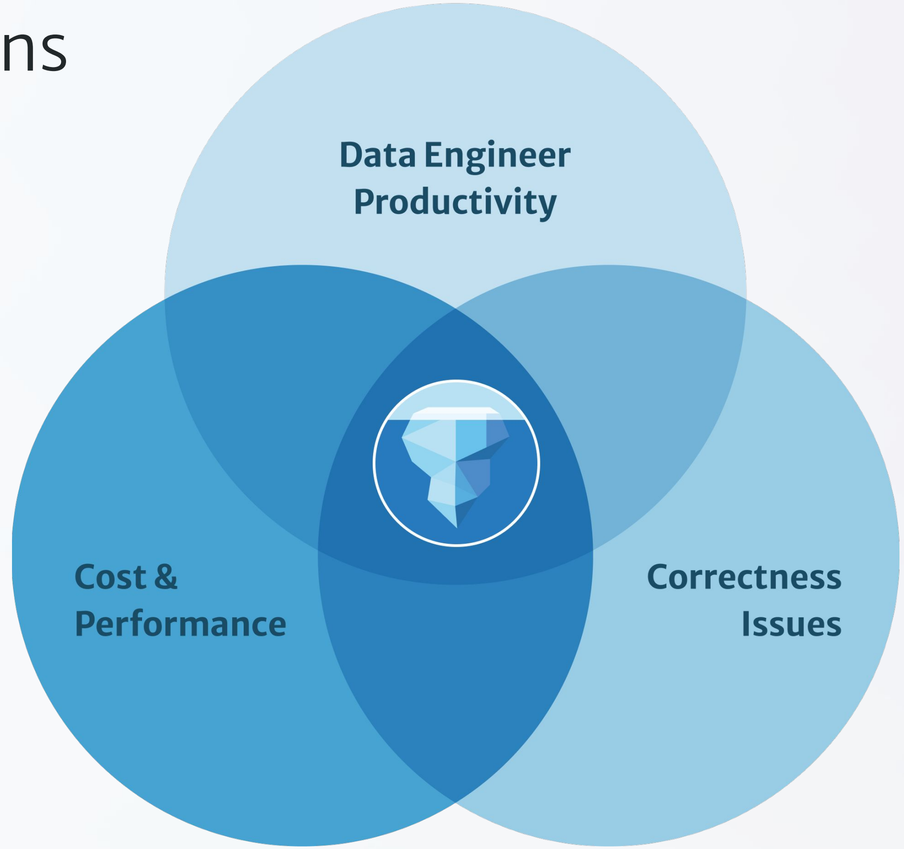
Apache Iceberg: The Happy Accident Disrupting the Data Industry

Ryan Blue
OSA Con 2023

Apache Iceberg Cookbook

3 rules:
-no wishing for death
-no falling in love
-no bring back dead people
I want to use Excel as a database
There are 4 rules
imgflip.com

Iceberg's origins



Iceberg: an open standard for tables with SQL behavior

Solve the hard problems

- High performance design for S3
- Full ACID semantics
- No unpleasant surprises (🧟 data)

Make people productive

- Time travel
- Hidden partitioning
- Row-level commands: MERGE, UPDATE, ...
- Automatic compaction, optimization

Are these the same change?

```
ALTER TABLE profiles
  RENAME COLUMN id TO profile_id
```

```
ALTER TABLE profiles
  DROP COLUMN id

ALTER TABLE profiles
  ADD COLUMN profile_id bigint
```

Are these the same change?

```
ALTER TABLE profiles
  RENAME COLUMN id TO profile_id
```

```
ALTER TABLE profiles
  DROP COLUMN id

ALTER TABLE profiles
  ADD COLUMN profile_id bigint
```

No! Renames preserve values

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

- Transactional guarantees (ACID)
- 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 IF WE APPLIED
DATABASE FUNDAMENTALS
imgflip.com

Design differences

Hive: track directories of data files

- Two sources of truth: HMS, file system
- Cannot perform atomic operations
- O(n) directory listing, **not cloud native**
- Can only filter by path/partition

Iceberg: track data files directly

- Persistent tree for flexible access
- Full ACID semantics
- O(1) metadata reads
- Filter by partition tuple, column stats

Additional problems

- Exposes physical problems to people
- Schema evolution is unsafe

Preventing problems

- Be invisible – use SQL behavior
- Take on hard problems

Iceberg is an open standard for tables with SQL behavior

Why does SQL behavior matter?

Are these the same change?

```
ALTER TABLE profiles                -- No changes
  DROP COLUMN profile_id

ALTER TABLE profiles
  ADD COLUMN profile_id bigint
```

Are these the same change?

```
ALTER TABLE profiles                    -- No changes
  DROP COLUMN profile_id

ALTER TABLE profiles
  ADD COLUMN profile_id bigint
```

No! Drop discards values

What does Iceberg unlock?

Expressive SQL

Declarative, row-level commands

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

```
-- 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 => 6123669799074059670)
```

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"="...")
```

Tabular

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

The happy accident

A happy accident

ACID in Spark, Trino, Flink, etc. => Universal analytic storage

From *Back to the Future (1985)*

- Shared database storage
  - One central repository (center of gravity)
  - Uniform governance & access control
- Any compute
  - Batch, streaming, and ad-hoc
  - Python script to data warehouse

snowflake

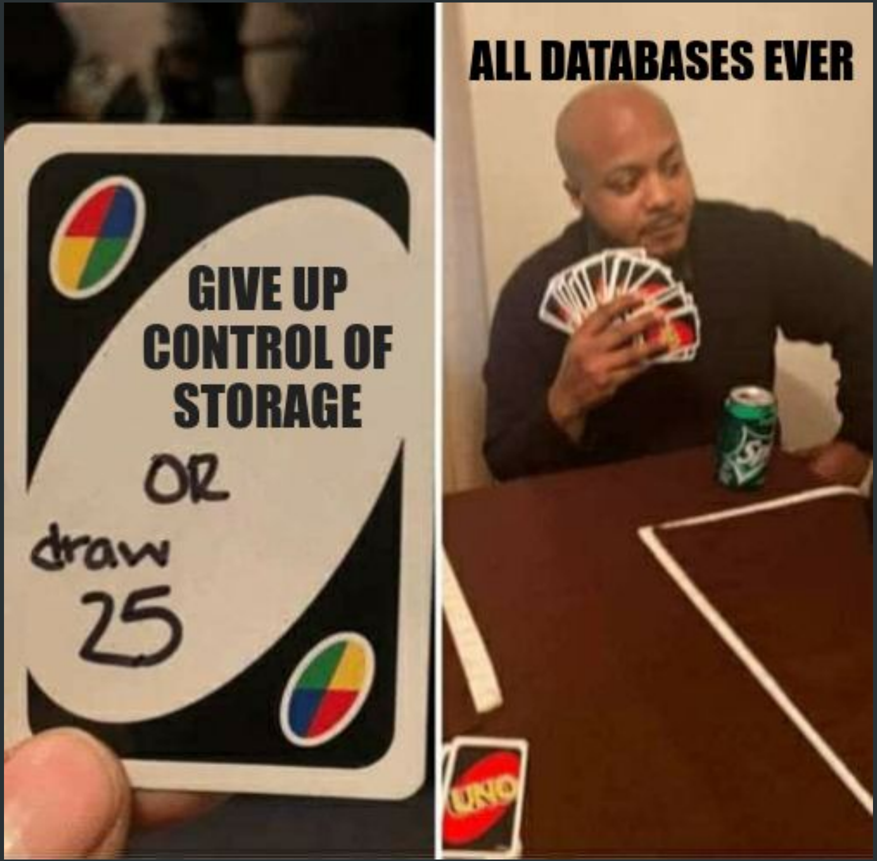
Google Big Query

aws

Starburst

dremio

CLOUDERA


GIVE UP CONTROL OF STORAGE
OR
draw 25
ALL DATABASES EVER
UNO

Unprecedented transformation

Shared analytic storage

- Storage provides opportunities for performance gains
- Tightly coupled storage & compute creates natural silos
  - Copying and syncing is hard
  - Testing new engines is costly
  - Migration is risky
- Control of storage is now **uncertain**

Iceberg disrupts the industry business model

Modular data architecture

Principles and best practices

- You control your storage
- You choose where to run compute workloads
- Like LEGO, everything fits together
- Secure the data, not the access
- Build on open standards
    - Iceberg tables and views
    - Iceberg REST catalog
    - OAuth2
- Declarative engineering approaches
- Data automation services

Lego blocks, by Alan Chia, from https://commons.wikimedia.org/wiki/File:Lego_Color_Bricks.jpg

Modular data architecture

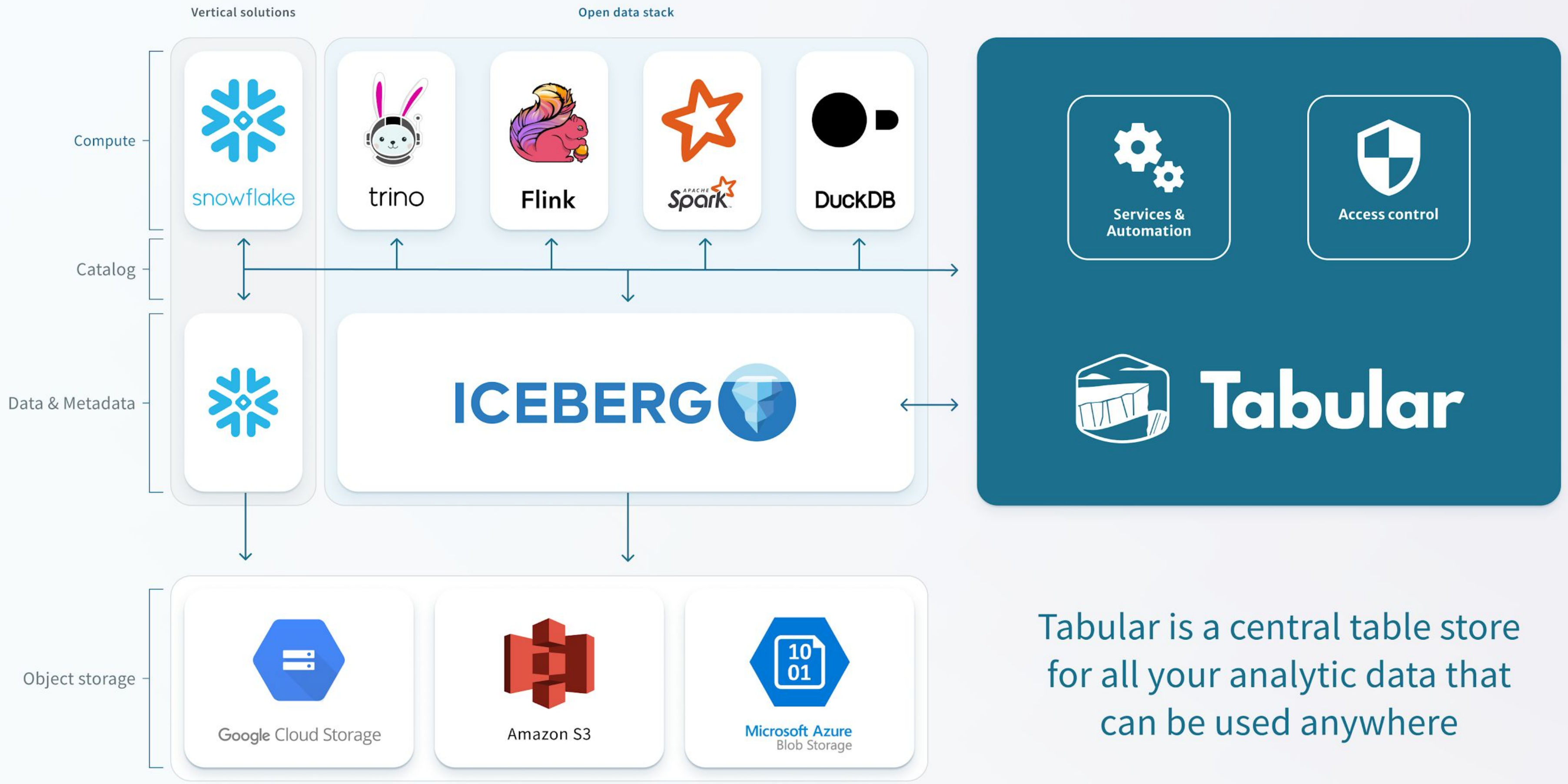
Principles and best practices

- **You control your storage**
- **You choose where to run compute workloads**
- Like LEGO, everything fits together
- Secure the data, not the access
- Build on open standards
  - Iceberg tables and views
  - Iceberg REST catalog
  - OAuth2
- Declarative engineering approaches
- Data automation services

Lego blocks, by Alan Chia, from https://commons.wikimedia.org/wiki/File:Lego_Color_Bricks.jpg

Tabular

Demand neutral or independent storage



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

Apache Iceberg Cookbook

Iceberg cheat sheets for Spark and Trino

Thank you for listening!