Apache Iceberg Crash Course

The Read and Write Process for Apache Iceberg Tables

Curriculum

July 11: What is a Data Lakehouse and What is a Table Format?

July 16: The Architecture of Apache Iceberg, Apache Hudi and Delta Lake

July 23: The Read and Write Process for Apache Iceberg Tables

Aug 13: Understanding Apache Iceberg's Partitioning Features

Aug 27: Optimizing Apache Iceberg Tables

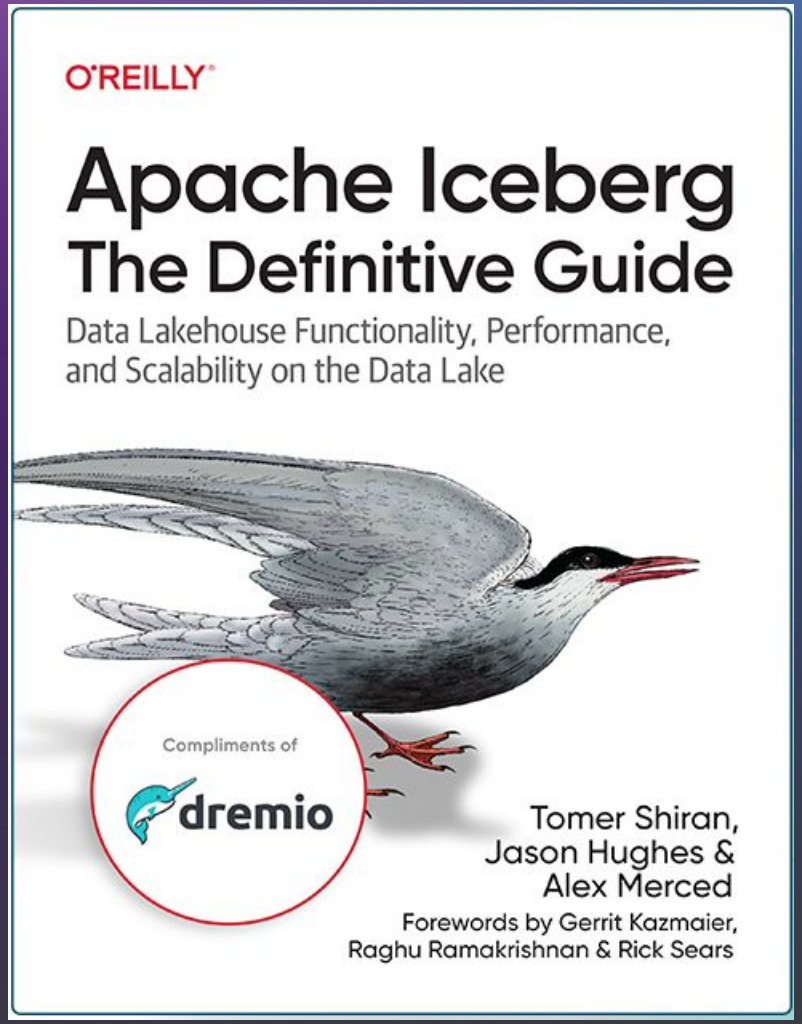
Sep 3: Streaming with Apache Iceberg

Sep 17: The Role of Apache Iceberg Catalogs

Oct 1: Versioning with Apache Iceberg

Oct 15: Ingesting Data into Apache Iceberg with Apache Spark

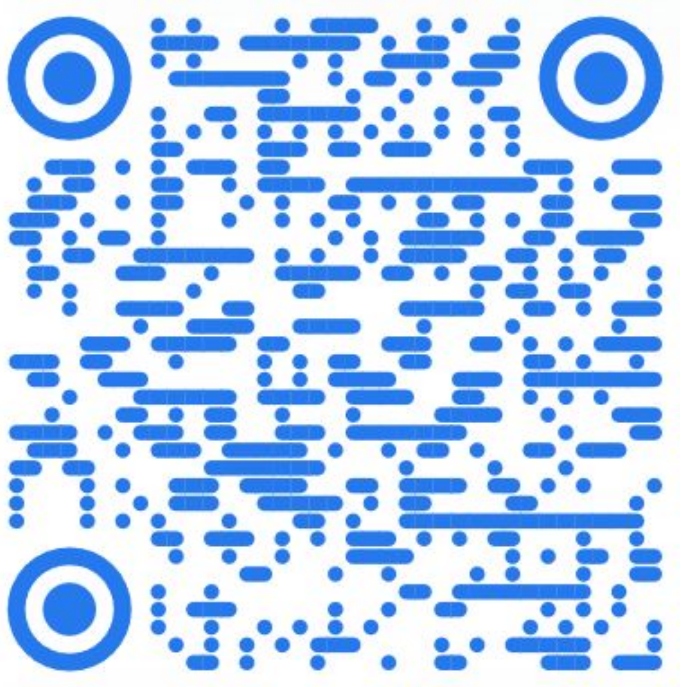
Oct 29: Ingesting Data into Apache Iceberg with Dremio


O'REILLY
Apache Iceberg
The Definitive Guide
Data Lakehouse Functionality, Performance, and Scalability on the Data Lake
Compliments of
dremio
Tomer Shiran, Jason Hughes & Alex Merced
Forewords by Gerrit Kazmaier, Raghu Ramakrishnan & Rick Sears

Podcast

dremio.com/gnarly-data-waves
Youtube | Spotify| iTunes

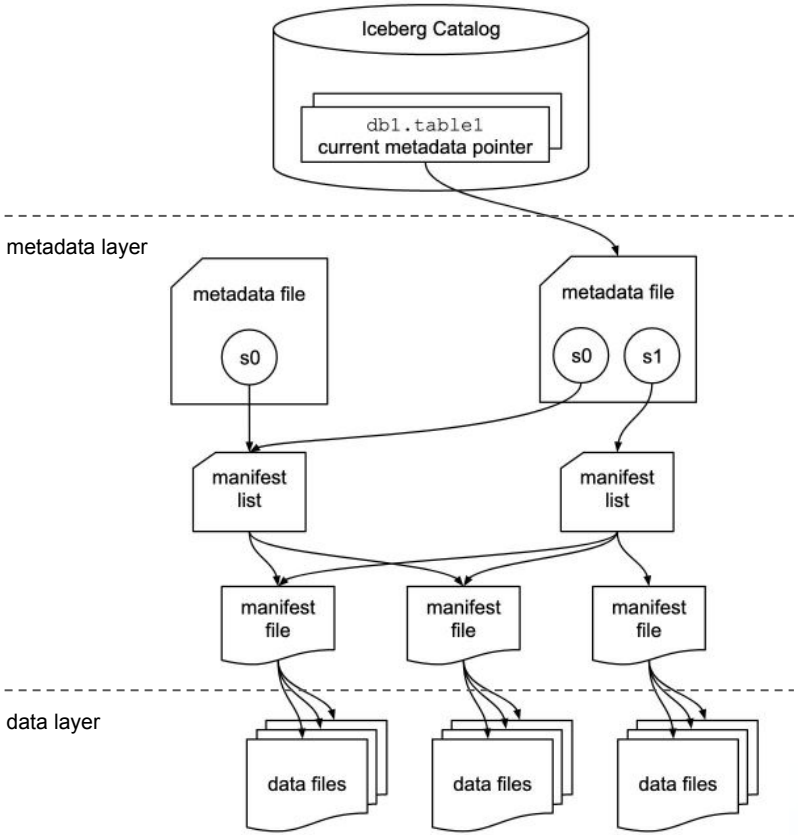
Questions



community.dremio.com
Apache Iceberg Category

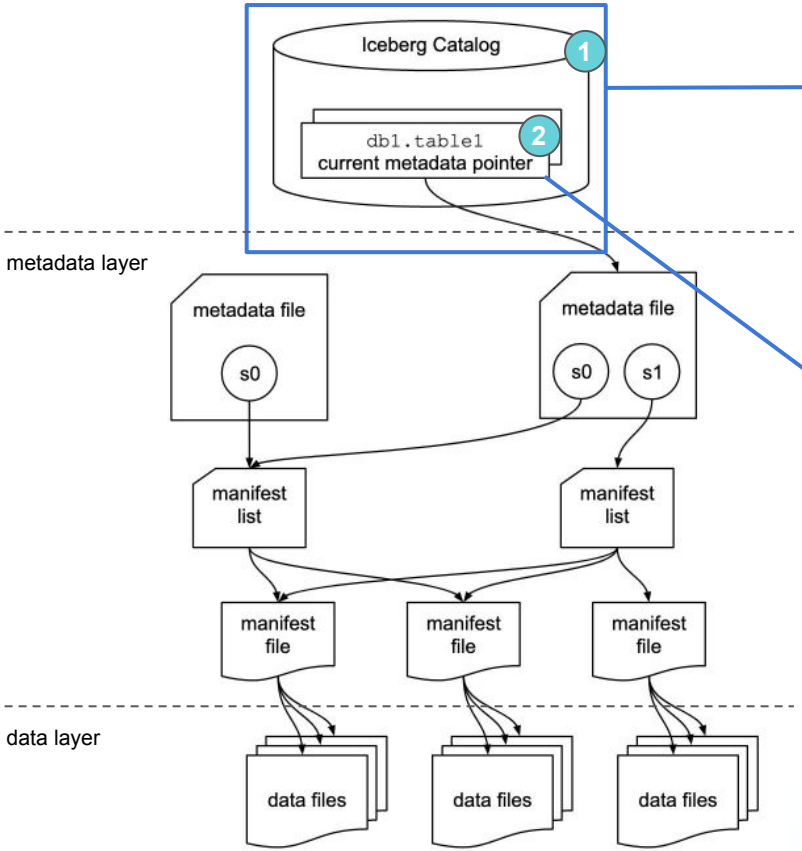
How Iceberg Works

Iceberg table format



- **Overview of the components**
- **Summary of the read path (SELECT)** #

Iceberg components: Catalog



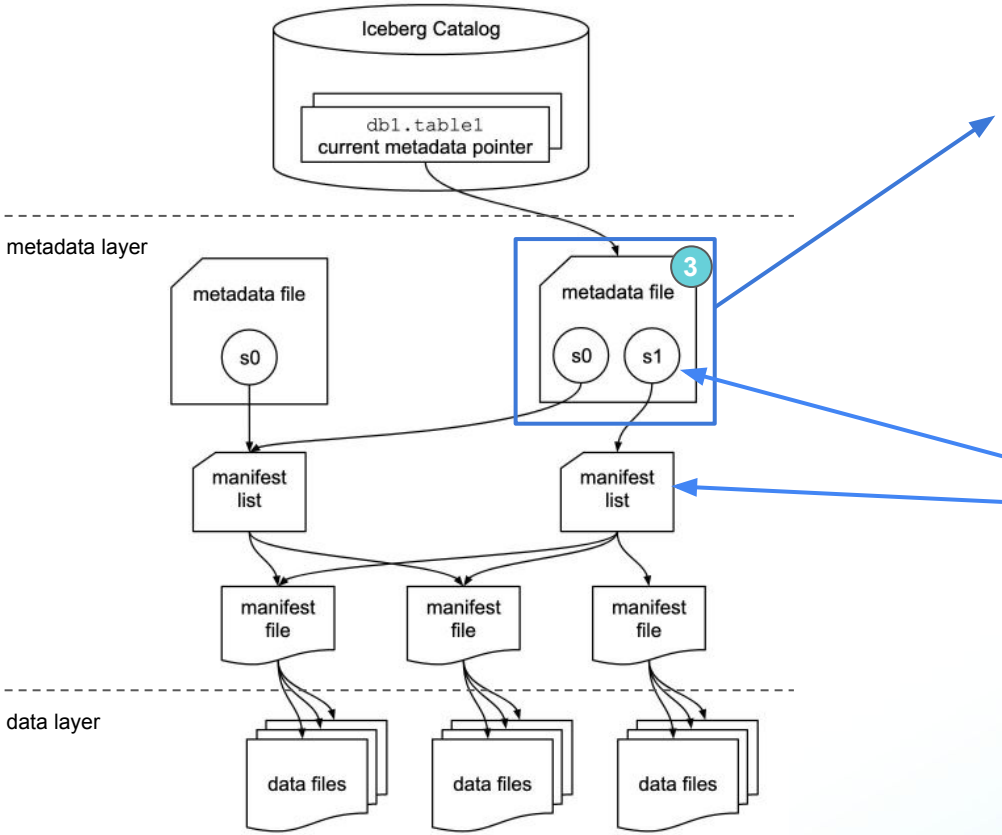
Iceberg Catalog

- A store that houses the current metadata pointer for Iceberg tables
- Must support atomic operations for updating the current metadata pointer (e.g. HDFS, HMS, Nessie)

`table1`'s current metadata pointer

- Mapping of table name to the location of current metadata file

Iceberg components: Metadata File



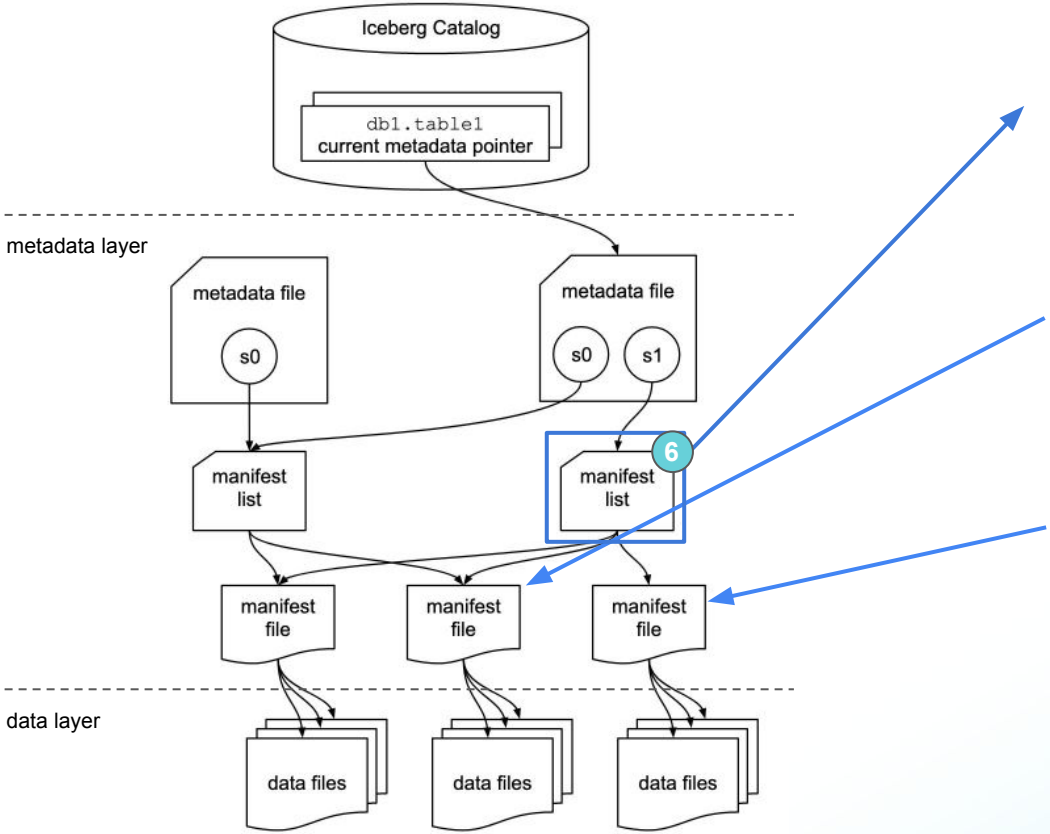
Metadata file - stores metadata about a table at a certain point in time

```
{
  "table-uuid" : "<uuid>",
  "location" : "/path/to/table/dir",
  "schema": {...},
  "partition-spec": [ {<partition-details>}, ...],
  "current-snapshot-id": <snapshot-id>,
  "snapshots": [ {
      "snapshot-id": <snapshot-id>
      "manifest-list": "/path/to/manifest/list.avro"
  }, ...],
  ...
}
```

4

5

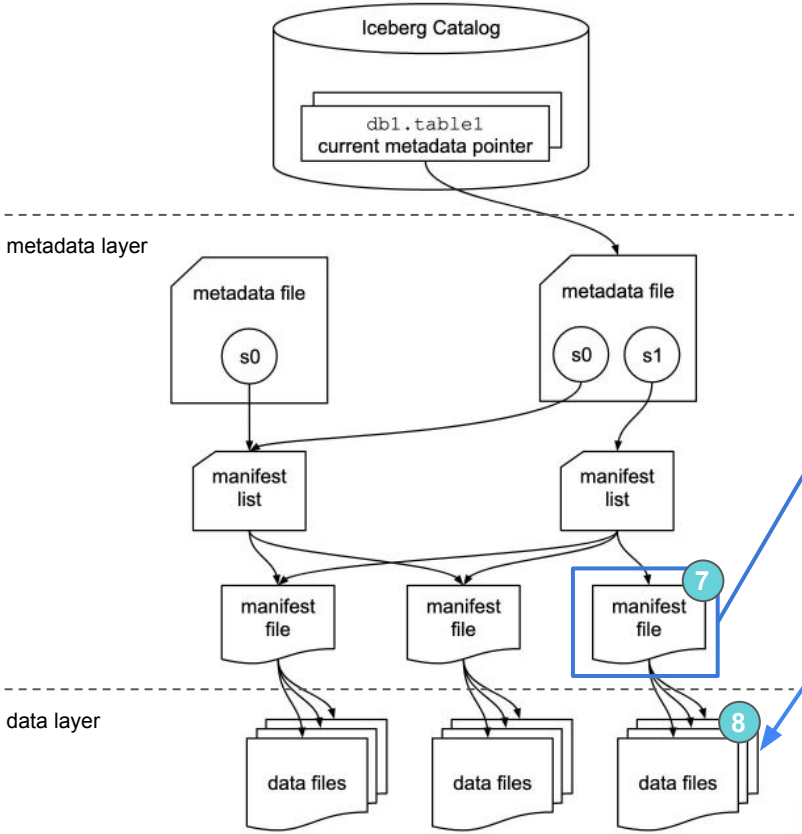
Iceberg components: Manifest List



Manifest list file - a list of manifest files

```
{
  "manifest-path" : "/path/to/manifest/file.avro",
  "added-snapshot-id": <snapshot-id>,
  "partition-spec-id": <partition-spec-id>,
  "partitions": [ {partition-info}, ...],
  ...
}
{
  "manifest-path" : "/path/to/manifest/file2.avro",
  "added-snapshot-id": <snapshot-id>,
  "partition-spec-id": <partition-spec-id>,
  "partitions": [ {partition-info}, ...],
  ...
}
```

Iceberg components: Manifest file



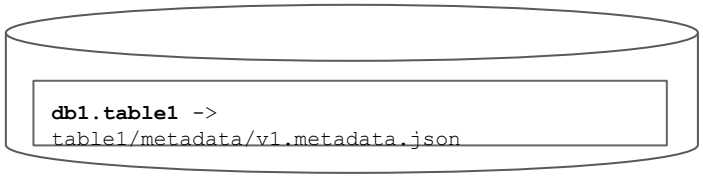
Manifest file - a list of data files, along with details and stats about each data file

```
{
  "data-file": {
    "file-path": "/path/to/data/file.parquet",
    "file-format": "PARQUET",
{"<partition-field>":{"<data-type>":<value>}},
    "record-count": <num-records>,
    "null-value-counts": [{
      "column-index": "1", "value": 4
    }, ...],
    "lower-bounds": [{
      "column-index": "1", "value": "aaa"
    }, ...],
    "upper-bounds": [{
      "column-index": "1", "value": "eee"
    }, ...],
  }
  ...
}
{
 ...
}
```

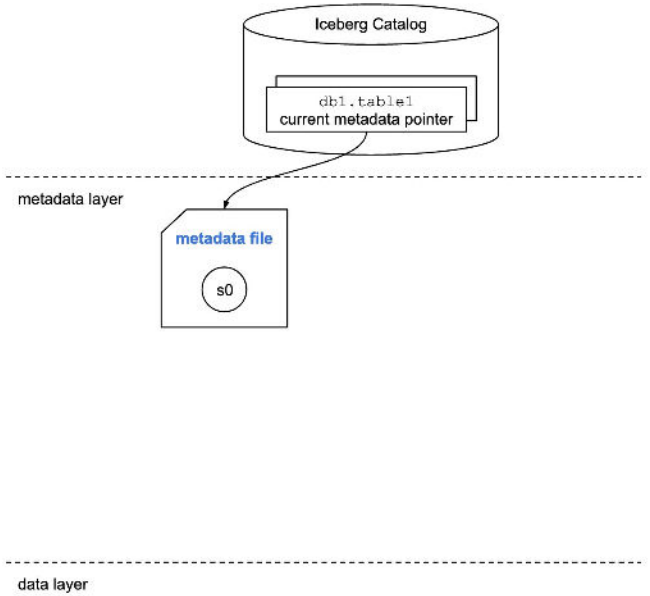
Example Transactions

✅

```
CREATE TABLE db1.table1 (
        order_id bigint,
        customer_id bigint,
        order_amount DECIMAL(10, 2),
        order_ts TIMESTAMP
)
USING iceberg
PARTITIONED BY ( hour(order_ts) );
```

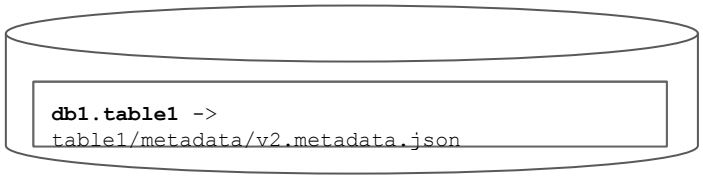



```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |- data/
```

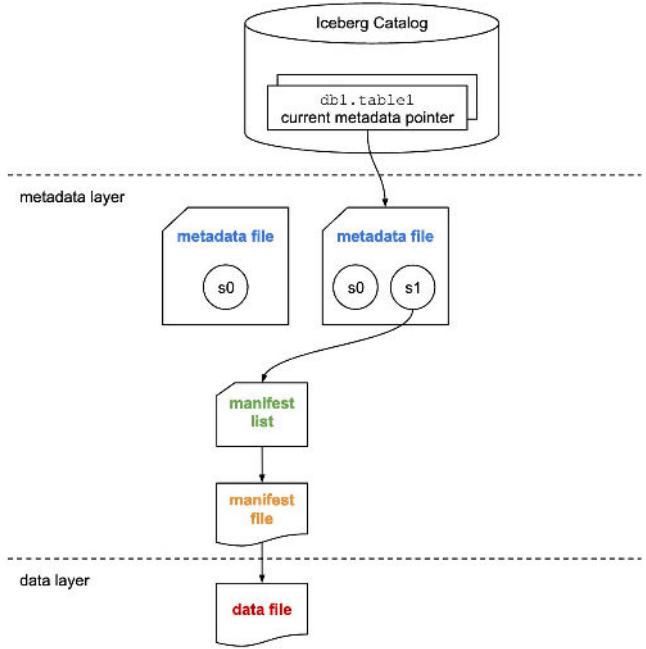


✅

```
INSERT INTO db1.table1 VALUES (
        123,
        456,
        36.17,
        '2021-01-26 08:10:23'
);
```

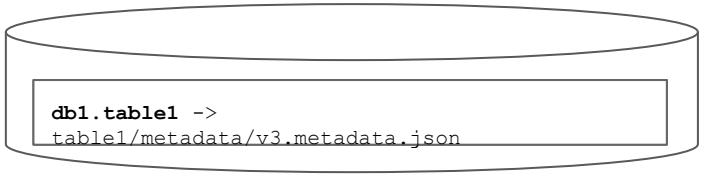



```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- snap-2938-1-4103.avro
 |    |- d8f9-ad19-4e.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
```

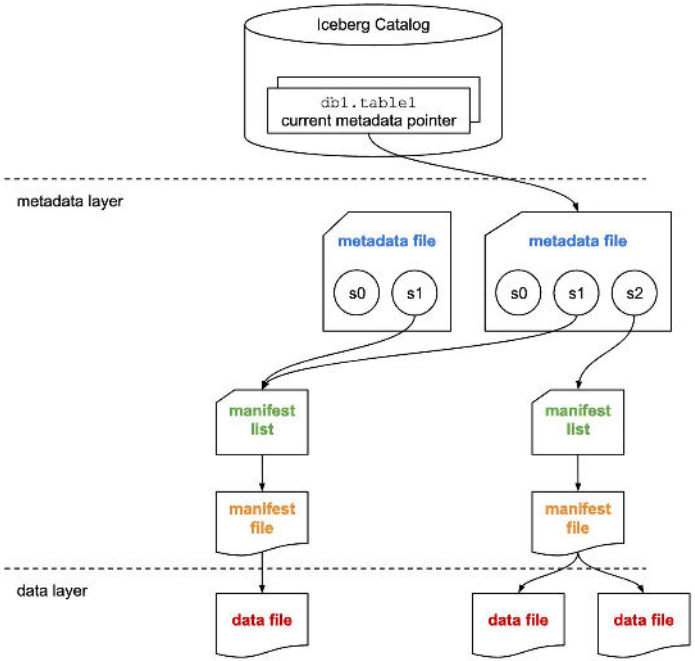


✅

```
MERGE INTO db1.table1
USING ( SELECT * FROM table1_stage ) s
ON table1.order_id = s.order_id
WHEN MATCHED THEN UPDATE table1.order_amount = s.order_amount
WHEN NOT MATCHED THEN INSERT *
```

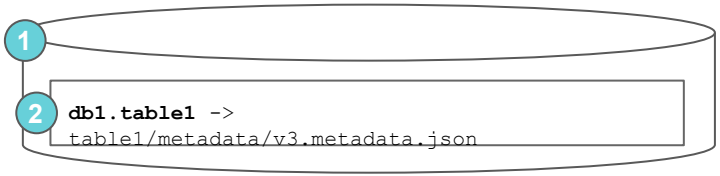



```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- v3.metadata.json
 |    |- snap-29c8-1-b103.avro
 |    |- snap-9fa1-3-16c3.avro
 |    |- d8f9-ad19-4e.avro
 |    |- 0d9a-98fa-77.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
 |    |    |- 00000-1-aef71.parquet
 |    |- order_ts_hour=2021-01-27-10/
 |    |    |- 00000-3-0fa3a.parquet
```

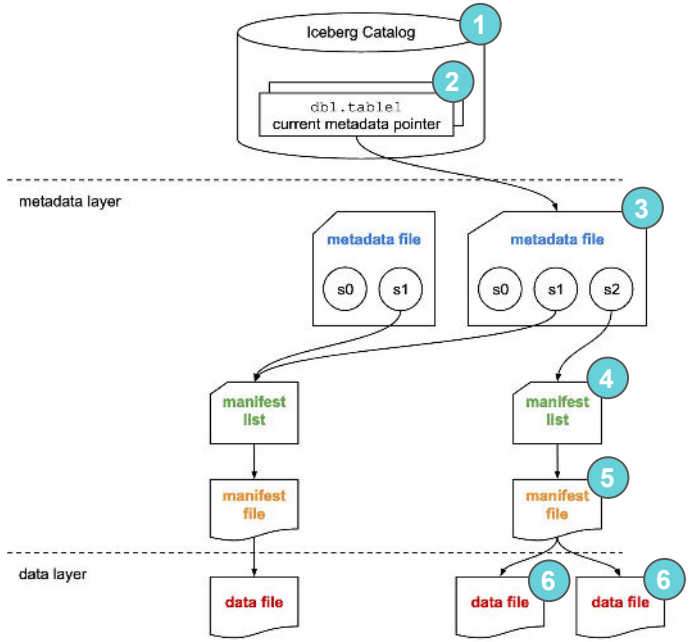


✅ SELECT *

FROM db1.table1



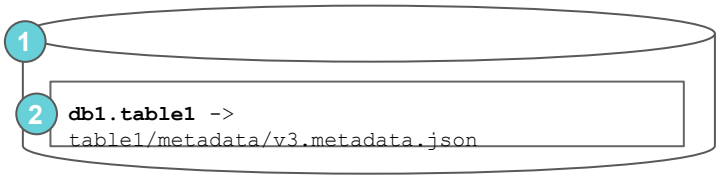

```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- v3.metadata.json
 |    |- snap-29c8-1-b103.avro
 |    |- snap-9fa1-3-16c3.avro
 |    |- d8f9-ad19-4e.avro
 |    |- 0d9a-98fa-77.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
 |    |    |- 00000-1-aef71.parquet
 |    |- order_ts_hour=2021-01-27-10/
 |    |    |- 00000-3-0fa3a.parquet
```



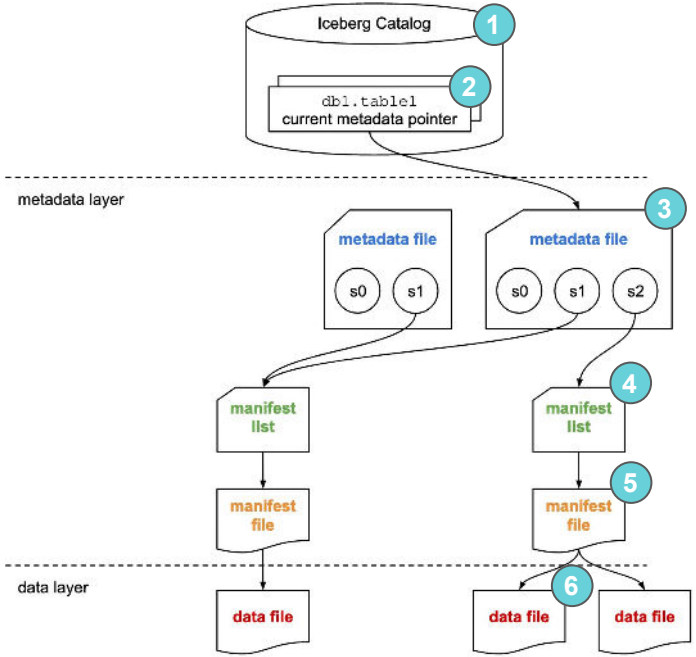
✅
```
SELECT *

FROM db1.table1

WHERE order_ts = DATE '2021-01-26'
```



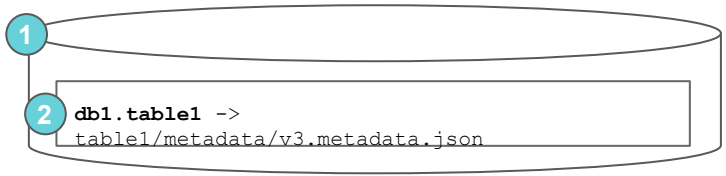

```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- v3.metadata.json
 |    |- snap-29c8-1-b103.avro
 |    |- snap-9fa1-3-16c3.avro
 |    |- d8f9-ad19-4e.avro
 |    |- 0d9a-98fa-77.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
 |    |    |- 00000-1-aef71.parquet
 |    |- order_ts_hour=2021-01-27-10/
 |    |    |- 00000-3-0fa3a.parquet
```



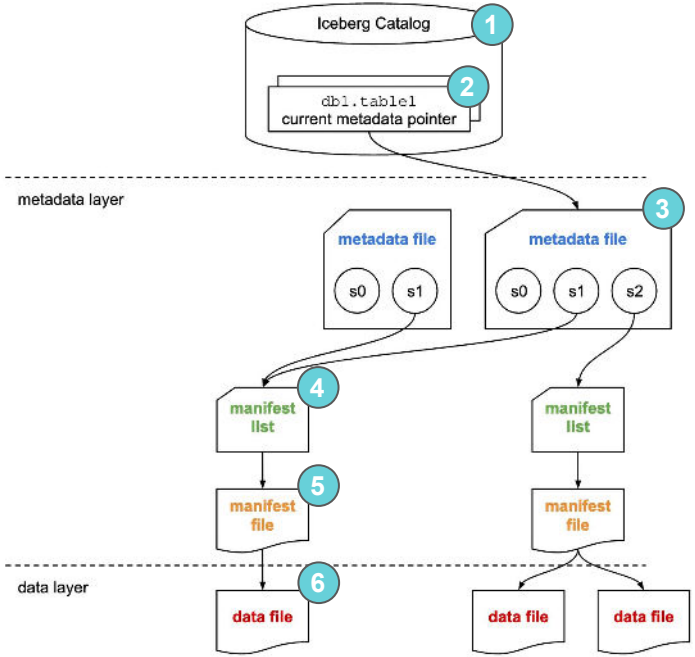
✅ SELECT *

FROM db1.table1 AS OF '2021-05-26 09:30:00'

-- (timestamp is from before MERGE INTO operation)

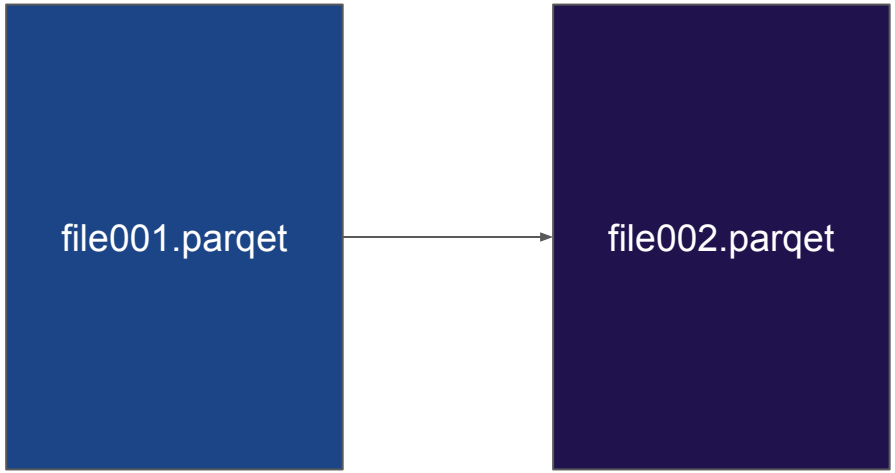



```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- v3.metadata.json
 |    |- snap-29c8-1-b103.avro
 |    |- snap-9fa1-3-16c3.avro
 |    |- d8f9-ad19-4e.avro
 |    |- 0d9a-98fa-77.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
 |    |    |- 00000-1-aef71.parquet
 |    |- order_ts_hour=2021-01-27-10/
 |    |    |- 00000-3-0fa3a.parquet
```

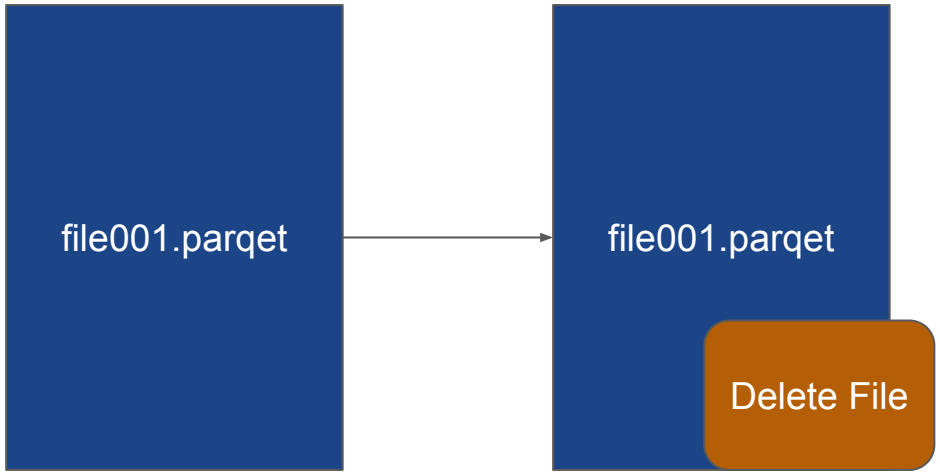


Copy on Write vs Merge on Read

Copy-on-Write | COW
(rewrite data file)



Merge-on-Read | MOR
(write changes to file)



Two Types of Delete Files

1. Position Deletes
    a. Track rows in parquet files to be ignored
    b. Faster to reconcile at read time vs equality deletes
    c. Slower at write time vs equality deletes

2. Equality Deletes
    a. Captures values of records that should be ignored
    b. Slower to reconcile at read time vs position deletes
    c. Faster to write that position deletes

Other Files

Other Files

1. Puffin Files
    a. Files that contain binary blobs
    b. Can be used to store indexes and sketches to accelerate queries

2. Partition Stats Files
    a. Contains statistics on a particular partition

A Iceberg/Dremio Lakehouse on your laptop exercise

Deploy Dremio Software or Dremio Cloud

Hands-On

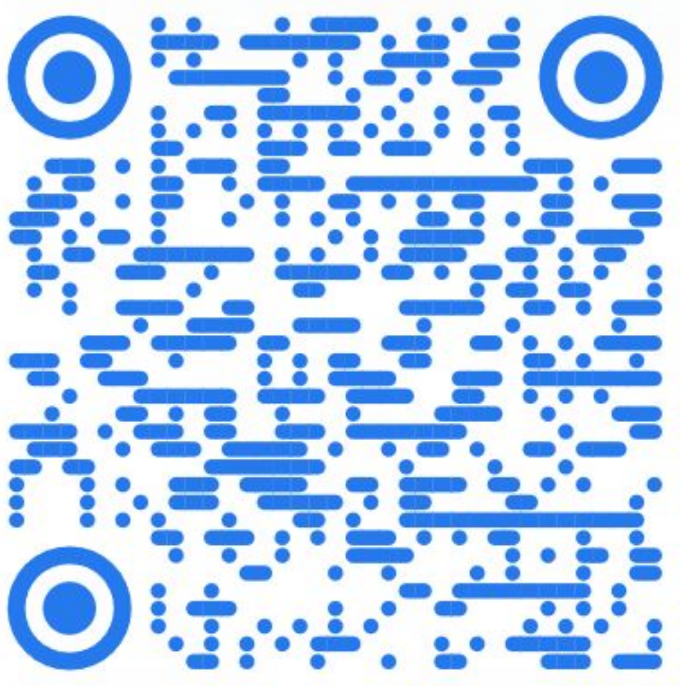
Postgres -> Iceberg -> Dashboard

SQLServer -> Iceberg -> Dashboard

MongoDB -> Iceberg -> Dashboard

dremio.com/blog

Questions



community.dremio.com
Apache Iceberg Category