조하십시오.

AIOps

인공 지능 운영을 참조하십시오.

익명화

데이터세트에서 개인 정보를 영구적으로 삭제하는 프로세스입니다. 익명화는 개인 정보 보호에 도움이 될 수 있습니다. 익명화된 데이터는 더 이상 개인 데이터로 간주되지 않습니다.

안티 패턴

솔루션이 다른 솔루션보다 비생산적이거나 비효율적이거나 덜 효과적이어서 반복되는 문제에 자주 사용되는 솔루션입니다.

애플리케이션 제어

시스템을 멀웨어로부터 보호하기 위해 승인된 애플리케이션만 사용할 수 있는 보안 접근 방식입니다.

애플리케이션 포트폴리오

애플리케이션 구축 및 유지 관리 비용과 애플리케이션의 비즈니스 가치를 비롯하여 조직에서 사용하는 각 애플리케이션에 대한 세부 정보 모음입니다. 이 정보는 포트폴리오 검색 및 분석 프로세스의 핵심이며 마이그레이션, 현대화 및 최적화할 애플리케이션을 식별하고 우선순위를 정하는 데 도움이 됩니다.

인공 지능

컴퓨터 기술을 사용하여 학습, 문제 해결, 패턴 인식 등 일반적으로 인간과 관련된 인지 기능을 수행하는 것을 전문으로 하는 컴퓨터 과학 분야입니다. 자세한 내용은 What is Artificial Intelligence?를 참조하십시오.

인공 지능 운영(AIOps)

기계 학습 기법을 사용하여 운영 문제를 해결하고, 운영 인시던트 및 사용자 개입을 줄이고, 서비스 품질을 높이는 프로세스입니다. AWS 마이그레이션 전략에서 AIOps가 사용되는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

비대칭 암호화

한 쌍의 키, 즉 암호화를 위한 퍼블릭 키와 복호화를 위한 프라이빗 키를 사용하는 암호화 알고리즘입니다. 퍼블릭 키는 복호화에 사용되지 않으므로 공유할 수 있지만 프라이빗 키에 대한 액세스는 엄격히 제한되어야 합니다.

원자성, 일관성, 격리성, 내구성(ACID)

오류, 정전 또는 기타 문제가 발생한 경우에도 데이터베이스의 데이터 유효성과 운영 신뢰성을 보장하는 소프트웨어 속성 세트입니다.

ABAC(속성 기반 액세스 제어)

부서, 직무, 팀 이름 등의 사용자 속성을 기반으로 세분화된 권한을 생성하는 방식입니다. 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 AWS ABAC for를 참조하십시오.

신뢰할 수 있는 데이터 소스

가장 신뢰할 수 있는 정보 소스로 간주되는 기본 버전의 데이터를 저장하는 위치입니다. 익명화, 편집 또는 가명화와 같은 데이터 처리 또는 수정의 목적으로 신뢰할 수 있는 데이터 소스의 데이터를 다른 위치로 복사할 수 있습니다.

가용 영역

다른 가용 영역의 장애로부터 격리되고 동일한 지역 내 다른 가용 영역에 저렴하고 지연 시간이 짧은 네트워크 연결을 제공하는 별도의 위치. AWS 리전

AWS 클라우드 채택 프레임워크 (AWS CAF)

조직이 클라우드로 성공적으로 AWS 전환하기 위한 효율적이고 효과적인 계획을 개발하는 데 도움이 되는 지침 및 모범 사례 프레임워크입니다. AWS CAF는 지침을 관점이라고 하는 6가지 중점 영역, 즉 비즈니스, 사람, 거버넌스, 플랫폼, 보안, 운영으로 분류합니다. 비즈니스, 사람 및 거버넌스 관점은 비즈니스 기술과 프로세스에 초점을 맞추고, 플랫폼, 보안 및 운영 관점은 전문 기술과 프로세스에 중점을 둡니다. 예를 들어, 사람 관점은 인사(HR), 직원 배치 기능 및 인력 관리를 담당하는 이해관계자를 대상으로 합니다. 이러한 관점에서 AWS CAF는 조직이 성공적인 클라우드 채택을 준비할 수 있도록 인력 개발, 교육 및 커뮤니케이션에 대한 지침을 제공합니다. 자세한 내용은 AWS CAF 웹 사이트와 AWS CAF 백서를 참조하십시오.

AWS 워크로드 검증 프레임워크 (AWS WQF)

데이터베이스 마이그레이션 워크로드를 평가하고 마이그레이션 전략을 권장하며 작업 예상치를 제공하는 도구입니다. AWS WQF는 () 에 포함됩니다. AWS Schema Conversion Tool AWS SCT데이터베이스 스키마 및 코드 객체, 애플리케이션 코드, 종속성 및 성능 특성을 분석하고 평가 보고서를 제공합니다.

B

배드 봇

개인이나 조직을 방해하거나 피해를 입히려는 의도를 가진 봇입니다.

BCP

비즈니스 연속성 계획을 참조하십시오.

동작 그래프

리소스 동작과 시간 경과에 따른 상호 작용에 대한 통합된 대화형 뷰입니다. Amazon Detective에서 동작 그래프를 사용하여 실패한 로그온 시도, 의심스러운 API 호출 및 유사한 작업을 검사할 수 있습니다. 자세한 내용은 Detective 설명서의 Data in a behavior graph를 참조하십시오.

빅 엔디안 시스템

가장 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하십시오.

바이너리 분류

바이너리 결과(가능한 두 클래스 중 하나)를 예측하는 프로세스입니다. 예를 들어, ML 모델이 “이 이메일이 스팸인가요, 스팸이 아닌가요?”, '이 제품은 책인가요, 자동차인가요?' 등의 문제를 예측해야 할 수 있습니다.

블룸 필터

요소가 세트의 멤버인지 여부를 테스트하는 데 사용되는 메모리 효율성이 높은 확률론적 데이터 구조입니다.

블루/그린(Blue/Green) 배포

서로 다르지만 동일한 환경을 두 개 만드는 배포 전략입니다. 현재 애플리케이션 버전을 한 환경 (파란색) 에서 실행하고 다른 환경 (녹색) 에서 새 애플리케이션 버전을 실행합니다. 이 전략을 사용하면 영향을 최소화하면서 신속하게 롤백할 수 있습니다.

bot

인터넷을 통해 자동화된 작업을 실행하고 사람의 활동이나 상호 작용을 시뮬레이션하는 소프트웨어 애플리케이션입니다. 인터넷에서 정보를 인덱싱하는 웹 크롤러와 같은 일부 봇은 유용하거나 유용합니다. 배드 봇으로 알려진 일부 다른 봇은 개인이나 조직을 방해하거나 피해를 입히기 위한 것입니다.

봇넷

멀웨어에감염되어 봇 허더 또는 봇 운영자로 알려진 단일 당사자의 통제 하에 있는 봇 네트워크. 봇넷은 봇과 그 영향을 확장하는 가장 잘 알려진 메커니즘입니다.

브랜치

코드 리포지토리의 포함된 영역입니다. 리포지토리에 생성되는 첫 번째 브랜치가 기본 브랜치입니다. 기존 브랜치에서 새 브랜치를 생성한 다음 새 브랜치에서 기능을 개발하거나 버그를 수정할 수 있습니다. 기능을 구축하기 위해 생성하는 브랜치를 일반적으로 기능 브랜치라고 합니다. 기능을 출시할 준비가 되면 기능 브랜치를 기본 브랜치에 다시 병합합니다. 자세한 내용은 브랜치 정보 (문서) 를 참조하십시오. GitHub

브레이크 글래스 액세스

예외적인 상황에서 승인된 프로세스를 통해 사용자가 일반적으로 액세스 권한이 없는 데이터에 빠르게 액세스할 수 AWS 계정 있는 수단입니다. 자세한 내용은 Well-Architected AWS 지침의 브레이크 글래스 절차 구현 표시기를 참조하십시오.

브라운필드 전략

사용자 환경의 기존 인프라 시스템 아키텍처에 브라운필드 전략을 채택할 때는 현재 시스템 및 인프라의 제약 조건을 중심으로 아키텍처를 설계합니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

버퍼 캐시

가장 자주 액세스하는 데이터가 저장되는 메모리 영역입니다.

사업 역량

기업이 가치를 창출하기 위해 하는 일(예: 영업, 고객 서비스 또는 마케팅)입니다. 마이크로서비스 아키텍처 및 개발 결정은 비즈니스 역량에 따라 이루어질 수 있습니다. 자세한 내용은 백서의 AWS에서 컨테이너화된 마이크로서비스 실행의 비즈니스 역량 중심의 구성화 섹션을 참조하십시오.

비즈니스 연속성 계획(BCP)

대규모 마이그레이션과 같은 중단 이벤트가 운영에 미치는 잠재적 영향을 해결하고 비즈니스가 신속하게 운영을 재개할 수 있도록 지원하는 계획입니다.

C

CAF

클라우드 채택 프레임워크를 참조하십시오AWS .

카나리아 배포

최종 사용자에게 버전을 느리고 점진적으로 릴리스하는 것입니다. 확신이 들면 새 버전을 배포하고 현재 버전을 완전히 교체합니다.

CCoE

클라우드 센터 오브 엑설런스를 참조하십시오.

CDC

변경 데이터 캡처를 참조하십시오.

변경 데이터 캡처(CDC)

데이터베이스 테이블과 같은 데이터 소스의 변경 내용을 추적하고 변경 사항에 대한 메타데이터를 기록하는 프로세스입니다. 대상 시스템의 변경 내용을 감사하거나 복제하여 동기화를 유지하는 등의 다양한 용도로 CDC를 사용할 수 있습니다.

카오스 엔지니어링

시스템의 복원력을 테스트하기 위해 의도적으로 장애나 장애를 일으키는 이벤트를 발생시키는 행위 AWS Fault Injection Service (AWS FIS) 를 사용하여 AWS 워크로드에 스트레스를 주는 실험을 수행하고 응답을 평가할 수 있습니다.

CI/CD

지속적 통합 및 지속적 전달을 참조하십시오.

분류

예측을 생성하는 데 도움이 되는 분류 프로세스입니다. 분류 문제에 대한 ML 모델은 이산 값을 예측합니다. 이산 값은 항상 서로 다릅니다. 예를 들어, 모델이 이미지에 자동차가 있는지 여부를 평가해야 할 수 있습니다.

클라이언트측 암호화

대상이 데이터를 AWS 서비스 수신하기 전에 데이터를 로컬로 암호화합니다.

클라우드 혁신 센터(CCoE)

클라우드 모범 사례 개발, 리소스 동원, 마이그레이션 타임라인 설정, 대규모 혁신을 통한 조직 선도 등 조직 전체에서 클라우드 채택 노력을 추진하는 다분야 팀입니다. 자세한 내용은 AWS 클라우드 기업 전략 블로그의 CCoE 게시물을 참조하십시오.

클라우드 컴퓨팅

원격 데이터 스토리지와 IoT 디바이스 관리에 일반적으로 사용되는 클라우드 기술 클라우드 컴퓨팅은 일반적으로 엣지 컴퓨팅 기술과 연결됩니다.

클라우드 운영 모델

IT 조직에서 하나 이상의 클라우드 환경을 구축, 성숙화 및 최적화하는 데 사용되는 운영 모델입니다. 자세한 내용은 클라우드 운영 모델 구축을 참조하십시오.

클라우드 채택 단계

조직이 마이그레이션할 때 일반적으로 거치는 4단계는 다음과 같습니다. AWS 클라우드

- 프로젝트 - 개념 증명 및 학습 목적으로 몇 가지 클라우드 관련 프로젝트 실행
- 기반 – 클라우드 채택 확장을 위한 기초 투자(예: 랜딩 존 생성, CCoE 정의, 운영 모델 구축)
- 마이그레이션 - 개별 애플리케이션 마이그레이션
- Re-invention - 제품 및 서비스 최적화와 클라우드 혁신

Stephen Orban은 기업 전략 블로그의 클라우드 우선주의를 향한 여정 및 채택 단계에 대한 블로그 게시물에서 이러한 단계를 정의했습니다. AWS 클라우드 이들이 AWS 마이그레이션 전략과 어떤 관련이 있는지에 대한 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오.

CMDB

구성 관리 데이터베이스를 참조하십시오.

코드 리포지토리

소스 코드와 설명서, 샘플, 스크립트 등의 기타 자산이 버전 관리 프로세스를 통해 저장되고 업데이트되는 위치입니다. 일반 클라우드 리포지토리에는 또는 이 포함됩니다 GitHub . AWS CodeCommit코드의 각 버전을 브랜치라고 합니다. 마이크로서비스 구조에서 각 리포지토리는 단일 기능 전용입니다. 단일 CI/CD 파이프라인은 여러 리포지토리를 사용할 수 있습니다.

콜드 캐시

비어 있거나, 제대로 채워지지 않았거나, 오래되었거나 관련 없는 데이터를 포함하는 버퍼 캐시입니다. 주 메모리나 디스크에서 데이터베이스 인스턴스를 읽어야 하기 때문에 성능에 영향을 미치며, 이는 버퍼 캐시에서 읽는 것보다 느립니다.

콜드 데이터

거의 액세스되지 않고 일반적으로 과거 데이터인 데이터. 이런 종류의 데이터를 쿼리할 때는 일반적으로 느린 쿼리가 허용됩니다. 이 데이터를 성능이 낮고 비용이 저렴한 스토리지 계층 또는 클래스로 옮기면 비용을 절감할 수 있습니다.

컴퓨터 비전 (CV)

기계 학습을 사용하여 디지털 이미지 및 비디오와 같은 시각적 형식에서 정보를 분석하고 추출하는 AI 분야. 예를 들어 AWS Panorama 는 온프레미스 카메라 네트워크에 CV를 추가하는 디바이스를 제공하고, SageMaker Amazon은 CV용 이미지 처리 알고리즘을 제공합니다.

구성 드리프트

워크로드의 경우 구성이 예상 상태에서 변경됩니다. 이로 인해 워크로드가 규정을 준수하지 않게 될 수 있으며, 일반적으로 점진적이고 의도하지 않은 방식으로 진행됩니다.

구성 관리 데이터베이스(CMDB)

하드웨어 및 소프트웨어 구성 요소와 해당 구성을 포함하여 데이터베이스와 해당 IT 환경에 대한 정보를 저장하고 관리하는 리포지토리입니다. 일반적으로 마이그레이션의 포트폴리오 검색 및 분석 단계에서 CMDB의 데이터를 사용합니다.

규정 준수 팩

AWS Config 규정 준수 및 보안 검사를 사용자 지정하기 위해 조합할 수 있는 규칙 및 수정 조치 모음입니다. YAML 템플릿을 사용하여 한 AWS 계정 및 지역 또는 조직 전체에 단일 엔티티로 적합성 팩을 배포할 수 있습니다. 자세한 내용은 설명서의 적합성 팩을 참조하십시오. AWS Config

지속적 통합 및 지속적 전달(CI/CD)

소프트웨어 릴리스 프로세스의 소스, 빌드, 테스트, 스테이징 및 프로덕션 단계를 자동화하는 프로세스입니다. CI/CD는 일반적으로 파이프라인으로 설명됩니다. CI/CD를 통해 프로세스를 자동화하고, 생산성을 높이고, 코드 품질을 개선하고, 더 빠르게 제공할 수 있습니다. 자세한 내용은 지속적 전달의 이점을 참조하십시오. CD는 지속적 배포를 의미하기도 합니다. 자세한 내용은 지속적 전달(Continuous Delivery)과 지속적인 개발을 참조하십시오.

CV

컴퓨터 비전을 참조하십시오.

D

저장 데이터

스토리지에 있는 데이터와 같이 네트워크에 고정되어 있는 데이터입니다.

데이터 분류

중요도와 민감도를 기준으로 네트워크의 데이터를 식별하고 분류하는 프로세스입니다. 이 프로세스는 데이터에 대한 적절한 보호 및 보존 제어를 결정하는 데 도움이 되므로 사이버 보안 위험 관리 전략의 중요한 구성 요소입니다. 데이터 분류는 AWS Well-Architected 프레임워크의 보안 핵심 요소입니다. 자세한 내용은 데이터 분류를 참조하십시오.

데이터 드리프트

프로덕션 데이터와 ML 모델 학습에 사용된 데이터 간의 상당한 차이 또는 시간 경과에 따른 입력 데이터의 의미 있는 변화. 데이터 드리프트는 ML 모델 예측의 전반적인 품질, 정확성 및 공정성을 저하시킬 수 있습니다.

전송 중 데이터

네트워크를 통과하고 있는 데이터입니다. 네트워크 리소스 사이를 이동 중인 데이터를 예로 들 수 있습니다.

데이터 메시

중앙 집중식 관리 및 거버넌스와 함께 분산되고 분산된 데이터 소유권을 제공하는 아키텍처 프레임워크입니다.

데이터 최소화

꼭 필요한 데이터만 수집하고 처리하는 원칙입니다. 에서 데이터 최소화를 실천하면 개인 정보 보호 위험, 비용 및 분석에 따른 탄소 발자국을 줄일 AWS 클라우드 수 있습니다.

데이터 경계

신뢰할 수 있는 ID만 예상 네트워크에서 신뢰할 수 있는 리소스에 액세스하도록 하는 데 도움이 되는 AWS 환경 내 일련의 예방 가드레일입니다. 자세한 내용은 데이터 경계 구축을 참조하십시오. AWS

데이터 사전 처리

원시 데이터를 ML 모델이 쉽게 구문 분석할 수 있는 형식으로 변환하는 것입니다. 데이터를 사전 처리한다는 것은 특정 열이나 행을 제거하고 누락된 값, 일관성이 없는 값 또는 중복 값을 처리함을 의미할 수 있습니다.

데이터 출처

라이프사이클 전반에 걸쳐 데이터의 출처와 기록을 추적하는 프로세스(예: 데이터 생성, 전송, 저장 방법).

데이터 주체

데이터를 수집 및 처리하는 개인입니다.

데이터 웨어하우스

분석과 같은 비즈니스 인텔리전스를 지원하는 데이터 관리 시스템. 데이터 웨어하우스에는 일반적으로 대량의 과거 데이터가 포함되며 일반적으로 쿼리 및 분석에 사용됩니다.

데이터 정의 언어(DDL)

데이터베이스에서 테이블 및 객체의 구조를 만들거나 수정하기 위한 명령문 또는 명령입니다.

데이터베이스 조작 언어(DML)

데이터베이스에서 정보를 수정(삽입, 업데이트 및 삭제)하기 위한 명령문 또는 명령입니다.

DDL

데이터베이스 정의 언어를 참조하십시오.

딥 앙상블

예측을 위해 여러 딥 러닝 모델을 결합하는 것입니다. 딥 앙상블을 사용하여 더 정확한 예측을 얻거나 예측의 불확실성을 추정할 수 있습니다.

딥 러닝

여러 계층의 인공 신경망을 사용하여 입력 데이터와 관심 대상 변수 간의 매핑을 식별하는 ML 하위 분야입니다.

defense-in-depth

네트워크와 그 안의 데이터 기밀성, 무결성 및 가용성을 보호하기 위해 컴퓨터 네트워크 전체에 일련의 보안 메커니즘과 제어를 신중하게 계층화하는 정보 보안 접근 방식입니다. 이 전략을 채택하면 AWS Organizations 구조의 여러 계층에 AWS여러 컨트롤을 추가하여 리소스를 보호하는 데 도움이 됩니다. 예를 들어 다단계 인증, 네트워크 세분화, 암호화를 결합한 defense-in-depth 접근 방식을 사용할 수 있습니다.

위임된 관리자

에서 AWS Organizations호환 가능한 서비스는 AWS 구성원 계정을 등록하여 조직의 계정을 관리하고 해당 서비스에 대한 권한을 관리할 수 있습니다. 이러한 계정을 해당 서비스의 위임된 관리자라고 합니다. 자세한 내용과 호환되는 서비스 목록은 AWS Organizations 설명서의 AWS Organizations와 함께 사용할 수 있는 AWS 서비스를 참조하십시오.

배포

대상 환경에서 애플리케이션, 새 기능 또는 코드 수정 사항을 사용할 수 있도록 하는 프로세스입니다. 배포에는 코드 베이스의 변경 사항을 구현한 다음 애플리케이션 환경에서 해당 코드베이스를 구축하고 실행하는 작업이 포함됩니다.

개발 환경

환경을 참조하십시오.

탐지 제어

이벤트 발생 후 탐지, 기록 및 알림을 수행하도록 설계된 보안 제어입니다. 이러한 제어는 기존의 예방적 제어를 우회한 보안 이벤트를 알리는 2차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Detective controls를 참조하십시오.

개발 가치 흐름 매핑 (DVSM)

소프트웨어 개발 라이프사이클에서 속도와 품질에 부정적인 영향을 미치는 제약 조건을 식별하고 우선 순위를 지정하는 데 사용되는 프로세스입니다. DVSM은 원래 린 제조 방식을 위해 설계된 가치 흐름 매핑 프로세스를 확장합니다. 소프트웨어 개발 프로세스를 통해 가치를 창출하고 이동하는 데 필요한 단계와 팀에 중점을 둡니다.

디지털 트윈

건물, 공장, 산업 장비 또는 생산 라인과 같은 실제 시스템을 가상으로 표현한 것입니다. 디지털 트윈은 예측 유지 보수, 원격 모니터링, 생산 최적화를 지원합니다.

치수 표

스타 스키마에서 팩트 테이블의 양적 데이터에 대한 데이터 속성을 포함하는 작은 테이블입니다. 차원 테이블 속성은 일반적으로 텍스트처럼 동작하는 텍스트 필드 또는 불연속형 숫자입니다. 이러한 속성은 일반적으로 쿼리 제한, 필터링 및 결과 집합 레이블 지정에 사용됩니다.

재해

워크로드 또는 시스템이 기본 배포 위치에서 비즈니스 목표를 달성하지 못하게 방해하는 이벤트입니다. 이러한 이벤트는 자연재해, 기술적 오류, 의도하지 않은 구성 오류 또는 멀웨어 공격과 같은 사람의 행동으로 인한 결과일 수 있습니다.

재해 복구(DR)

재해로 인한 다운타임과 데이터 손실을 최소화하기 위해 사용하는 전략과 프로세스입니다. 자세한 내용은 워크로드의 재해 복구 AWS: AWS Well-Architected 프레임워크에서의 클라우드 복구를 참조하십시오.

DML

데이터베이스 조작 언어를 참조하십시오.

도메인 기반 설계

구성 요소를 각 구성 요소가 제공하는 진화하는 도메인 또는 핵심 비즈니스 목표에 연결하여 복잡한 소프트웨어 시스템을 개발하는 접근 방식입니다. 이 개념은 에릭 에반스에 의해 그의 저서인 도메인 기반 디자인: 소프트웨어 중심의 복잡성 해결(Boston: Addison-Wesley Professional, 2003)에서 소개되었습니다. Strangler Fig 패턴과 함께 도메인 기반 설계를 사용하는 방법에 대한 자세한 내용은 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

DR

재해 복구를 참조하십시오.

드리프트 감지

기준 구성으로부터의 편차 추적. 예를 들어 시스템 리소스의 편차를 감지하는 AWS CloudFormation 데 사용하거나 거버넌스 요구 사항 준수에 영향을 미칠 수 있는 착륙 지대의 변경 사항을 탐지하는 AWS Control Tower 데 사용할 수 있습니다.

DVSM

개발 가치 흐름 매핑을 참조하십시오.

E

EDA

탐색적 데이터 분석을 참조하십시오.

엣지 컴퓨팅

IoT 네트워크의 엣지에서 스마트 디바이스의 컴퓨팅 성능을 개선하는 기술 클라우드 컴퓨팅과 비교할 때 엣지 컴퓨팅은 통신 대기 시간을 줄이고 응답 시간을 개선할 수 있습니다.

암호화

사람이 읽을 수 있는 일반 텍스트 데이터를 암호문으로 변환하는 컴퓨팅 프로세스입니다.

암호화 키

암호화 알고리즘에 의해 생성되는 무작위 비트의 암호화 문자열입니다. 키의 길이는 다양할 수 있으며 각 키는 예측할 수 없고 고유하게 설계되었습니다.

엔디안

컴퓨터 메모리에 바이트가 저장되는 순서입니다. 빅 엔디안 시스템은 가장 중요한 바이트를 먼저 저장합니다. 리틀 엔디안 시스템은 가장 덜 중요한 바이트를 먼저 저장합니다.

엔드포인트

서비스 엔드포인트를 참조하십시오.

엔드포인트 서비스

Virtual Private Cloud(VPC)에서 호스팅하여 다른 사용자와 공유할 수 있는 서비스입니다. 다른 주체 AWS 계정 또는 AWS Identity and Access Management (IAM) 보안 주체에 권한을 부여하여 엔드포인트 서비스를 생성하고 권한을 부여할 수 있습니다. AWS PrivateLink 이러한 계정 또는 보안 주체는 인터페이스 VPC 엔드포인트를 생성하여 엔드포인트 서비스에 비공개로 연결할 수 있습니

다. 자세한 내용은 Amazon Virtual Private Cloud(VPC) 설명서의 엔드포인트 서비스 생성을 참조하십시오.

ERP (전사적 자원 관리)

기업의 주요 비즈니스 프로세스 (예: 회계, MES, 프로젝트 관리) 를 자동화하고 관리하는 시스템입니다.

봉투 암호화

암호화 키를 다른 암호화 키로 암호화하는 프로세스입니다. 자세한 내용은 AWS Key Management Service (AWS KMS) 설명서의 봉투 암호화를 참조하십시오.

환경

실행 중인 애플리케이션의 인스턴스입니다. 다음은 클라우드 컴퓨팅의 일반적인 환경 유형입니다.

- 개발 환경 - 애플리케이션 유지 관리를 담당하는 핵심 팀만 사용할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. 개발 환경은 변경 사항을 상위 환경으로 승격하기 전에 테스트하는 데 사용됩니다. 이러한 유형의 환경을 테스트 환경이라고도 합니다.
- 하위 환경 - 초기 빌드 및 테스트에 사용되는 환경을 비롯한 애플리케이션의 모든 개발 환경입니다.
- 프로덕션 환경 - 최종 사용자가 액세스할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. CI/CD 파이프라인에서 프로덕션 환경이 마지막 배포 환경입니다.
- 상위 환경 - 핵심 개발 팀 이외의 사용자가 액세스할 수 있는 모든 환경입니다. 프로덕션 환경, 프로덕션 이전 환경 및 사용자 수용 테스트를 위한 환경이 여기에 포함될 수 있습니다.

에픽

애자일 방법론에서 작업을 구성하고 우선순위를 정하는 데 도움이 되는 기능적 범주입니다. 에픽은 요구 사항 및 구현 작업에 대한 개괄적인 설명을 제공합니다. 예를 들어 AWS CAF 보안 에픽에는 ID 및 액세스 관리, 탐지 제어, 인프라 보안, 데이터 보호, 사고 대응 등이 포함됩니다. AWS 마이그레이션 전략의 에픽에 대한 자세한 내용은 프로그램 구현 가이드를 참조하십시오.

ERP

엔터프라이즈 리소스 계획을 참조하십시오.

탐색 데이터 분석(EDA)

데이터 세트를 분석하여 주요 특성을 파악하는 프로세스입니다. 데이터를 수집 또는 집계한 다음 초기 조사를 수행하여 패턴을 찾고, 이상을 탐지하고, 가정을 확인합니다. EDA는 요약 통계를 계산하고 데이터 시각화를 생성하여 수행됩니다.

F

팩트 테이블

[스타 스키마의](#) 중앙 테이블. 비즈니스 운영에 대한 정량적 데이터를 저장합니다. 일반적으로 팩트 테이블에는 측정값이 포함된 열과 차원 테이블의 외부 키가 포함된 열 등 두 가지 유형의 열이 포함됩니다.

빨리 실패하세요

빈번하고 점진적인 테스트를 통해 개발 라이프사이클을 단축하는 철학. 이는 애자일 접근 방식의 중요한 부분입니다.

장애 격리 경계

장애 영향을 제한하고 워크로드의 복원력을 개선하는 데 도움이 되는 가용 영역 AWS 리전, 컨트롤 플레인 또는 데이터 플레인과 같은 경계 AWS 클라우드자세한 내용은 [AWS 장애 격리](#) 경계를 참조하십시오.

기능 브랜치

[브랜치를](#) 참조하십시오.

기능

예측에 사용하는 입력 데이터입니다. 예를 들어, 제조 환경에서 기능은 제조 라인에서 주기적으로 캡처되는 이미지일 수 있습니다.

기능 중요도

모델의 예측에 특성이 얼마나 중요한지를 나타냅니다. 이는 일반적으로 SHAP(Shapley Additive Descriptions) 및 통합 그레디언트와 같은 다양한 기법을 통해 계산할 수 있는 수치 점수로 표현됩니다. 자세한 내용은 [다음을AWS사용한 기계 학습 모델 해석 가능성을](#) 참조하십시오.

기능 변환

추가 소스로 데이터를 보강하거나, 값을 조정하거나, 단일 데이터 필드에서 여러 정보 세트를 추출하는 등 ML 프로세스를 위해 데이터를 최적화하는 것입니다. 이를 통해 ML 모델이 데이터를 활용할 수 있습니다. 예를 들어, 날짜 '2021-05-27 00:15:37'을 '2021년', '5월', '목', '15일'로 분류하면 학습 알고리즘이 다양한 데이터 구성 요소와 관련된 미묘한 패턴을 학습하는 데 도움이 됩니다.

FGAC

[세분화된 액세스 제어를](#) 참조하십시오.

세분화된 액세스 제어(FGAC)

여러 조건을 사용하여 액세스 요청을 허용하거나 거부합니다.

플래시컷 마이그레이션

단계별 접근 방식 대신 변경 데이터 캡처를 통한 지속적인 데이터 복제를 통해 최단 시간에 데이터를 마이그레이션하는 데이터베이스 마이그레이션 방법입니다. 목표는 가동 중지 시간을 최소화하는 것입니다.

G

지리적 차단

지리적 제한을 참조하십시오.

지리적 제한(지리적 차단)

CloudFrontAmazon에서는 특정 국가의 사용자가 콘텐츠 배포에 액세스하지 못하도록 하는 옵션을 제공합니다. 허용 목록 또는 차단 목록을 사용하여 승인된 국가와 차단된 국가를 지정할 수 있습니다. 자세한 내용은 설명서의 콘텐츠의 지리적 배포 제한을 참조하십시오. CloudFront

Gitflow 워크플로

하위 환경과 상위 환경이 소스 코드 리포지토리의 서로 다른 브랜치를 사용하는 방식입니다. Gitflow 워크플로는 레거시로 간주되며 트렁크 기반 워크플로는 현대적이고 선호되는 접근 방식입니다.

브라운필드 전략

새로운 환경에서 기존 인프라의 부재 시스템 아키텍처에 대한 그린필드 전략을 채택할 때 브라운필드라고도 하는 기존 인프라와의 호환성 제한 없이 모든 새로운 기술을 선택할 수 있습니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

가드레일

조직 단위(OU) 전체에서 리소스, 정책 및 규정 준수를 관리하는 데 도움이 되는 중요 규칙입니다. 예방 가드레일은 규정 준수 표준에 부합하도록 정책을 시행하며, 서비스 제어 정책과 IAM 권한 경계를 사용하여 구현됩니다. 탐지 가드레일은 정책 위반 및 규정 준수 문제를 감지하고 해결을 위한 알림을 생성하며, 이들은, Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, Amazon Inspector 및 사용자 지정 AWS Lambda 검사를 사용하여 구현됩니다.

H

하

고가용성을 확인하세요.

이기종 데이터베이스 마이그레이션

다른 데이터베이스 엔진을 사용하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Oracle에서 Amazon Aurora로) 이기종 마이그레이션은 일반적으로 리아키텍트 작업의 일부이며 스키마를 변환하는 것은 복잡한 작업일 수 있습니다.AWS 는 스키마 변환에 도움이 되는 AWS SCT를 제공합니다.

높은 가용성(HA)

문제나 재해 발생 시 개입 없이 지속적으로 운영할 수 있는 워크로드의 능력. HA 시스템은 자동으로 장애 조치되고, 지속적으로 고품질 성능을 제공하고, 성능에 미치는 영향을 최소화하면서 다양한 부하와 장애를 처리하도록 설계되었습니다.

히스토리언 현대화

제조 산업의 요구 사항을 더 잘 충족하도록 운영 기술(OT) 시스템을 현대화하고 업그레이드하는 데 사용되는 접근 방식입니다. 히스토리언은 공장의 다양한 출처에서 데이터를 수집하고 저장하는 데 사용되는 일종의 데이터베이스입니다.

동종 데이터베이스 마이그레이션

동일한 데이터베이스 엔진을 공유하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Microsoft SQL Server에서 Amazon RDS for SQL Server로) 동종 마이그레이션은 일반적으로 리호스팅 또는 리플랫포밍 작업의 일부입니다. 네이티브 데이터베이스 유틸리티를 사용하여 스키마를 마이그레이션할 수 있습니다.

핫 데이터

자주 액세스하는 데이터(예: 실시간 데이터 또는 최근 번역 데이터). 일반적으로 이 데이터에는 빠른 쿼리 응답을 제공하기 위한 고성능 스토리지 계층 또는 클래스가 필요합니다.

핫픽스

프로덕션 환경의 중요한 문제를 해결하기 위한 긴급 수정입니다. 긴급성 때문에 핫픽스는 일반적으로 일반적인 DevOps 릴리스 워크플로 외부에서 만들어집니다.

하이퍼케어 기간

전환 직후 마이그레이션 팀이 문제를 해결하기 위해 클라우드에서 마이그레이션된 애플리케이션을 관리하고 모니터링하는 기간입니다. 일반적으로 이 기간은 1~4일입니다. 하이퍼케어 기간이 끝나면 마이그레이션 팀은 일반적으로 애플리케이션에 대한 책임을 클라우드 운영 팀에 넘깁니다.

I

IaC

인프라를 코드로 보세요.

자격 증명 기반 정책

환경 내에서 권한을 정의하는 하나 이상의 IAM 보안 주체에 연결된 정책입니다. AWS 클라우드

유휴 애플리케이션

90일 동안 평균 CPU 및 메모리 사용량이 5~20%인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하거나 온프레미스에 유지하는 것이 일반적입니다.

IIoT

산업용 사물 인터넷을 참조하십시오.

불변의 인프라

기존 인프라를 업데이트, 패치 또는 수정하는 대신 프로덕션 워크로드용 새 인프라를 배포하는 모델입니다. 변경 불가능한 인프라는 기본적으로 변경 가능한 인프라보다 더 일관되고 안정적이며 예측 가능합니다. 자세한 내용은 Well-Architected AWS 프레임워크의 변경 불가능한 인프라를 사용한 배포 모범 사례를 참조하십시오.

인바운드(수신) VPC

AWS 다중 계정 아키텍처에서 VPC는 애플리케이션 외부에서 네트워크 연결을 허용, 검사 및 라우팅합니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

증분 마이그레이션

한 번에 전체 전환을 수행하는 대신 애플리케이션을 조금씩 마이그레이션하는 전환 전략입니다. 예를 들어, 처음에는 소수의 마이크로서비스나 사용자만 새 시스템으로 이동할 수 있습니다. 모든 것

이 제대로 작동하는지 확인한 후에는 레거시 시스템을 폐기할 수 있을 때까지 추가 마이크로서비스 또는 사용자를 점진적으로 이동할 수 있습니다. 이 전략을 사용하면 대규모 마이그레이션과 관련된 위험을 줄일 수 있습니다.

Industry 4.0

Klaus Schwab이 연결성, 실시간 데이터, 자동화, 분석 및 AI/ML의 발전을 통한 제조 프로세스의 현대화를 지칭하기 위해 2016년 도입한 용어입니다.

인프라

애플리케이션의 환경 내에 포함된 모든 리소스와 자산입니다.

코드형 인프라(IaC)

구성 파일 세트를 통해 애플리케이션의 인프라를 프로비저닝하고 관리하는 프로세스입니다. IaC는 새로운 환경의 반복 가능성, 신뢰성 및 일관성을 위해 인프라 관리를 중앙 집중화하고, 리소스를 표준화하고, 빠르게 확장할 수 있도록 설계되었습니다.

산업용 사물 인터넷(IIoT)

제조, 에너지, 자동차, 의료, 생명과학, 농업 등의 산업 부문에서 인터넷에 연결된 센서 및 디바이스의 사용 자세한 내용은 산업용 사물 인터넷(IoT) 디지털 트랜스포메이션 전략 구축을 참조하십시오.

검사 VPC

AWS 다중 계정 아키텍처에서 VPC (동일하거나 AWS 리전다른), 인터넷 및 온프레미스 네트워크 간의 네트워크 트래픽 검사를 관리하는 중앙 집중식 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

사물 인터넷(IoT)

인터넷이나 로컬 통신 네트워크를 통해 다른 디바이스 및 시스템과 통신하는 센서 또는 프로세서가 내장된 연결된 물리적 객체의 네트워크 자세한 내용은 IoT란?을 참조하십시오.

해석력

모델의 예측이 입력에 따라 어떻게 달라지는지를 사람이 이해할 수 있는 정도를 설명하는 기계 학습 모델의 특성입니다. 자세한 내용은 Machine learning model interpretability with AWS를 참조하십시오.

IoT

사물 인터넷을 참조하십시오.

IT 정보 라이브러리(TIL)

IT 서비스를 제공하고 이러한 서비스를 비즈니스 요구 사항에 맞게 조정하기 위한 일련의 모범 사례 ITIL은 ITSM의 기반을 제공합니다.

IT 서비스 관리(TSM)

조직의 IT 서비스 설계, 구현, 관리 및 지원과 관련된 활동 클라우드 운영을 ITSM 도구와 통합하는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

ITIL

IT 정보 라이브러리를 참조하십시오.

ITSM

IT 서비스 관리를 참조하십시오.

L

레이블 기반 액세스 제어(LBAC)

사용자 및 데이터 자체에 각각 보안 레이블 값을 명시적으로 할당하는 필수 액세스 제어(MAC)를 구현한 것입니다. 사용자 보안 레이블과 데이터 보안 레이블 간의 교차 부분에 따라 사용자가 볼 수 있는 행과 열이 결정됩니다.

랜딩 존

Landing Zone은 확장 가능하고 안전한 잘 설계된 다중 계정 AWS 환경입니다. 조직은 여기에서부터 보안 및 인프라 환경에 대한 확신을 가지고 워크로드와 애플리케이션을 신속하게 시작하고 배포할 수 있습니다. 랜딩 존에 대한 자세한 내용은 안전하고 확장 가능한 다중 계정 AWS 환경 설정을 참조하십시오.

대규모 마이그레이션

300대 이상의 서버 마이그레이션입니다.

LBAC

레이블 기반 액세스 제어를 참조하십시오.

최소 권한

작업을 수행하는 데 필요한 최소 권한을 부여하는 보안 모범 사례입니다. 자세한 내용은 IAM 설명서의 최소 권한 적용을 참조하십시오.

리프트 앤드 시프트

7 R을 참조하십시오.

리틀 엔디안 시스템

가장 덜 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안 참조.

하위 환경

환경 참조.

M

기계 학습(ML)

패턴 인식 및 학습에 알고리즘과 기법을 사용하는 인공 지능의 한 유형입니다. ML은 사물 인터넷(IoT) 데이터와 같은 기록된 데이터를 분석하고 학습하여 패턴을 기반으로 통계 모델을 생성합니다. 자세한 내용은 기계 학습을 참조하십시오.

기본 브랜치

브랜치 참조.

악성 코드

컴퓨터 보안 또는 개인 정보를 침해하도록 설계된 소프트웨어 멀웨어는 컴퓨터 시스템을 방해하거나, 민감한 정보를 유출하거나, 무단 액세스를 얻을 수 있습니다. 멀웨어의 예로는 바이러스, 웜, 랜섬웨어, 트로이 목마, 스파이웨어, 키로거 등이 있습니다.

매니지드 서비스

AWS 서비스 인프라 계층, 운영 체제 및 플랫폼을 AWS 운영하며 사용자는 엔드포인트에 액세스하여 데이터를 저장하고 검색합니다. 관리형 서비스의 예로는 아마존 심플 스토리지 서비스 (Amazon S3) 와 아마존 DynamoDB가 있습니다. 이러한 서비스를 추상화된 서비스라고도 합니다.

제조 실행 시스템 (MES)

제조 현장에서 원자재를 완제품으로 전환하는 생산 프로세스를 추적, 모니터링, 문서화 및 제어하기 위한 소프트웨어 시스템입니다.

MAP

Migration Acceleration 프로그램을 참조하십시오.

기구

도구를 만들고 도구 채택을 유도한 다음 결과를 검토하여 조정하는 전체 프로세스입니다. 메커니즘은 작동하면서 자체적으로 강화되고 개선되는 사이클입니다. 자세한 내용은 AWS Well-Architected 프레임워크에서의 메커니즘 구축을 참조하십시오.

멤버 계정

조직의 일부인 관리 계정을 AWS 계정 제외한 모든 계정 AWS Organizations하나의 계정은 한 번에 하나의 조직 멤버만 될 수 있습니다.

MES

제조 실행 시스템을 참조하십시오.

메시지 큐 텔레메트리 전송 (MQTT)

퍼블리시/구독 패턴을 기반으로 하는 리소스가 제한된 IoT 디바이스를 위한 경량 machine-to-machine (M2M) 통신 프로토콜입니다.

마이크로서비스

잘 정의된 API를 통해 통신하고 일반적으로 소규모 자체 팀이 소유하는 소규모 독립 서비스입니다. 예를 들어, 보험 시스템에는 영업, 마케팅 등의 비즈니스 역량이나 구매, 청구, 분석 등의 하위 영역에 매핑되는 마이크로 서비스가 포함될 수 있습니다. 마이크로서비스의 이점으로 민첩성, 유연한 확장, 손쉬운 배포, 재사용 가능한 코드, 복원력 등이 있습니다. 자세한 내용은 서버리스 서비스를 사용하여 마이크로서비스 통합을 참조하십시오. AWS

마이크로서비스 아키텍처

각 애플리케이션 프로세스를 마이크로서비스로 실행하는 독립 구성 요소를 사용하여 애플리케이션을 구축하는 접근 방식입니다. 이러한 마이크로서비스는 경량 API를 사용하여 잘 정의된 인터페이스를 통해 통신합니다. 애플리케이션의 특정 기능에 대한 수요에 맞게 이 아키텍처의 각 마이크로서비스를 업데이트, 배포 및 조정할 수 있습니다. 자세한 내용은 마이크로서비스 구현을 참조하십시오. AWS

Migration Acceleration Program(MAP)

조직이 클라우드로 전환하기 위한 강력한 운영 기반을 구축하고 초기 마이그레이션 비용을 상쇄할 수 있도록 컨설팅 지원, 교육 및 서비스를 제공하는 AWS 프로그램입니다. MAP에는 레거시 마이그레이션을 체계적인 방식으로 실행하기 위한 마이그레이션 방법론과 일반적인 마이그레이션 시나리오를 자동화하고 가속화하는 도구 세트가 포함되어 있습니다.

대규모 마이그레이션

애플리케이션 포트폴리오의 대다수를 웨이브를 통해 클라우드로 이동하는 프로세스로, 각 웨이브에서 더 많은 애플리케이션이 더 빠른 속도로 이동합니다. 이 단계에서는 이전 단계에서 배운 모범 사례와 교훈을 사용하여 팀, 도구 및 프로세스의 마이그레이션 팩토리를 구현하여 자동화 및 민첩한 제공을 통해 워크로드 마이그레이션을 간소화합니다. 이것은 AWS 마이그레이션 전략의 세 번째 단계입니다.

마이그레이션 팩토리

자동화되고 민첩한 접근 방식을 통해 워크로드 마이그레이션을 간소화하는 다기능 팀입니다. 마이그레이션 팩토리 팀에는 일반적으로 운영, 비즈니스 분석가 및 소유자, 마이그레이션 엔지니어, 개발자 및 스프린트에서 일하는 DevOps 전문가가 포함됩니다. 엔터프라이즈 애플리케이션 포트폴리오의 20~50%는 공장 접근 방식으로 최적화할 수 있는 반복되는 패턴으로 구성되어 있습니다. 자세한 내용은 이 콘텐츠 세트의 클라우드 마이그레이션 팩토리 가이드와 마이그레이션 팩토리에 대한 설명을 참조하십시오.

마이그레이션 메타데이터

마이그레이션을 완료하는 데 필요한 애플리케이션 및 서버에 대한 정보 각 마이그레이션 패턴에는 서로 다른 마이그레이션 메타데이터 세트가 필요합니다. 마이그레이션 메타데이터의 예로는 대상 서브넷, 보안 그룹, 계정 등이 있습니다. AWS

마이그레이션 패턴

사용되는 마이그레이션 전략, 마이그레이션 대상, 마이그레이션 애플리케이션 또는 서비스를 자세히 설명하는 반복 가능한 마이그레이션 작업입니다. 예: 애플리케이션 마이그레이션 서비스를 사용하여 Amazon EC2로 AWS 마이그레이션을 재호스팅합니다.

Migration Portfolio Assessment(MPA)

로 마이그레이션하기 위한 비즈니스 사례를 검증하기 위한 정보를 제공하는 온라인 도구입니다. AWS 클라우드 MPA는 상세한 포트폴리오 평가(서버 적정 규모 조정, 가격 책정, TCO 비교, 마이그레이션 비용 분석)와 마이그레이션 계획(애플리케이션 데이터 분석 및 데이터 수집, 애플리케이션 그룹화, 마이그레이션 우선순위 지정, 웨이브 계획)을 제공합니다. MPA 도구 (로그인 필요) 는 모든 컨설턴트와 APN 파트너 AWS 컨설턴트에게 무료로 제공됩니다.

마이그레이션 준비 상태 평가(MRA)

CAF를 사용하여 조직의 클라우드 준비 상태에 대한 통찰력을 얻고, 강점과 약점을 파악하고, 식별된 격차를 해소하기 위한 실행 계획을 수립하는 프로세스입니다. AWS 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오. MRA는 AWS 마이그레이션 전략의 첫 번째 단계입니다.

마이그레이션 전략

워크로드를 로 마이그레이션하는 데 사용된 접근 방식. AWS 클라우드자세한 내용은 이 용어집의 7R 항목 및 대규모 마이그레이션 가속화를 위한 조직 동원을 참조하십시오.

ML

기계 학습을 참조하십시오.

현대화

비용을 절감하고 효율성을 높이고 혁신을 활용하기 위해 구식(레거시 또는 모놀리식) 애플리케이션과 해당 인프라를 클라우드의 민첩하고 탄력적이고 가용성이 높은 시스템으로 전환하는 것입니다. 자세한 내용은 의 AWS 클라우드애플리케이션 현대화 전략을 참조하십시오.

현대화 준비 상태 평가

조직 애플리케이션의 현대화 준비 상태를 파악하고, 이점, 위험 및 종속성을 식별하고, 조직이 해당 애플리케이션의 향후 상태를 얼마나 잘 지원할 수 있는지를 확인하는 데 도움이 되는 평가입니다. 평가 결과는 대상 아키텍처의 청사진, 현대화 프로세스의 개발 단계와 마일스톤을 자세히 설명하는 로드맵 및 파악된 격차를 해소하기 위한 실행 계획입니다. 자세한 내용은 에서 애플리케이션의 현대화 준비 상태 평가를 참조하십시오. AWS 클라우드

모놀리식 애플리케이션(모놀리식 유형)

긴밀하게 연결된 프로세스를 사용하여 단일 서비스로 실행되는 애플리케이션입니다. 모놀리식 애플리케이션에는 몇 가지 단점이 있습니다. 한 애플리케이션 기능에 대한 수요가 급증하면 전체 아키텍처 규모를 조정해야 합니다. 코드 베이스가 커지면 모놀리식 애플리케이션의 기능을 추가하거나 개선하는 것도 더 복잡해집니다. 이러한 문제를 해결하기 위해 마이크로서비스 아키텍처를 사용할 수 있습니다. 자세한 내용은 마이크로서비스로 모놀리식 유형 분해를 참조하십시오.

MPA

마이그레이션 포트폴리오 평가를 참조하십시오.

MQTT

메시지 큐 원격 분석 전송을 참조하십시오.

멀티클래스 분류

여러 클래스에 대한 예측(2개 이상의 결과 중 하나 예측)을 생성하는 데 도움이 되는 프로세스입니다. 예를 들어, ML 모델이 '이 제품은 책인가요, 자동차인가요, 휴대폰인가요?' 또는 '이 고객이 가장 관심을 갖는 제품 범주는 무엇인가요?'라고 물을 수 있습니다.

변경 가능한 인프라

프로덕션 워크로드를 위해 기존 인프라를 업데이트하고 수정하는 모델입니다. 일관성, 안정성 및 예측 가능성을 개선하기 위해 AWS Well-Architected Framework는 변경 불가능한 인프라를 모범 사례로 사용할 것을 권장합니다.

O

OAC

원본 액세스 제어를 참조하십시오.

좋아요

원본 액세스 ID를 참조하십시오.

OCM

조직 변경 관리를 참조하십시오.

오프라인 마이그레이션

마이그레이션 프로세스 중 소스 워크로드가 중단되는 마이그레이션 방법입니다. 이 방법은 가동 중지 증가를 수반하며 일반적으로 작고 중요하지 않은 워크로드에 사용됩니다.

OI

운영 통합을 참조하십시오.

안녕하세요.

운영 수준 계약을 참조하십시오.

온라인 마이그레이션

소스 워크로드를 오프라인 상태로 전환하지 않고 대상 시스템에 복사하는 마이그레이션 방법입니다. 워크로드에 연결된 애플리케이션은 마이그레이션 중에도 계속 작동할 수 있습니다. 이 방법은 가동 중지 차단 또는 최소화를 수반하며 일반적으로 중요한 프로덕션 워크로드에 사용됩니다.

OPC-UA

오픈 프로세스 커뮤니케이션 - 통합 아키텍처를 참조하십시오.

오픈 프로세스 커뮤니케이션 - 통합 아키텍처 (OPC-UA)

산업 machine-to-machine 자동화를 위한 (M2M) 통신 프로토콜. OPC-UA는 데이터 암호화, 인증 및 권한 부여 체계와 함께 상호 운용성 표준을 제공합니다.

운영 수준 협약(OLA)

서비스 수준에 관한 계약(SLA)을 지원하기 위해 직무 IT 그룹이 서로에게 제공하기로 약속한 내용을 명확히 하는 계약입니다.

운영 준비 검토 (ORR)

인시던트 및 발생 가능한 실패의 범위를 이해, 평가, 예방 또는 줄이는 데 도움이 되는 질문 및 관련 모범 사례로 구성된 체크리스트입니다. 자세한 내용은 Well-Architected AWS 프레임워크의 운영 준비 상태 검토 (ORR) 를 참조하십시오.

운영 기술 (OT)

물리적 환경과 함께 작동하여 산업 운영, 장비 및 인프라를 제어하는 하드웨어 및 소프트웨어 시스템. 제조 분야에서는 OT와 정보 기술 (IT) 시스템의 통합이 인더스트리 4.0 혁신의 핵심 초점입니다.

운영 통합(OI)

클라우드에서 운영을 현대화하는 프로세스로 준비 계획, 자동화 및 통합을 수반합니다. 자세한 내용은 운영 통합 가이드를 참조하십시오.

조직 트레일

이를 통해 AWS CloudTrail 생성되는 트레일은 조직 AWS 계정 내 모든 사용자의 모든 이벤트를 기록합니다. AWS Organizations이 트레일은 조직에 속한 각 AWS 계정 에 생성되고 각 계정의 활동을 추적합니다. 자세한 내용은 CloudTrail 설명서에서 조직을 위한 트레일 만들기를 참조하십시오.

조직 변경 관리(OCM)

사람, 문화 및 리더십 관점에서 중대하고 파괴적인 비즈니스 혁신을 관리하기 위한 프레임워크입니다. OCM은 변화 채택을 가속화하고, 과도기적 문제를 해결하고, 문화 및 조직적 변화를 주도함으로써 조직이 새로운 시스템 및 전략을 준비하고 전환할 수 있도록 지원합니다. 클라우드 채택 프로젝트에 필요한 변화 속도 때문에 AWS 마이그레이션 전략에서는 이 프레임워크를 사용자 가속화라고 합니다. 자세한 내용은 사용 가이드를 참조하십시오.

오리진 액세스 제어(OAC)

CloudFront에서는 Amazon Simple Storage Service (Amazon S3) 콘텐츠의 보안을 위해 액세스를 제한하는 향상된 옵션을 제공합니다. OAC는 모든 S3 버킷 AWS 리전, AWS KMS (SSE-KMS) 를 사용한 서버 측 암호화, S3 버킷에 대한 동적 및 요청을 모두 지원합니다. PUT DELETE

오리진 액세스 ID(OAI)

CloudFront에서는 Amazon S3 콘텐츠 보안을 위해 액세스를 제한하는 옵션입니다. OAI를 사용하면 Amazon S3가 인증할 수 있는 보안 주체를 CloudFront 생성합니다. 인증된 보안 주체는 특정 배

포를 통해서만 S3 버킷의 콘텐츠에 액세스할 수 있습니다. CloudFront 더 세분화되고 향상된 액세스 제어를 제공하는 OAC도 참조하십시오.

또는

운영 준비 상태 검토를 참조하십시오.

아니요

운영 기술을 참조하십시오.

아웃바운드(송신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 내에서 시작되는 네트워크 연결을 처리하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

P

권한 경계

사용자나 역할이 가질 수 있는 최대 권한을 설정하기 위해 IAM 보안 주체에 연결되는 IAM 관리 정책입니다. 자세한 내용은 IAM 설명서의 권한 경계를 참조하십시오.

개인 식별 정보(PII)

직접 보거나 다른 관련 데이터와 함께 짝을 지을 때 개인의 신원을 합리적으로 추론하는 데 사용할 수 있는 정보입니다. PII의 예로는 이름, 주소, 연락처 정보 등이 있습니다.

PII

개인 식별 정보를 참조하십시오.

플레이북

클라우드에서 핵심 운영 기능을 제공하는 등 마이그레이션과 관련된 작업을 캡처하는 일련의 사전 정의된 단계입니다. 플레이북은 스크립트, 자동화된 런북 또는 현대화된 환경을 운영하는 데 필요한 프로세스나 단계 요약의 형태를 취할 수 있습니다.

PLC

프로그래머블 로직 컨트롤러를 참조하십시오.

PLM

제품 라이프사이클 관리를 참조하십시오.

정책

권한을 정의 (ID 기반 정책 참조) 하거나, 액세스 조건을 지정 (리소스 기반 정책 참조) 하거나, 조직 내 모든 계정에 대한 최대 권한을 정의 AWS Organizations (서비스 제어 정책 참조) 할 수 있는 개체입니다.

다국어 지속성

데이터 액세스 패턴 및 기타 요구 사항을 기반으로 독립적으로 마이크로서비스의 데이터 스토리지 기술 선택. 마이크로서비스가 동일한 데이터 스토리지 기술을 사용하는 경우 구현 문제가 발생하거나 성능이 저하될 수 있습니다. 요구 사항에 가장 적합한 데이터 스토어를 사용하면 마이크로서비스를 더 쉽게 구현하고 성능과 확장성을 높일 수 있습니다. 자세한 내용은 마이크로서비스에서 데이터 지속성 활성화를 참조하십시오.

포트폴리오 평가

마이그레이션을 계획하기 위해 애플리케이션 포트폴리오를 검색 및 분석하고 우선순위를 정하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 상태 평가를 참조하십시오.

조건자

일반적으로 조항에 있는 `true` `false` OR를 반환하는 쿼리 조건입니다. `WHERE`

조건부 푸시다운

전송하기 전에 쿼리의 데이터를 필터링하는 데이터베이스 쿼리 최적화 기법입니다. 이렇게 하면 관계형 데이터베이스에서 검색하고 처리해야 하는 데이터의 양이 줄어들고 쿼리 성능이 향상됩니다.

예방적 제어

이벤트 발생을 방지하도록 설계된 보안 제어입니다. 이 제어는 네트워크에 대한 무단 액세스나 원치 않는 변경을 방지하는 데 도움이 되는 1차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Preventative controls를 참조하십시오.

보안 주체

작업을 수행하고 리소스에 액세스할 수 AWS 있는 엔티티 이 엔티티는 일반적으로 IAM 역할의 루트 사용자 또는 사용자입니다. AWS 계정자세한 내용은 IAM 설명서의 역할 용어 및 개념의 보안 주체를 참조하십시오.

개인 정보 보호 중심 설계

전체 엔지니어링 프로세스에서 개인 정보를 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

프라이빗 호스팅 영역

Amazon Route 53에서 하나 이상의 VPC 내 도메인과 하위 도메인에 대한 DNS 쿼리에 응답하는 방법에 대한 정보가 담긴 컨테이너입니다. 자세한 내용은 Route 53 설명서의 프라이빗 호스팅 영역 작업을 참조하십시오.

사전 예방 제어

규정을 준수하지 않는 리소스의 배포를 방지하도록 설계된 보안 제어입니다. 이러한 컨트롤은 리소스를 프로비저닝하기 전에 리소스를 스캔합니다. 리소스가 컨트롤과 호환되지 않으면 프로비저닝되지 않습니다. 자세한 내용은 AWS Control Tower 설명서의 컨트롤 참조 안내서를 참조하고 보안 제어 구현의 사전 제어를 참조하십시오. AWS

제품 라이프사이클 관리 (PLM)

설계, 개발, 출시부터 성장 및 성숙도, 폐기 및 제거에 이르는 전체 라이프사이클에 걸쳐 제품에 대한 데이터 및 프로세스를 관리하는 것입니다.

프로덕션 환경

환경을 참조하십시오.

프로그래머블 로직 컨트롤러 (PLC)

제조 분야에서 기계를 모니터링하고 제조 프로세스를 자동화하는 매우 안정적이고 적응력이 뛰어난 컴퓨터입니다.

가명화

데이터세트의 개인 식별자를 자리 표시자 값으로 바꾸는 프로세스입니다. 가명화는 개인 정보를 보호하는 데 도움이 될 수 있습니다. 가명화된 데이터는 여전히 개인 데이터로 간주됩니다.

게시/구독 (게시/구독)

마이크로서비스 간의 비동기 통신을 통해 확장성과 응답성을 개선할 수 있는 패턴입니다. 예를 들어 마이크로서비스 기반 MES에서 마이크로서비스는 다른 마이크로서비스가 구독할 수 있는 채널에 이벤트 메시지를 게시할 수 있습니다. 시스템은 게시 서비스를 변경하지 않고도 새 마이크로서비스를 추가할 수 있습니다.

Q

쿼리 계획

SQL 관계형 데이터베이스 시스템의 데이터에 액세스하는 데 사용되는 일련의 단계 (예: 지침).

쿼리 계획 회귀

데이터베이스 서비스 최적화 프로그램이 데이터베이스 환경을 변경하기 전보다 덜 최적의 계획을 선택하는 경우입니다. 통계, 제한 사항, 환경 설정, 쿼리 파라미터 바인딩 및 데이터베이스 엔진 업데이트의 변경으로 인해 발생할 수 있습니다.

R

RACI 매트릭스

RACI (책임, 책임, 상담, 정보 제공) 를 참조하십시오.

랜섬웨어

결제가 완료될 때까지 컴퓨터 시스템이나 데이터에 대한 액세스를 차단하도록 설계된 악성 소프트웨어입니다.

RASCI 매트릭스

책임, 책임, 상담, 정보 제공 (RACI) 을 참조하십시오.

RCAC

행 및 열 액세스 제어를 참조하십시오.

읽기 전용 복제본

읽기 전용 용도로 사용되는 데이터베이스의 사본입니다. 쿼리를 읽기 전용 복제본으로 라우팅하여 기본 데이터베이스의 로드를 줄일 수 있습니다.

재설계

7 R을 참조하십시오.

Recovery Point Objective(RPO)

마지막 데이터 복구 시점 이후 허용되는 최대 시간입니다. 이에 따라 마지막 복구 시점과 서비스 중단 사이에 허용되는 데이터 손실로 간주되는 범위가 결정됩니다.

Recovery Time Objective(RTO)

서비스 중단과 서비스 복원 사이의 허용 가능한 지연 시간입니다.

리팩터링

7 R을 참조하십시오.

리전

지리적 AWS 영역별 리소스 모음. AWS 리전 각각은 격리되어 있고 서로 독립적이므로 내결함성, 안정성 및 복원력을 제공합니다. 자세한 내용은 사용할 수 있는 AWS 리전 계정 지정을 참조하십시오.

회귀

숫자 값을 예측하는 ML 기법입니다. 예를 들어, '이 집은 얼마에 팔릴까?'라는 문제를 풀기 위해 ML 모델은 선형 회귀 모델을 사용하여 주택에 대해 알려진 사실(예: 면적)을 기반으로 주택의 매매 가격을 예측할 수 있습니다.

리호스팅

7 R을 참조하십시오.

release

배포 프로세스에서 변경 사항을 프로덕션 환경으로 승격시키는 행위입니다.

고쳐 놓다

7 R을 참조하십시오.

리플랫포밍

7 R을 참조하십시오.

환매

7 R을 참조하십시오.

복원력

장애를 견디거나 장애를 복구할 수 있는 애플리케이션의 능력 고가용성 및 재해 복구는 복원력을 계획할 때 일반적으로 고려해야 할 사항입니다. AWS 클라우드자세한 내용은 AWS 클라우드 복원력을 참조하십시오.

리소스 기반 정책

Amazon S3 버킷, 엔드포인트, 암호화 키 등의 리소스에 연결된 정책입니다. 이 유형의 정책은 액세스가 허용된 보안 주체, 지원되는 작업 및 충족해야 하는 기타 조건을 지정합니다.

RACI(Responsible, Accountable, Consulted, Informed) 매트릭스

마이그레이션 활동 및 클라우드 운영에 참여하는 모든 당사자의 역할과 책임을 정의하는 매트릭스입니다. 매트릭스 이름은 매트릭스에 정의된 책임 유형에서 파생됩니다. 실무 담당자 (R), 의사 결

정권자 (A), 업무 수행 조언자 (C), 결과 통보 대상자 (I). 지원자는 (S) 선택사항입니다. 지원자를 포함하면 매트릭스를 RASCI 매트릭스라고 하고, 지원자를 제외하면 RACI 매트릭스라고 합니다.

대응 제어

보안 기준에서 벗어나거나 부정적인 이벤트를 해결하도록 설계된 보안 제어입니다. 자세한 내용은 Implementing security controls on AWS의 Responsive controls를 참조하십시오.

retain

7 R을 참조하십시오.

은퇴

7 R을 참조하십시오.

회전

공격자가 자격 증명에 액세스하는 것을 더 어렵게 만들기 위해 암호를 주기적으로 업데이트하는 프로세스입니다.

행 및 열 액세스 제어(RCAC)

액세스 규칙이 정의된 기본적이고 유연한 SQL 표현식을 사용합니다. RCAC는 행 권한과 열 마스크로 구성됩니다.

RPO

복구 지점 목표를 참조하십시오.

RTO

복구 시간 목표를 참조하십시오.

런북

특정 작업을 수행하는 데 필요한 일련의 수동 또는 자동 절차입니다. 일반적으로 오류율이 높은 반복 작업이나 절차를 간소화하기 위해 런북을 만듭니다.

S

SAML 2.0

많은 ID 제공업체 (IdPs) 가 사용하는 개방형 표준입니다. 이 기능을 사용하면 페더레이션 싱글 사인온 (SSO) 이 가능하므로 조직의 모든 사용자를 위해 IAM에서 사용자를 생성하지 않고도 사용자가 AWS API 작업에 AWS Management Console 로그인하거나 API 작업을 호출할 수 있습니다.

SAML 2.0 기반 페더레이션에 대한 자세한 내용은 IAM 설명서의 SAML 2.0 기반 페더레이션 정보를 참조하십시오.

SCADA

감독 제어 및 데이터 수집을 참조하십시오.

SCP

서비스 제어 정책을 참조하십시오.

secret

에는 AWS Secrets Manager암호화된 형태로 저장하는 비밀번호나 사용자 자격 증명과 같은 기밀 또는 제한된 정보. 비밀 값과 해당 메타데이터로 구성됩니다. 비밀 값은 바이너리, 단일 문자열 또는 여러 문자열일 수 있습니다. 자세한 내용은 Secrets Manager 시크릿에는 무엇이 들어 있나요? 를 참조하십시오. Secrets Manager 설명서에서 확인할 수 있습니다.

보안 제어

위협 행위자가 보안 취약성을 악용하는 능력을 방지, 탐지 또는 감소시키는 기술적 또는 관리적 가드레일입니다. 보안 제어에는 예방적, 탐정적, 대응적, 사전 예방적 제어의 네 가지 기본 유형이 있습니다.

보안 강화

공격 표면을 줄여 공격에 대한 저항력을 높이는 프로세스입니다. 더 이상 필요하지 않은 리소스 제거, 최소 권한 부여의 보안 모범 사례 구현, 구성 파일의 불필요한 기능 비활성화 등의 작업이 여기에 포함될 수 있습니다.

보안 정보 및 이벤트 관리(SIEM) 시스템

보안 정보 관리(SIM)와 보안 이벤트 관리(SEM) 시스템을 결합하는 도구 및 서비스입니다. SIEM 시스템은 서버, 네트워크, 디바이스 및 기타 소스에서 데이터를 수집, 모니터링 및 분석하여 위협과 보안 침해를 탐지하고 알림을 생성합니다.

보안 대응 자동화

보안 이벤트에 자동으로 대응하거나 보안 이벤트를 해결하도록 설계된 사전 정의되고 프로그래밍된 조치입니다. 이러한 자동화는 보안 모범 사례를 구현하는 데 도움이 되는 탐지 또는 대응형 보안 제어 역할을 합니다. AWS 자동 응답 조치의 예로는 VPC 보안 그룹 수정, Amazon EC2 인스턴스 패치, 자격 증명 교체 등이 있습니다.

서버 측 암호화

수신자에 의한 목적지의 데이터 암호화 AWS 서비스

서비스 제어 정책(SCP)

AWS Organizations에 속한 조직의 모든 계정에 대한 권한을 중앙 집중식으로 제어하는 정책입니다. SCP는 관리자가 사용자 또는 역할에 위임할 수 있는 작업에 대해 제한을 설정하거나 가드레일을 정의합니다. SCP를 허용 목록 또는 거부 목록으로 사용하여 허용하거나 금지할 서비스 또는 작업을 지정할 수 있습니다. 자세한 내용은 AWS Organizations 설명서의 서비스 제어 정책을 참조하십시오.

서비스 엔드포인트

의 진입점 URL입니다 AWS 서비스. 엔드포인트를 사용하여 대상 서비스에 프로그래밍 방식으로 연결할 수 있습니다. 자세한 내용은 AWS 일반 참조의 AWS 서비스 엔드포인트를 참조하십시오.

서비스 수준에 관한 계약(SLA)

IT 팀이 고객에게 제공하기로 약속한 내용(예: 서비스 가동 시간 및 성능)을 명시한 계약입니다.

서비스 수준 표시기 (SLI)

오류율, 가용성 또는 처리량과 같은 서비스의 성능 측면을 측정하는 것입니다.

서비스 수준 목표 (SLO)

서비스 수준 지표로 측정되는 서비스 상태를 나타내는 대상 지표입니다.

공동 책임 모델

클라우드 보안 및 규정 준수에 AWS 대한 책임을 공유하는 것을 설명하는 모델입니다. AWS 클라우드의 보안을 책임지는 반면, 사용자는 클라우드에서의 보안을 담당합니다. 자세한 내용은 공동 책임 모델을 참조하십시오.

시엠

보안 정보 및 이벤트 관리 시스템을 참조하십시오.

단일 장애 지점 (SPOF)

응용 프로그램의 중요한 단일 구성 요소에서 발생한 오류로 인해 시스템이 중단될 수 있습니다.

SLA

SLA (서비스 수준 계약) 를 참조하십시오.

SLI

서비스 수준 표시기 참조.

SLO

서비스 수준 목표를 참조하십시오.

split-and-seed 모델

현대화 프로젝트를 확장하고 가속화하기 위한 패턴입니다. 새로운 기능과 제품 릴리스가 정의되면 핵심 팀이 분할되어 새로운 제품 팀이 만들어집니다. 이를 통해 조직의 역량과 서비스 규모를 조정하고, 개발자 생산성을 개선하고, 신속한 혁신을 지원할 수 있습니다. 자세한 내용은 의 애플리케이션 현대화를 위한 단계별 접근 방식을 참조하십시오. AWS 클라우드

SPOF

단일 장애 지점 보기.

스타 스키마

하나의 큰 팩트 테이블을 사용하여 트랜잭션 또는 측정 데이터를 저장하고 하나 이상의 작은 차원 테이블을 사용하여 데이터 속성을 저장하는 데이터베이스 구성 구조입니다. 이 구조는 데이터 웨어하우스에서 사용하거나 비즈니스 인텔리전스 용도로 설계되었습니다.

Strangler Fig 패턴

레거시 시스템을 폐기할 수 있을 때까지 시스템 기능을 점진적으로 다시 작성하고 교체하여 모놀리식 시스템을 현대화하기 위한 접근 방식. 이 패턴은 무화과 덩굴이 나무로 자라 결국 숙주를 압도하고 대체하는 것과 비슷합니다. Martin Fowler가 모놀리식 시스템을 다시 작성할 때 위험을 관리하는 방법으로 이 패턴을 도입했습니다. 이 패턴을 적용하는 방법의 예는 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

서브넷

VPC의 IP 주소 범위입니다. 서브넷은 단일 가용 영역에 상주해야 합니다.

감독 통제 및 데이터 수집 (SCADA)

제조 시 하드웨어와 소프트웨어를 사용하여 물리적 자산과 생산 작업을 모니터링하는 시스템입니다.

대칭 암호화

동일한 키를 사용하여 데이터를 암호화하고 복호화하는 암호화 알고리즘입니다.

합성 테스트

잠재적 문제를 감지하거나 성능을 모니터링하기 위해 사용자 상호 작용을 시뮬레이션하는 방식으로 시스템을 테스트합니다. Amazon CloudWatch Synthetics를 사용하여 이러한 테스트를 생성할 수 있습니다.

T

tags

리소스 구성을 위한 메타데이터 역할을 하는 키-값 쌍. AWS 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색 및 필터링할 수 있습니다. 자세한 내용은 AWS 리소스에 태그 지정을 참조하십시오.

대상 변수

지도 ML에서 예측하려는 값으로, 결과 변수라고도 합니다. 예를 들어, 제조 설정에서 대상 변수는 제품 결함일 수 있습니다.

작업 목록

런북을 통해 진행 상황을 추적하는 데 사용되는 도구입니다. 작업 목록에는 런북의 개요와 완료해야 할 일반 작업 목록이 포함되어 있습니다. 각 일반 작업에 대한 예상 소요 시간, 소유자 및 진행 상황이 작업 목록에 포함됩니다.

테스트 환경

환경을 참조하십시오.

훈련

ML 모델이 학습할 수 있는 데이터를 제공하는 것입니다. 훈련 데이터에는 정답이 포함되어야 합니다. 학습 알고리즘은 훈련 데이터에서 대상(예측하려는 답)에 입력 데이터 속성을 매핑하는 패턴을 찾고, 이러한 패턴을 캡처하는 ML 모델을 출력합니다. 그런 다음 ML 모델을 사용하여 대상을 모르는 새 데이터에 대한 예측을 할 수 있습니다.

전송 게이트웨이

VPC와 온프레미스 네트워크를 상호 연결하는 데 사용할 수 있는 네트워크 전송 허브입니다. 자세한 내용은 AWS Transit Gateway 설명서의 트랜짓 게이트웨이란 무엇입니까? 를 참조하십시오.

트렁크 기반 워크플로

개발자가 기능 브랜치에서 로컬로 기능을 구축하고 테스트한 다음 해당 변경 사항을 기본 브랜치에 병합하는 접근 방식입니다. 이후 기본 브랜치는 개발, 프로덕션 이전 및 프로덕션 환경에 순차적으로 구축됩니다.

신뢰할 수 있는 액세스

조직 내 AWS Organizations 및 해당 계정에서 사용자를 대신하여 작업을 수행하도록 지정한 서비스에 권한 부여 신뢰할 수 있는 서비스는 필요할 때 각 계정에 서비스 연결 역할을 생성하여 관

리 작업을 수행합니다. 자세한 내용은 AWS Organizations 설명서의 다른 AWS 서비스와 AWS Organizations 함께 사용을 참조하십시오.

튜닝

ML 모델의 정확도를 높이기 위해 훈련 프로세스의 측면을 여러 변경하는 것입니다. 예를 들어, 레이블링 세트를 생성하고 레이블을 추가한 다음 다양한 설정에서 이러한 단계를 여러 번 반복하여 모델을 최적화하는 방식으로 ML 모델을 훈련할 수 있습니다.

피자 두 판 팀

피자 두 판만 들고 배불리 먹을 수 있는 소규모 DevOps 팀. 피자 두 판 팀 규모는 소프트웨어 개발에 있어 가능한 최상의 공동 작업 기회를 보장합니다.

U

불확실성

예측 ML 모델의 신뢰성을 저해할 수 있는 부정확하거나 불완전하거나 알려지지 않은 정보를 나타내는 개념입니다. 불확실성에는 두 가지 유형이 있습니다. 인식론적 불확실성은 제한적이고 불완전한 데이터에 의해 발생하는 반면, 우연한 불확실성은 데이터에 내재된 노이즈와 무작위성에 의해 발생합니다. 자세한 내용은 Quantifying uncertainty in deep learning systems 가이드를 참조하십시오.

차별화되지 않은 작업

애플리케이션을 만들고 운영하는 데 필요하지만 최종 사용자에게 직접적인 가치를 제공하거나 경쟁 우위를 제공하지 못하는 작업을 헤비 리프팅이라고도 합니다. 차별화되지 않은 작업의 예로는 조달, 유지보수, 용량 계획 등이 있습니다.

상위 환경

환경을 보세요.

V

정리

스토리지를 회수하고 성능을 향상시키기 위해 증분 업데이트 후 정리 작업을 수반하는 데이터베이스 유지 관리 작업입니다.

버전 제어

리포지토리의 소스 코드 변경과 같은 변경 사항을 추적하는 프로세스 및 도구입니다.

VPC 피어링

프라이빗 IP 주소를 사용하여 트래픽을 라우팅할 수 있게 하는 두 VPC 간의 연결입니다. 자세한 내용은 Amazon VPC 설명서의 VPC 피어링이란?을 참조하십시오.

취약성

시스템 보안을 손상시키는 소프트웨어 또는 하드웨어 결함입니다.

W

웜 캐시

자주 액세스하는 최신 관련 데이터를 포함하는 버퍼 캐시입니다. 버퍼 캐시에서 데이터베이스 인스턴스를 읽을 수 있기 때문에 주 메모리나 디스크에서 읽는 것보다 빠릅니다.

웜 데이터

자주 액세스하지 않는 데이터입니다. 이런 종류의 데이터를 쿼리할 때는 일반적으로 적절히 느린 쿼리가 허용됩니다.

윈도우 함수

현재 레코드와 어떤 식으로든 관련된 행 그룹에 대해 계산을 수행하는 SQL 함수입니다. 윈도우 함수는 이동 평균을 계산하거나 현재 행의 상대적 위치를 기반으로 행 값에 액세스하는 등의 작업을 처리하는 데 유용합니다.

워크로드

고객 대면 애플리케이션이나 백엔드 프로세스 같이 비즈니스 가치를 창출하는 리소스 및 코드 모음입니다.

워크스트림

마이그레이션 프로젝트에서 특정 작업 세트를 담당하는 직무 그룹입니다. 각 워크스트림은 독립적이지만 프로젝트의 다른 워크스트림을 지원합니다. 예를 들어, 포트폴리오 워크스트림은 애플리케이션 우선순위 지정, 웨이브 계획, 마이그레이션 메타데이터 수집을 담당합니다. 포트폴리오 워크스트림은 이러한 자산을 마이그레이션 워크스트림에 전달하고, 마이그레이션 워크스트림은 서버와 애플리케이션을 마이그레이션합니다.

웜

한 번 쓰고, 많이 읽으세요.

WQF

AWS 워크로드 검증 프레임워크를 참조하십시오.

한 번 작성하고 여러 번 읽기 (WORM)

데이터를 한 번 쓰고 데이터가 삭제되거나 수정되지 않도록 하는 스토리지 모델입니다. 인증된 사용자는 필요한 만큼 데이터를 여러 번 읽을 수 있지만 변경할 수는 없습니다. 이 데이터 스토리지 인프라는 변경할 수 없는 것으로 간주됩니다.

Z

제로데이 익스플로잇

제로데이 취약점을 악용하는 공격 (일반적으로 멀웨어) 입니다.

제로데이 취약성

프로덕션 시스템의 명백한 결함 또는 취약성입니다. 위협 행위자는 이러한 유형의 취약성을 사용하여 시스템을 공격할 수 있습니다. 개발자는 공격의 결과로 취약성을 인지하는 경우가 많습니다.

좀비 애플리케이션

평균 CPU 및 메모리 사용량이 5% 미만인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하는 것이 일반적입니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.