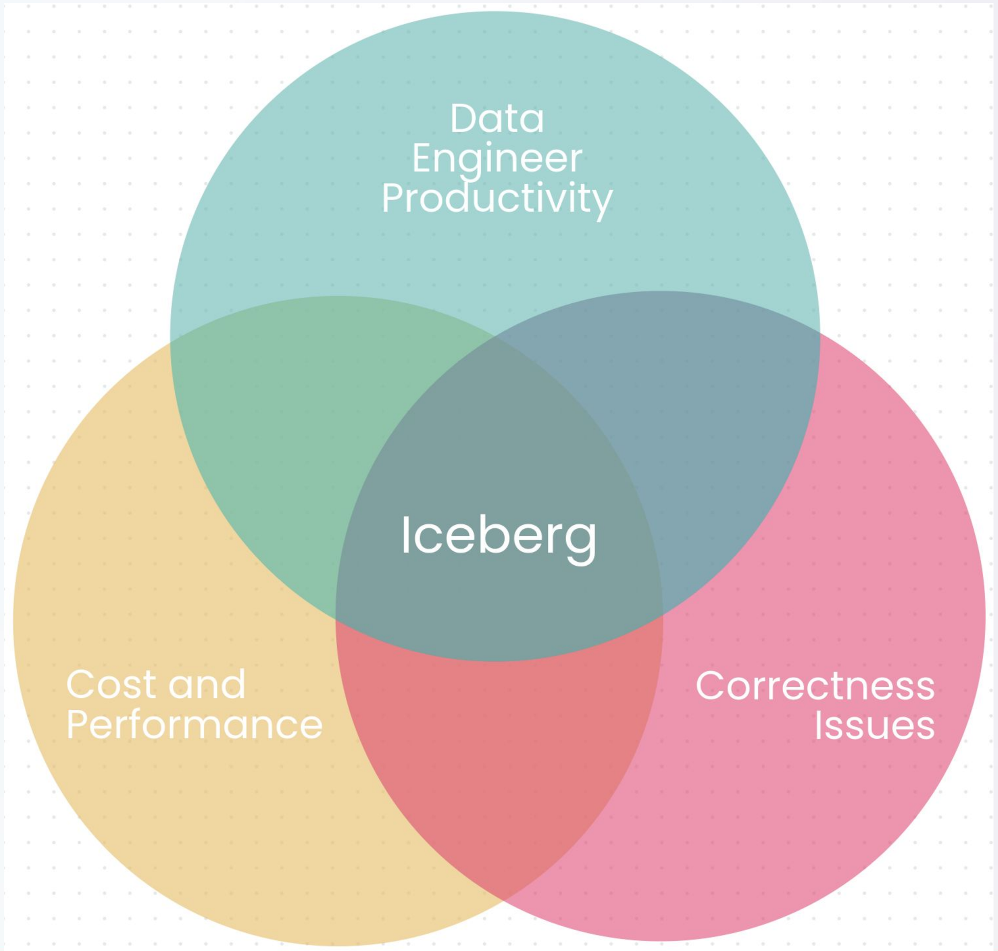
Iceberg 101

Ryan Blue
May 2023

Scan for an Iceberg cheat sheet for Spark or Trino

Tabular

Netflix problems



Iceberg is an open standard for tables with SQL behavior

The importance of an open standard

Commercial investment

snowflake

Google Big Query

aws

Starburst

dremio

CLOUDERA

Why does SQL behavior matter?

WHAT IF WE APPLIED
DATABASE FUNDAMENTALS
imgflip.com

Schema evolution

- Instantaneous – no rewrites
- Safe – no undead columns 🧟
- Saves days of headache

Hidden partitioning

- No silent correctness bugs
- No conversion mistakes
- Fast queries without needing an expert or DBA

FIX MY QUERY THAT'S TAKING FOREVER
I TOLD YOU TO ADD PARTITION PREDICATES
I SHOULDN'T HAVE TO KNOW THE TABLE'S FILE LAYOUT TO QUERY IT
SPARK HAS ALWAYS BEEN THIS WAY, GET USED TO IT
FIX THE DATABASE NOT THE ANALYST
imgflip.com

Iceberg should be invisible

Avoid unpleasant surprises

- No zombie columns
- Performance should not be mysterious

Don’t steal attention

- No rewriting to drop a column
- Don’t make people filter twice
- Fix problems without migration

What are the advantages of using Iceberg?

Expressive SQL

Declarative, row-level commands

- MERGE, UPDATE, and DELETE
- Let engines optimize plans
    - Dynamic partition pruning
    - Storage-partitioned joins

```sql
-- squash multiple updates
WITH updates AS (
    SELECT
        account_id,
        sum(amount) AS amount
    FROM transactions
    GROUP BY account_id
)
-- update or insert
MERGE INTO accounts a USING updates u
ON a.account_id = u.account_id
WHEN MATCHED THEN UPDATE
    SET a.balance = a.balance + u.amount
WHEN NOT MATCHED THEN INSERT *
```

Time travel and rollback

Every change is a snapshot

- History for debugging
- Rollback to known healthy states
- Incremental consumption

Tag snapshots for longer retention

```
-- time travel
SELECT
    sum(balance) AS bank_assets
FROM accounts
FOR TIMESTAMP AS OF "2023-04-01T08:00:00"

-- create a tag for the auditors
ALTER TABLE accounts
    CREATE TAG q1_2023 RETAIN 730 DAYS

-- roll back to a previous state
CALL system.rollback_to_snapshot(
    table => "bank.accounts",
    snapshot_id => 6123669799074O5967)
```

Tabular

Better engineering patterns

Branching

- Test and validate in context
  - How do you test a MERGE?
- Integrate audits into workflows

Transactions

- Only format supporting single-table
- Multi-table support coming soon

```
-- start a branch
ALTER TABLE accounts
    CREATE BRANCH test_new_transform
    RETAIN 14 DAYS

-- validate before publishing
SELECT
    count(1) AS bad_rows
FROM accounts
FOR VERSION AS OF test_new_transform
WHERE account_id IS NULL
```

Declarative data engineering

Declare the ideal state

- Partitioning
- Clustering
- Tuning

… and let the infrastructure get there itself

Unlocks **automatic optimization**

```
-- schema & layout
CREATE TABLE accounts (
    account_id bigint,
    balance decimal(12, 2))
PARTITIONED BY (
    bucket(4, account_id))

-- distribution & clustering
ALTER TABLE accounts
WRITE DISTRIBUTED BY PARTITION
      LOCALLY ORDERED BY account_id

-- tune tables, not jobs
ALTER TABLE accounts SET TBLPROPERTIES (
    "write.parquet.dict-size-bytes"="...")
```

And more . . .

Performance

- Automatic pruning
- Column-level filtering
- Indexed metadata – fast query plans

Portable

- PyIceberg CLI and Python SDK
- No JVM or Spark-specific features

Flexible update strategies

- Eager – rewrite to optimize reads (*copy-on-write*)
- Lazy – defer work to read time (*merge-on-read*)
- Background – optimize with services

Layout evolution

```
stack.pop()
```

What does Iceberg unlock?

Cloud-native data architecture

Flexible compute

- Center of gravity – don’t move data
- Unify batch, streaming, and ad-hoc
- Any language or framework

SQL warehouse behavior

- Make people productive
- Strong guarantees
- Maintain data in place

Companies using and contributing to Iceberg

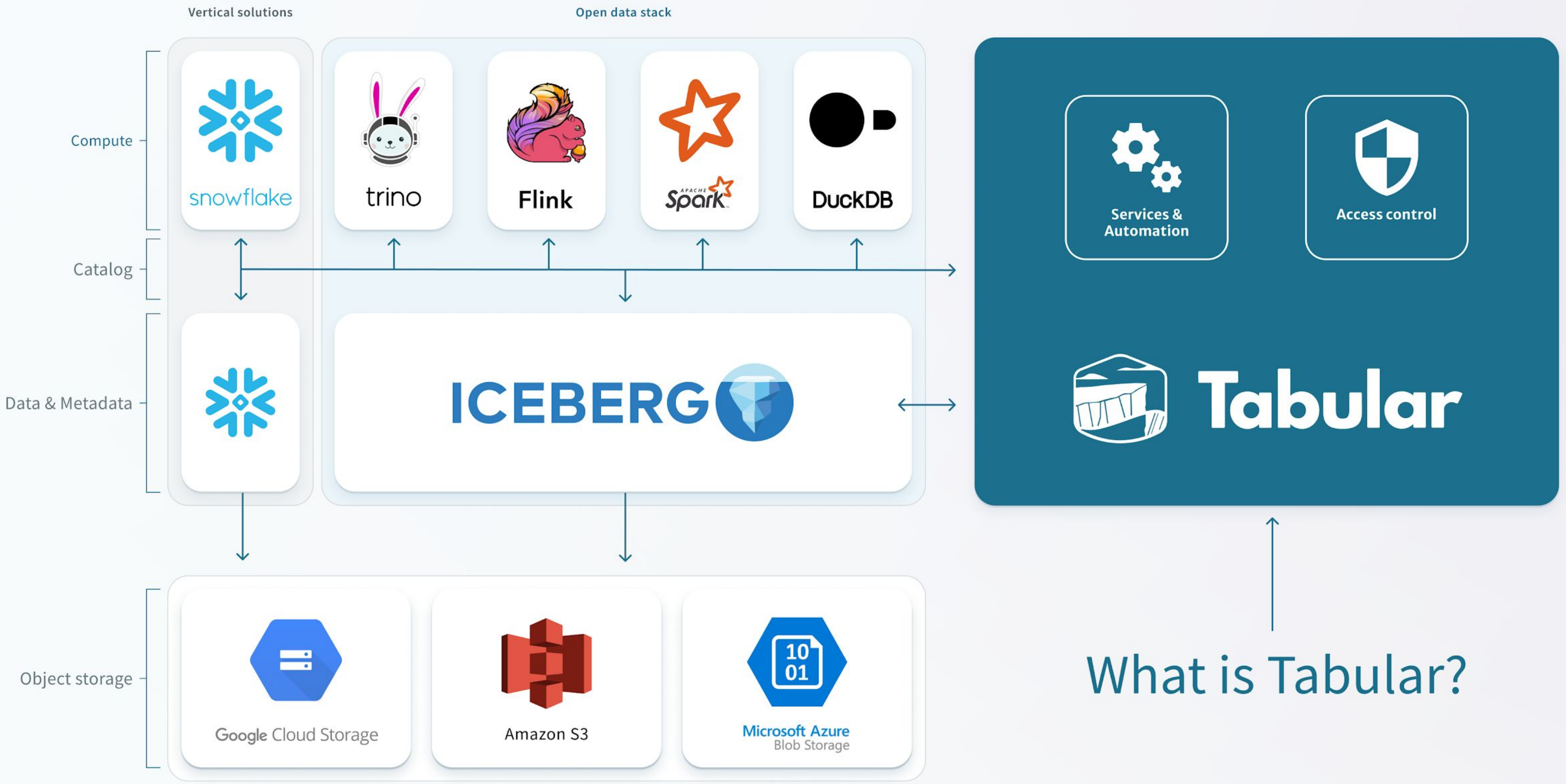
in

Adobe

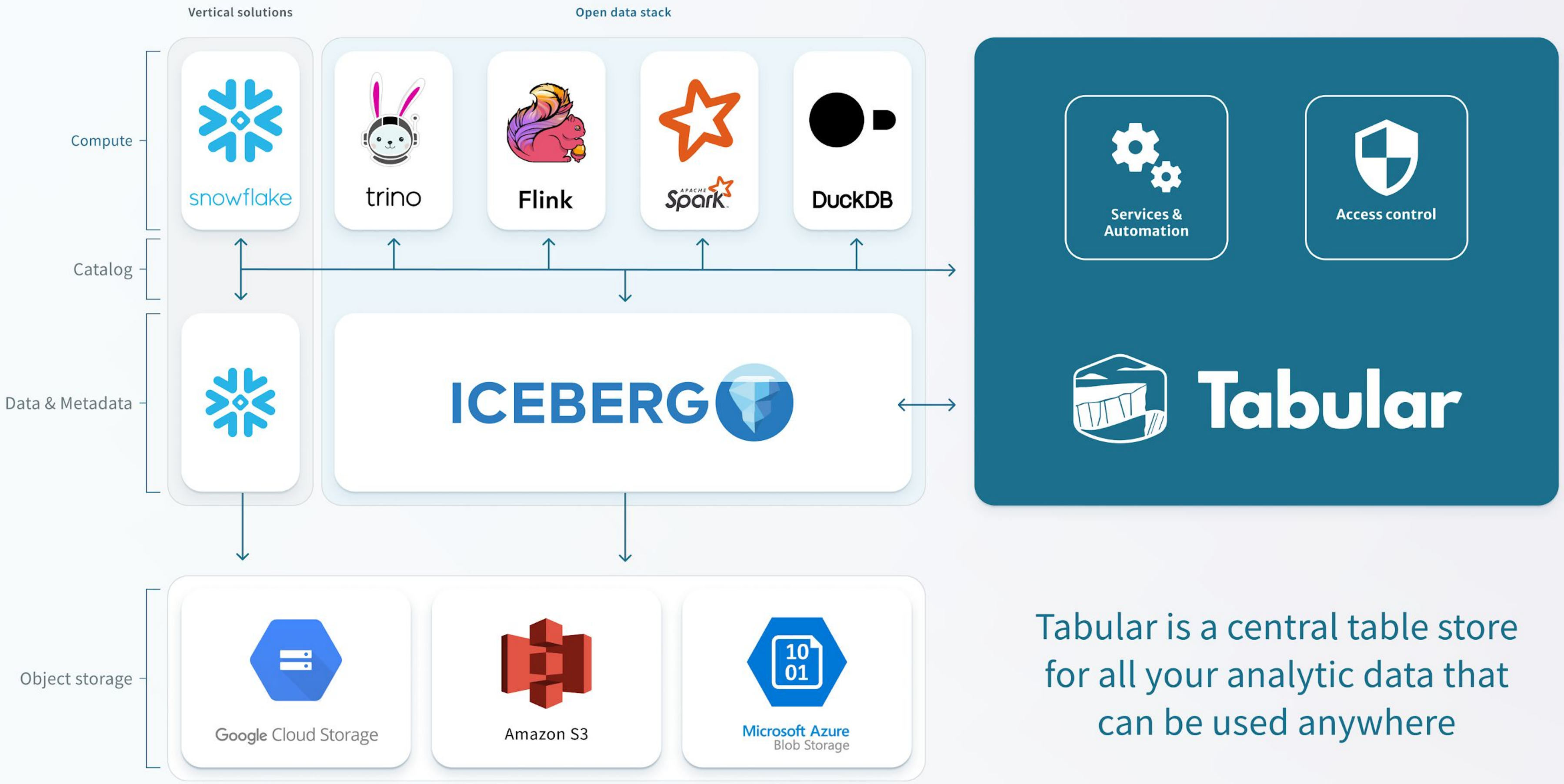
Expedia

salesforce

stripe



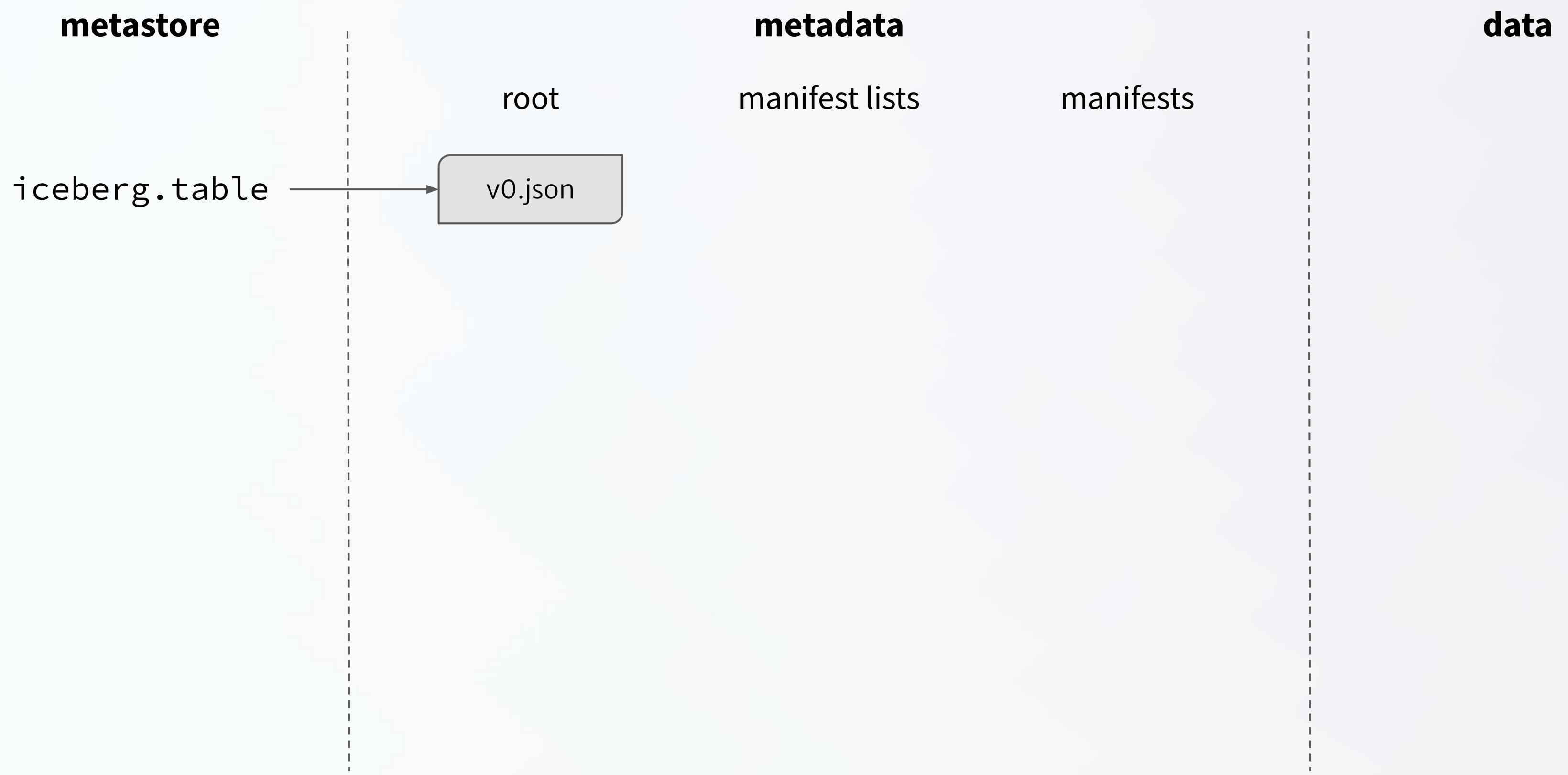
What is Tabular?



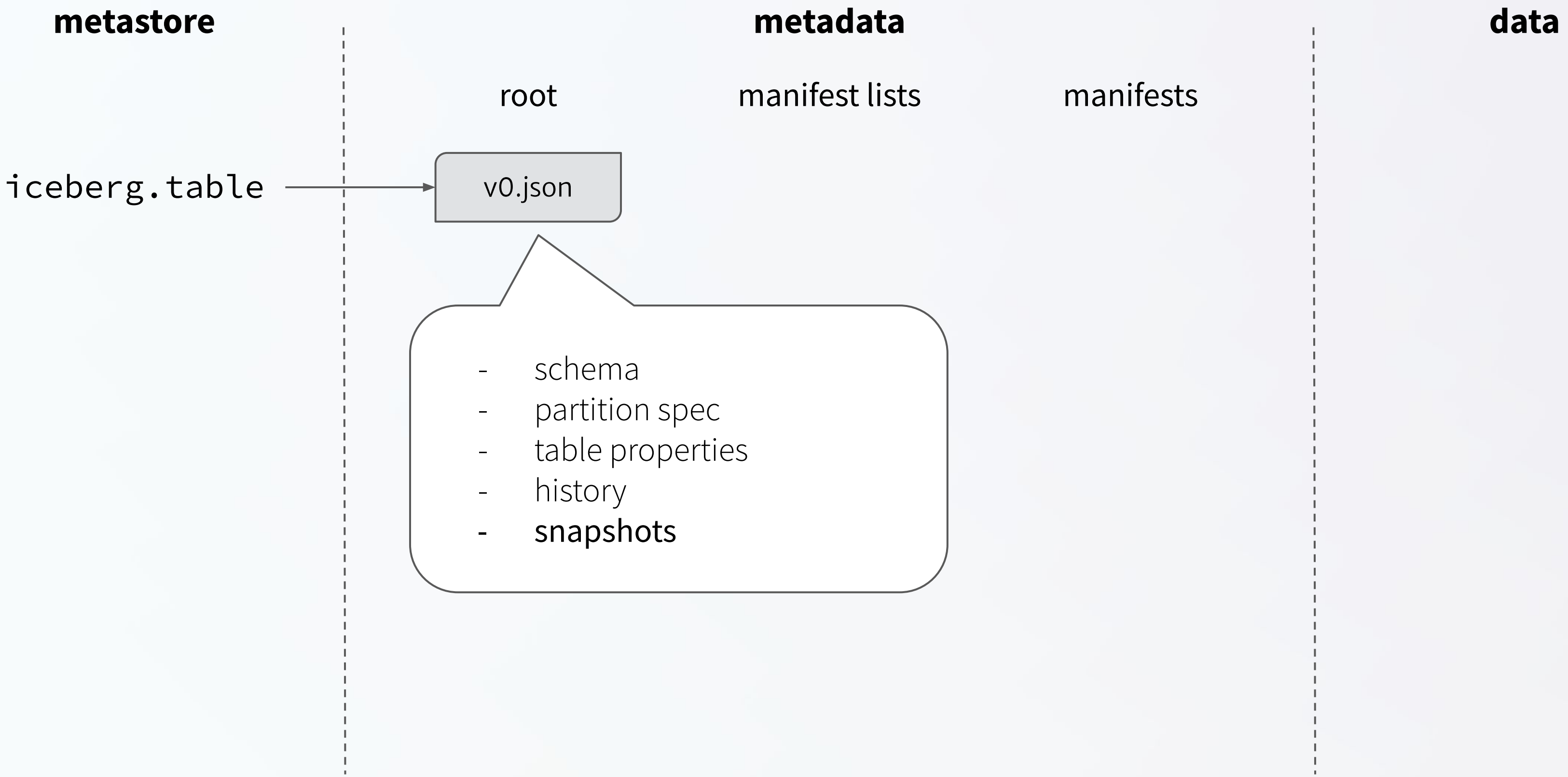
Tabular is a central table store for all your analytic data that can be used anywhere

Iceberg metadata structure

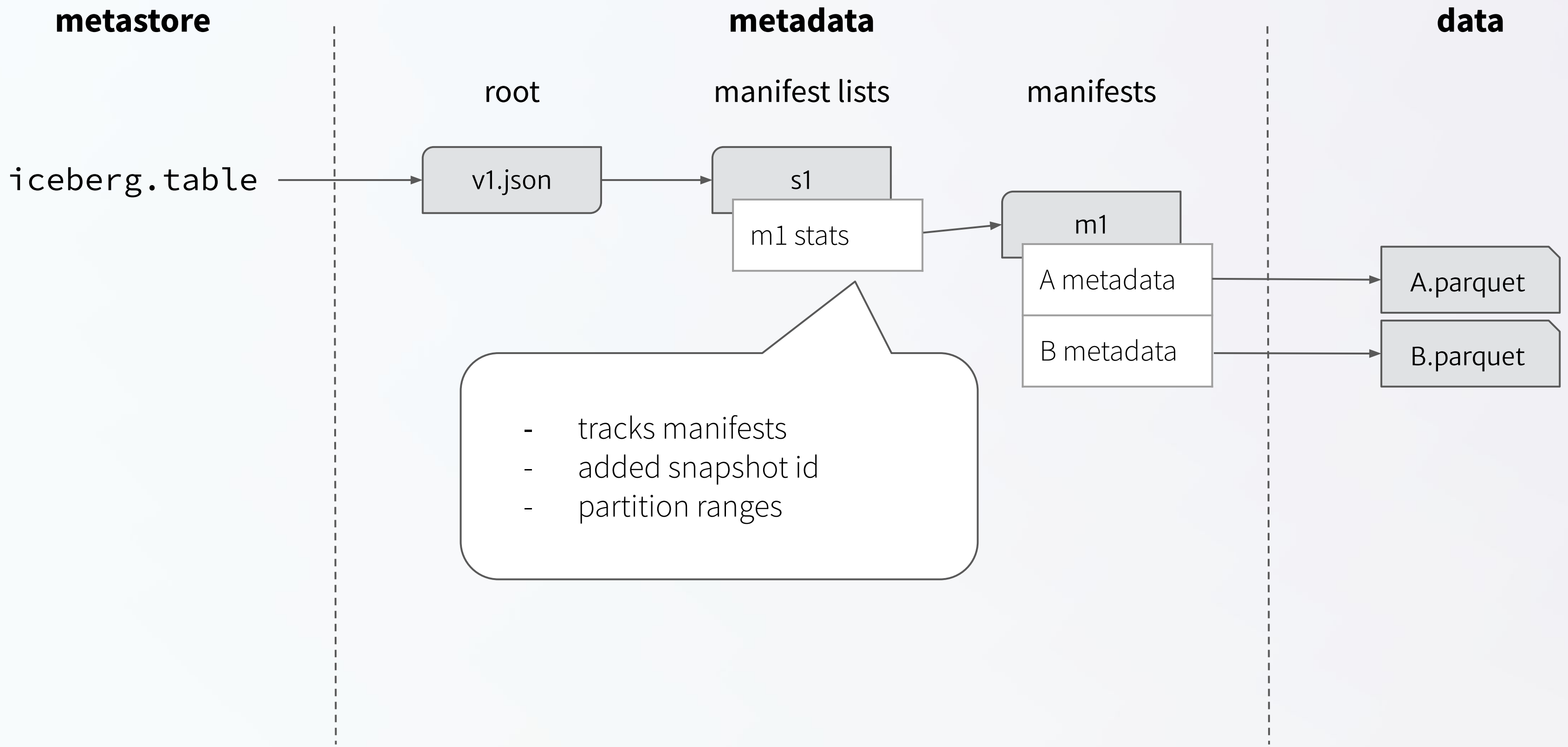
Metadata tree



Metadata tree

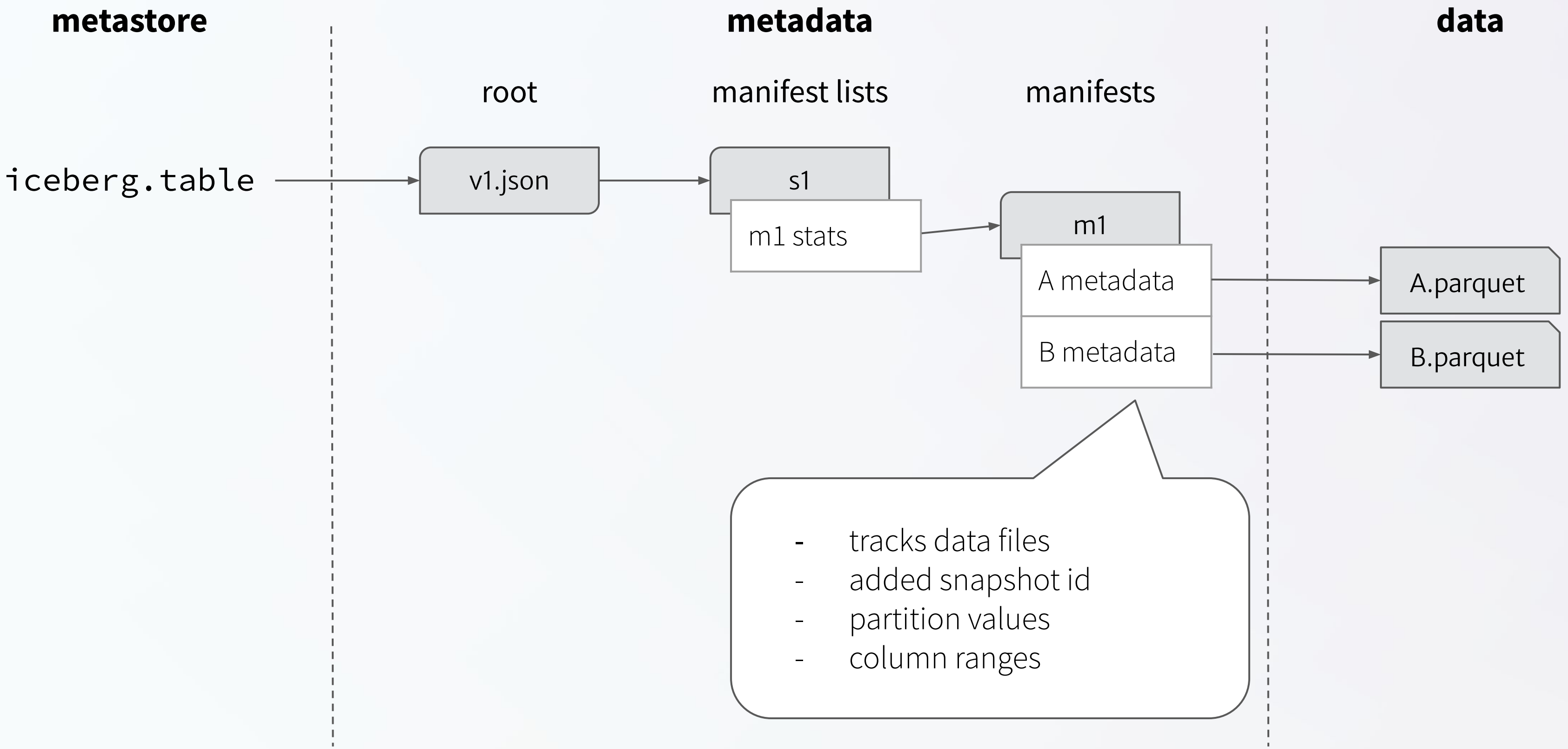


Metadata tree



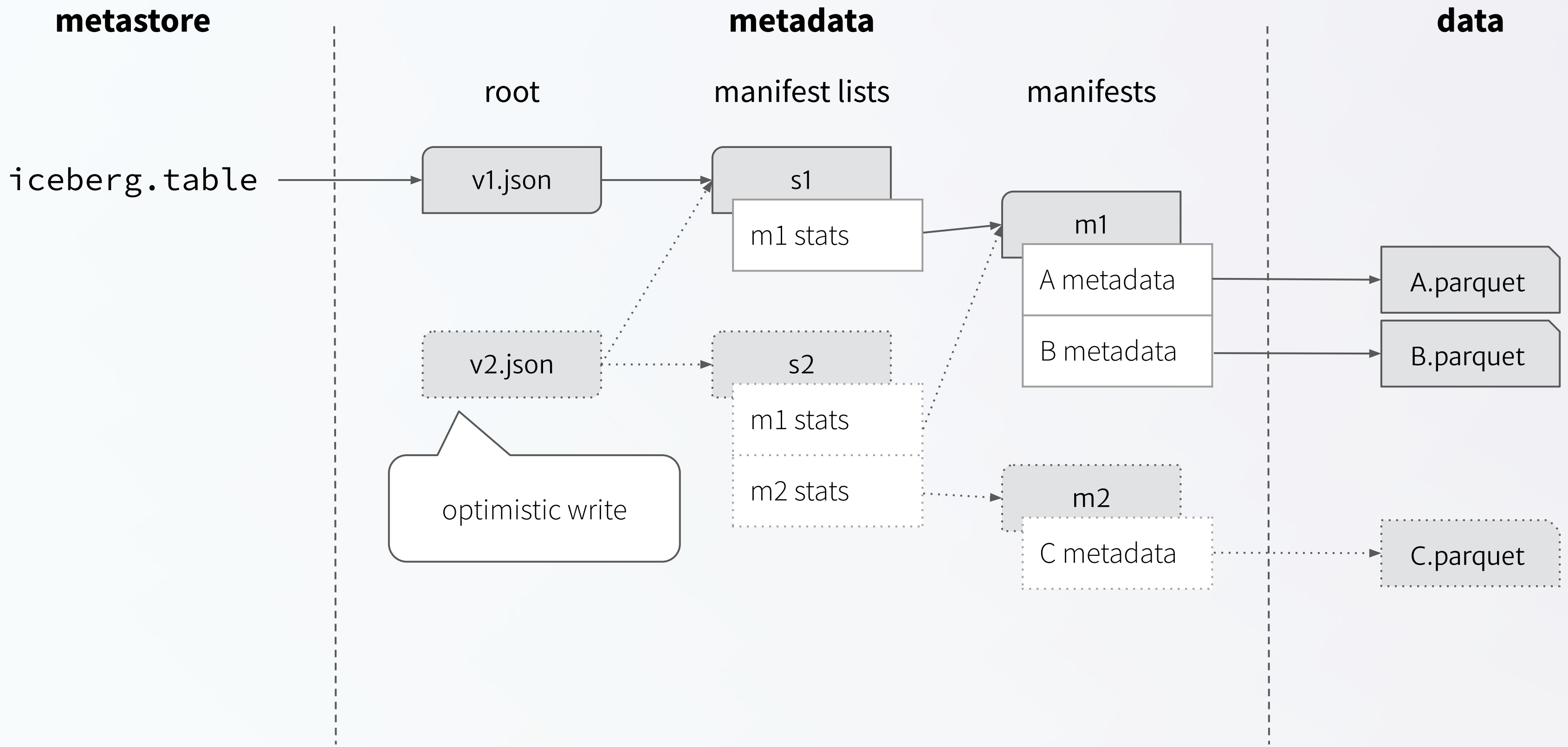
- tracks manifests
- added snapshot id
- partition ranges

Metadata tree

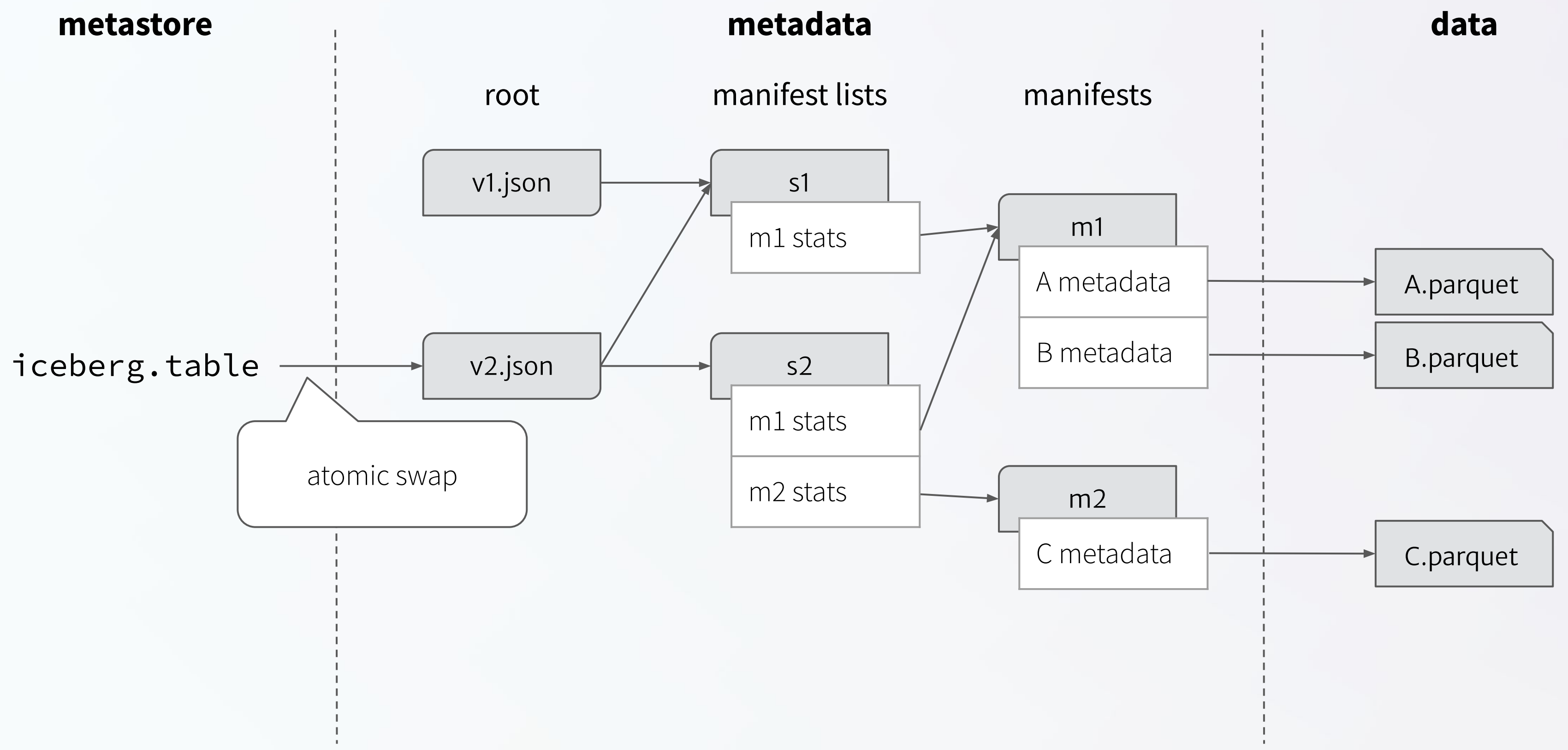


- tracks data files
- added snapshot id
- partition values
- column ranges

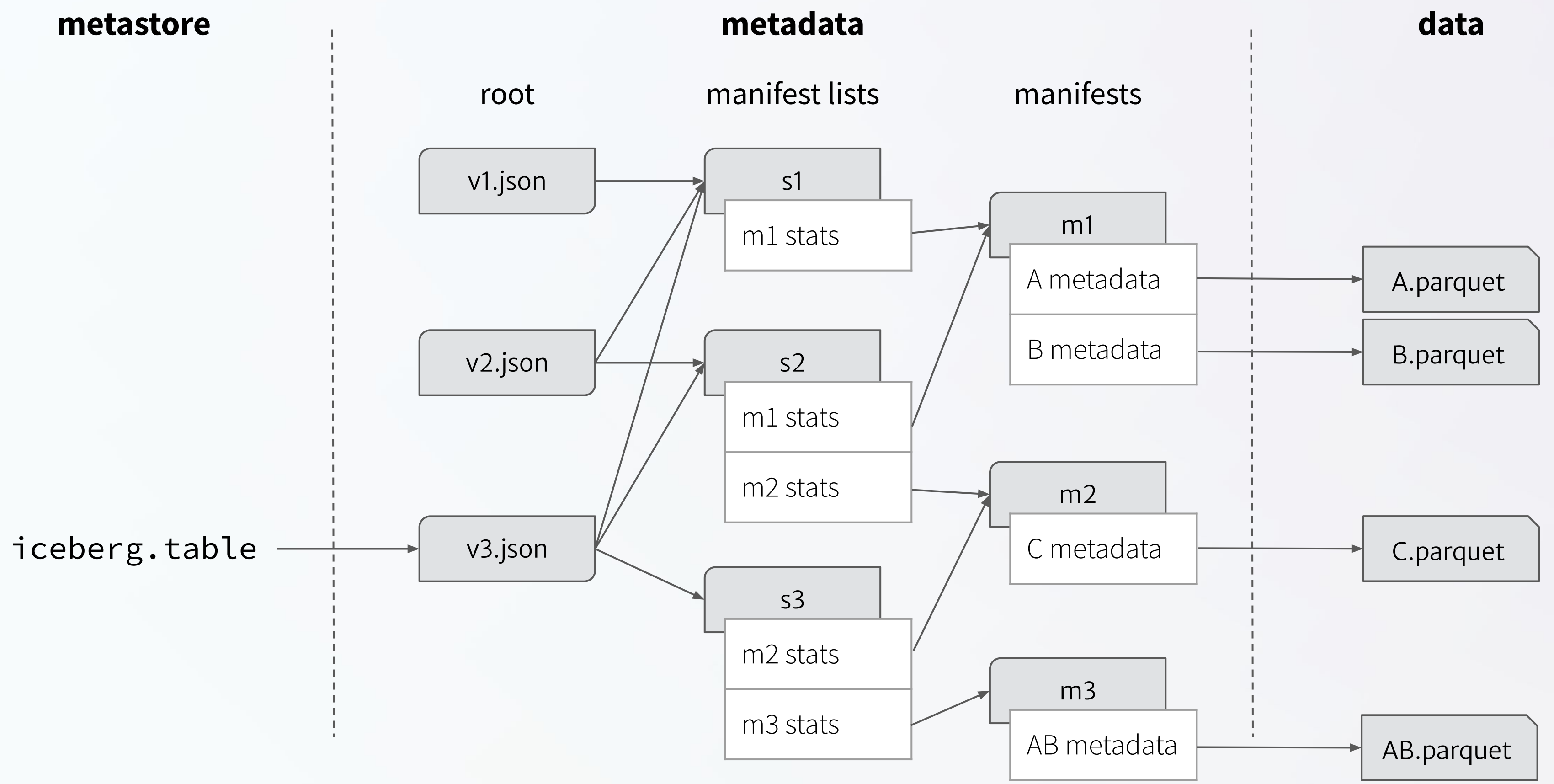
Metadata tree: Appending



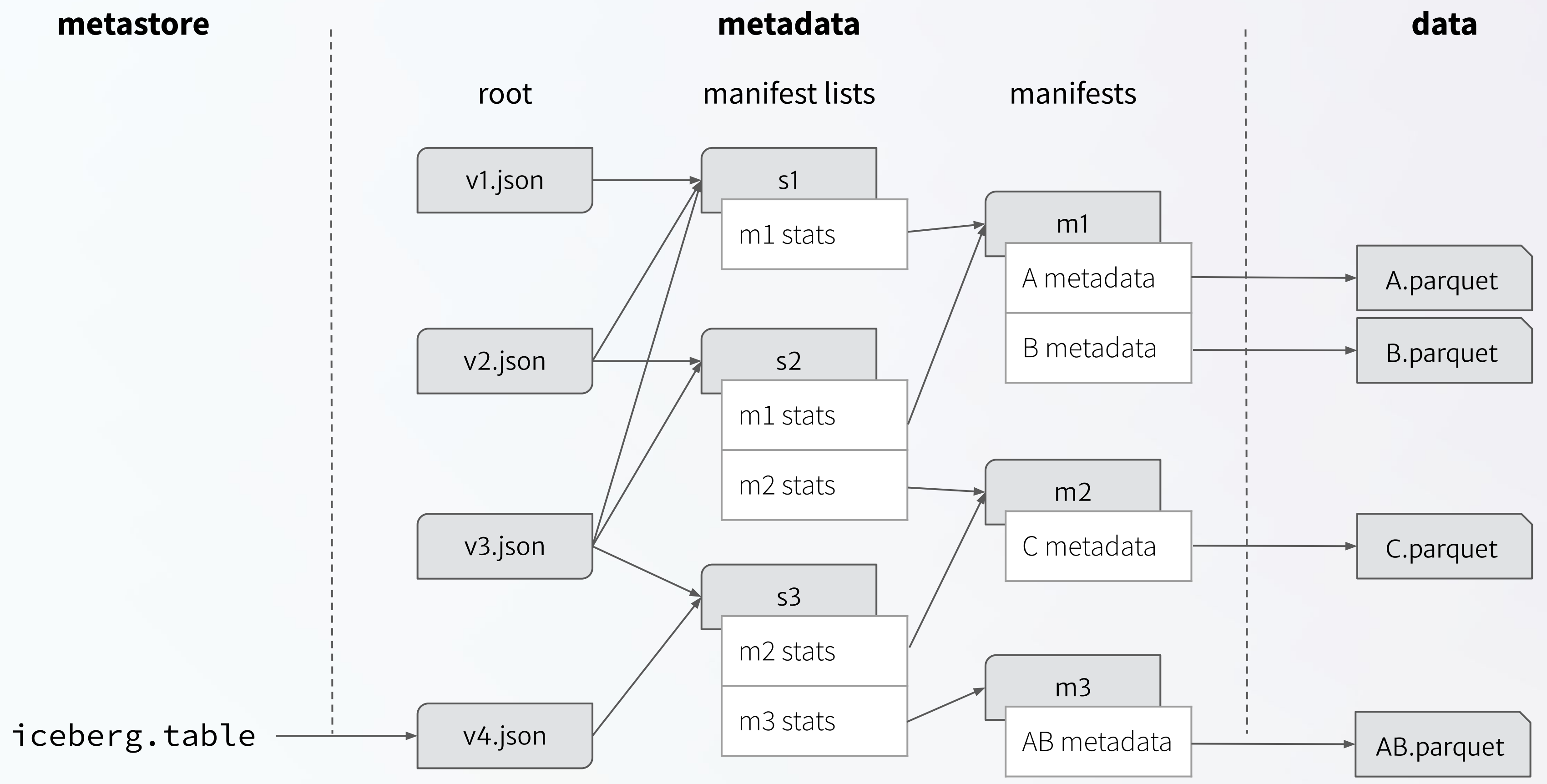
Metadata tree: Appending



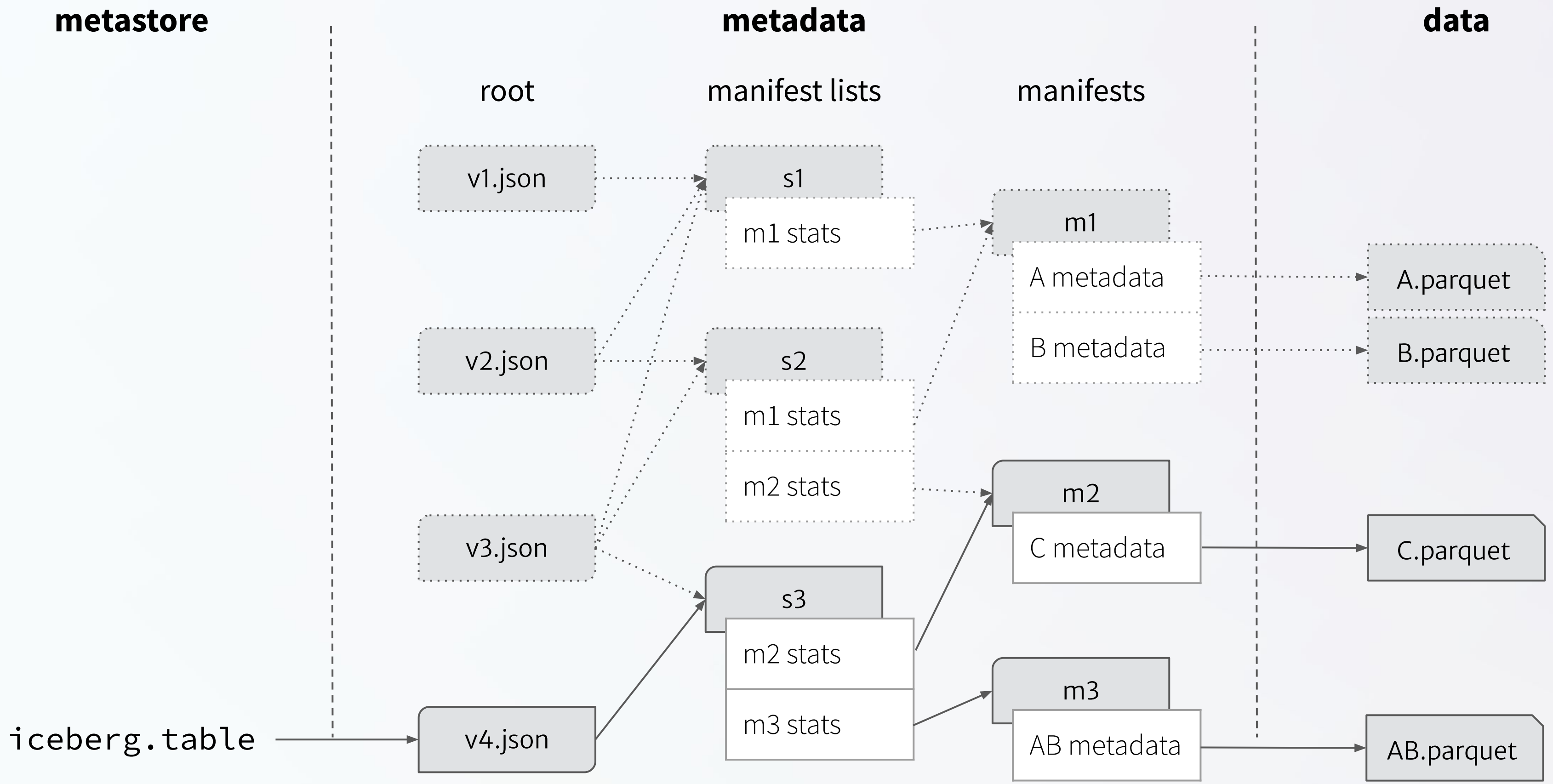
Metadata tree: Compaction



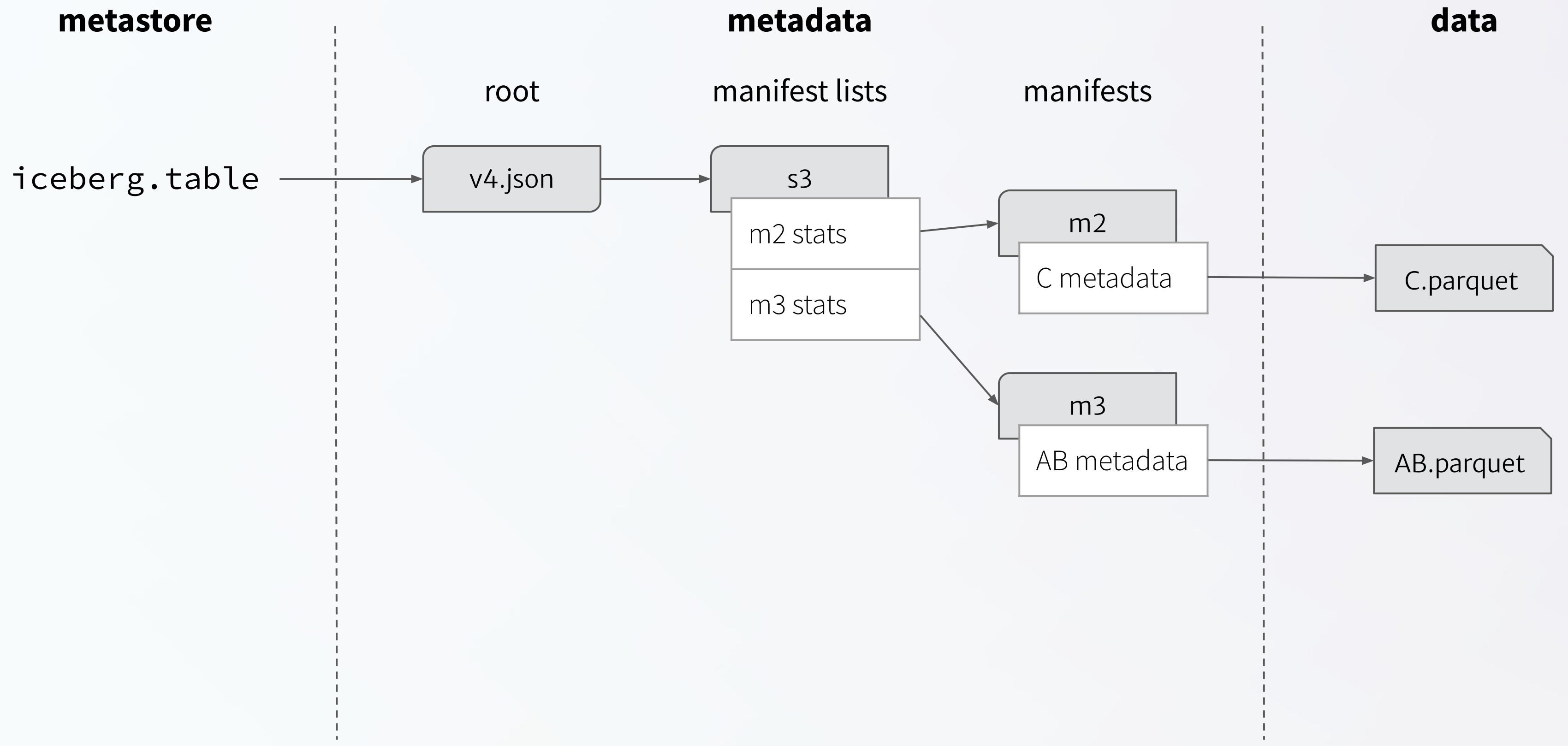
Metadata tree: Expiration



Metadata tree: Expiration



Metadata tree: Expiration



Questions?

Thanks for attending!

app.tabular.io/signup