Iceberg 数据湖技术

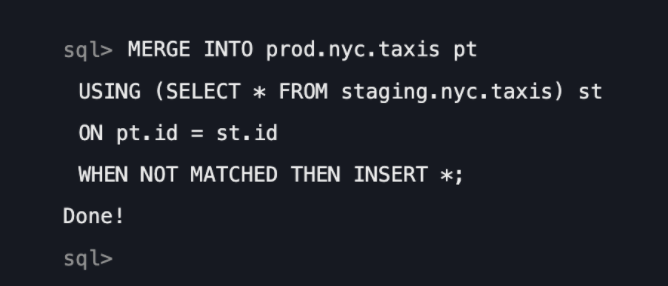
# 简介

Apache Iceberg是用于庞大分析数据集的高性能表格式（ table format ）。它为大数据带来了SQL表的可靠性和简单性，同时使得Spark\Trino\Flink\Presto等引擎可以同时安全的使用相同的表。

# 优势

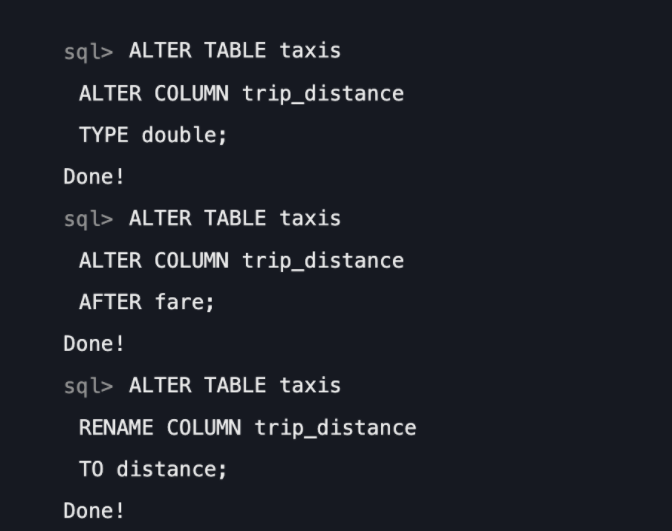
1.灵活的SQL：

Iceberg 支持灵活的 SQL 来合并新数据、更新现有行、执行有针对性的删除。 Iceberg 可以立即重写数据文件以提高读取性能，也可以使用删除增量来加快更新速度。



2.完整的模式演变：

模式演变是高效的。添加一列不会带来“僵尸”数据。列可以重命名且重新排序。最重要的是，架构的变更永远不需要重写您的表。



3.隐藏分区：

对表数据生成分区值的任务是枯燥且易错的，Iceberg将完全自行处理，并跳过不需要的分区和数据。无需为快速查询添加额外的过滤，表布局可以随着数据或查询的变化而更新。

4.时间旅行和回退：

时间旅行支持使用完全相同的表快照的可重现查询，或让用户轻松检查变更效果。版本回滚允许用户通过将表重置为之前的良好的状态来快速纠正问题。

5. 数据压缩：

开箱即用的数据压缩支持，您可以选择不同的重写策略，例如装箱或排序，以优化文件布局和大小。

6. 支持事务（ACID）：

上游数据写入即可见，不影响当前数据处理任务，简化ETL；提供upsert和merge into能力，可以极大地缩小数据入库延迟；

7. 支持多种计算引擎：

优秀的内核抽象使之不绑定特定的计算引擎，目前Iceberg支持的计算引擎有Spark、Flink、Presto以及Hive。

# ****稳定性和性能：****

关于可靠性与性能方面，Apache Iceberg 适用于查询大型表格，其中单个表可以包含数十 PB 的数据，甚至可以在没有分布式 SQL 引擎的情况下读取它们，并且旨在解决最终一致的云对象存储中的正确性问题：

* 快速的扫描计划：不需要分布式 SQL 引擎来读取表或查找文件
* 高级过滤：使用表元数据、分区和列级统计信息修剪数据文件
* 通过避免 listing 和 renames，适用于任何云存储并在 HDFS 中减少 NN 拥塞
* 可序列化隔离：表更改是原子的，读取时不会看到部分或未提交的更改
* 利用乐观锁控制并发写入，并会重试以确保兼容更新成功

# Iceberg数据存储格式：

1.data files（数据文件）:

数据文件是Apache Iceberg表真实存储数据的文件，一般是在表的数据存储目录的data目录下，如果我们的文件格式选择的是parquet,那么文件是以“.parquet”结尾，例如：

00000-0-root\_20211212192602\_8036d31b-9598-4e30-8e67-ce6c39f034da-job\_1639237002345\_0025-00001.parquet 就是一个数据文件。Iceberg每次更新会产生多个数据文件(data files)。

2.Snapshot（表快照）：

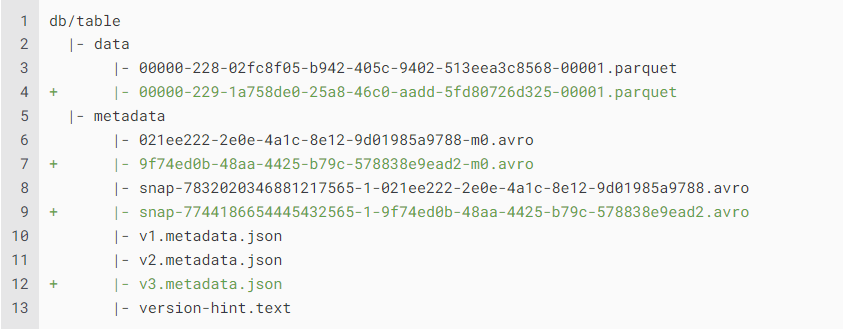
快照代表一张表在某个时刻的状态。每个快照里面会列出表在某个时刻的所有 data files 列表。data files是存储在不同的manifest files里面，manifest files是存储在一个Manifest list文件里面，而一个Manifest list文件代表一个快照。

3.Manifest list（清单列表）

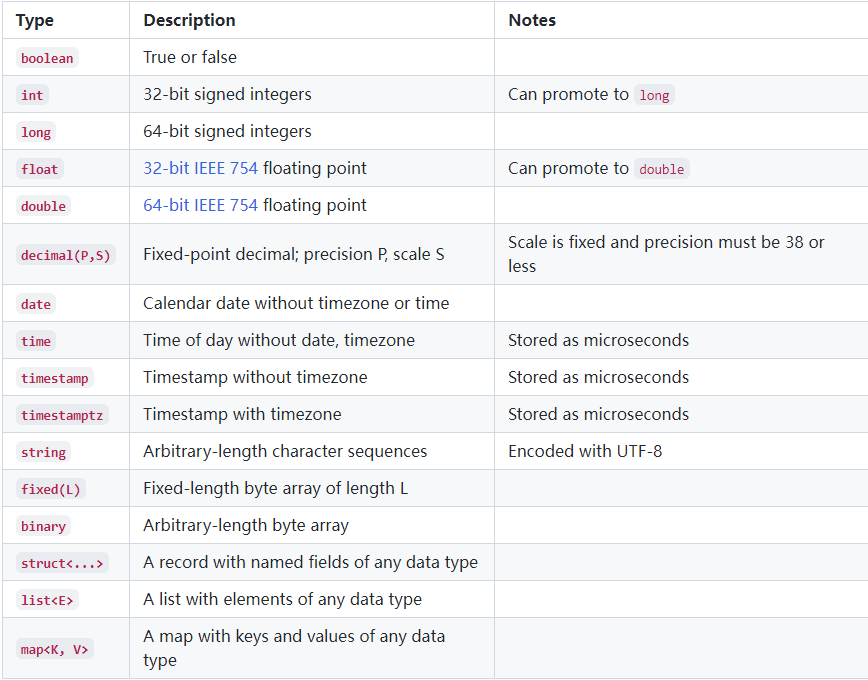
manifest list是一个元数据文件，它列出构建表快照（Snapshot）的清单（Manifest file）。这个元数据文件中存储的是Manifest file列表，每个Manifest file占据一行。每行中存储了Manifest file的路径、其存储的数据文件（data files）的分区范围，增加了几个数文件、删除了几个数据文件等信息，这些信息可以用来在查询时提供过滤，加快速度。

4.Manifest file（清单文件）

Manifest file也是一个元数据文件，它列出组成快照（snapshot）的数据文件（data files）的列表信息。每行都是每个数据文件的详细描述，包括数据文件的状态、文件路径、分区信息、列级别的统计信息（比如每列的最大最小值、空值数等）、文件的大小以及文件里面数据行数等信息。其中列级别的统计信息可以在扫描表数据时过滤掉不必要的文件。



# Iceberg支持的数据类型：



# Flink 整合iceberg：

准备环境：

Hadoop集群hive flink集群

### Flink SQL Client

1. 启动flink standalone集群

# HADOOP\_HOME is your hadoop root directory after unpack the binary package.

export HADOOP\_CLASSPATH=`$HADOOP\_HOME/bin/hadoop classpath`

# Start the flink standalone cluster

./bin/start-cluster.sh

1. 下载flink iceberg runtime的包，启动flink-sql

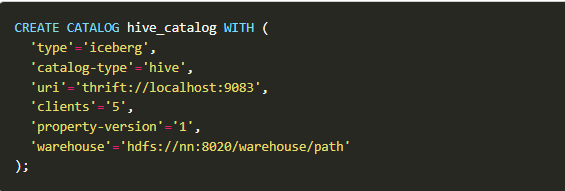
# HADOOP\_HOME is your hadoop root directory after unpack the binary package.

export HADOOP\_CLASSPATH=`$HADOOP\_HOME/bin/hadoop classpath`

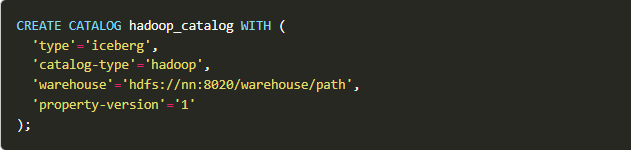
./bin/sql-client.sh embedded -j <flink-runtime-directory>/iceberg-flink-runtime-xxx.jar shell

1. 创建 catalog

Hive catalog



Hadoop catalog



1. DDL 语句

创建 数据库

CREATE DATABASE iceberg\_db;

USE iceberg\_db;

创建 表

CREATE TABLE `hive\_catalog`.`default`.`sample` (

id BIGINT COMMENT 'unique id',

data STRING

);

分区

CREATE TABLE `hive\_catalog`.`default`.`sample` (

id BIGINT COMMENT 'unique id',

data STRING

) PARTITIONED BY (data);

Create table like

CREATE TABLE `hive\_catalog`.`default`.`sample` (

id BIGINT COMMENT 'unique id',

data STRING

);

CREATE TABLE `hive\_catalog`.`default`.`sample\_like` LIKE `hive\_catalog`.`default`.`sample`;

Alter table



Alter table .. rename to



Drop table

DROP TABLE `hive\_catalog`.`default`.`sample`;

1. 查询，删除，插入

略

**批量读**

SET execution.runtime-mode = batch;

SELECT \* FROM sample ;

**Flink streaming read**

-- Submit the flink job in streaming mode for current session.

SET execution.runtime-mode = streaming;

-- Enable this switch because streaming read SQL will provide few job options in flink SQL hint options.

SET table.dynamic-table-options.enabled=true;

-- Read all the records from the iceberg current snapshot, and then read incremental data starting from that snapshot.

SELECT \* FROM sample /\*+ OPTIONS('streaming'='true', 'monitor-interval'='1s')\*/ ;

-- Read all incremental data starting from the snapshot-id '3821550127947089987' (records from this snapshot will be excluded).

SELECT \* FROM sample /\*+ OPTIONS('streaming'='true', 'monitor-interval'='1s', 'start-snapshot-id'='3821550127947089987')\*/ ;