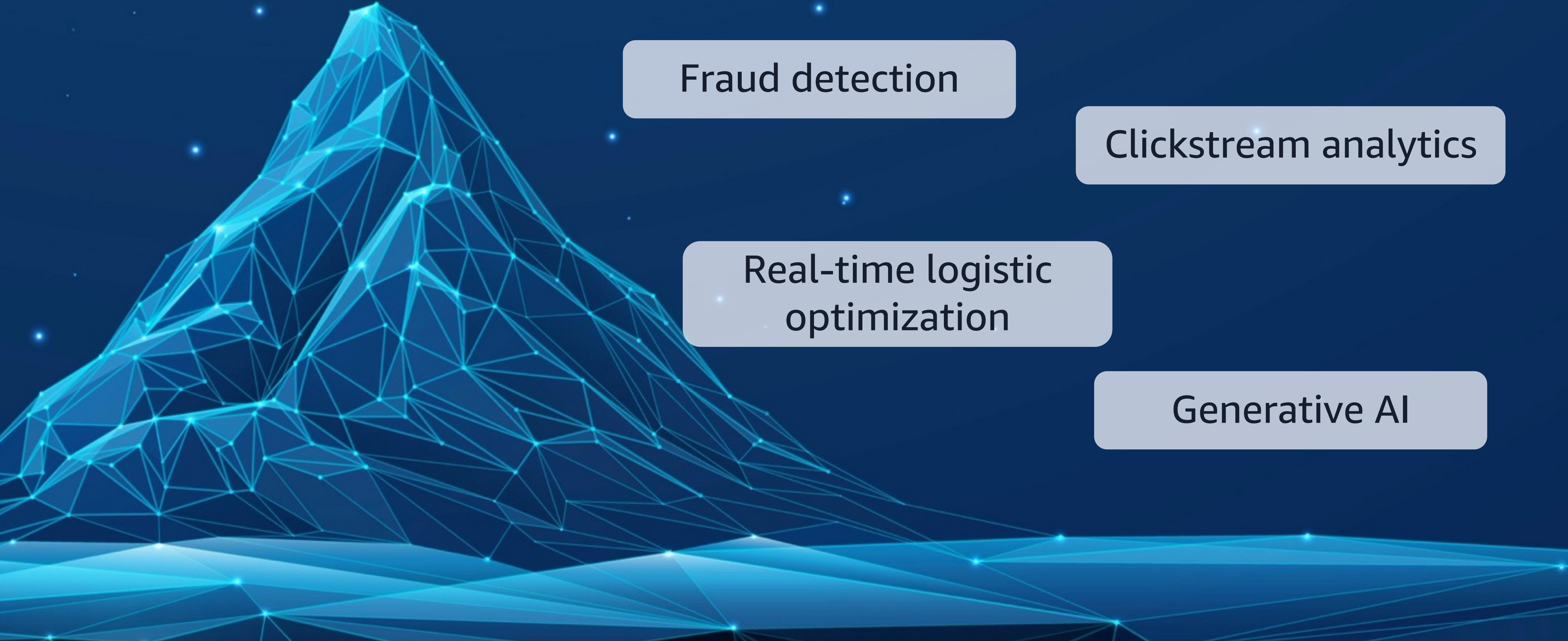


Personalization

Real-time dashboards

Fraud detection

Clickstream analytics

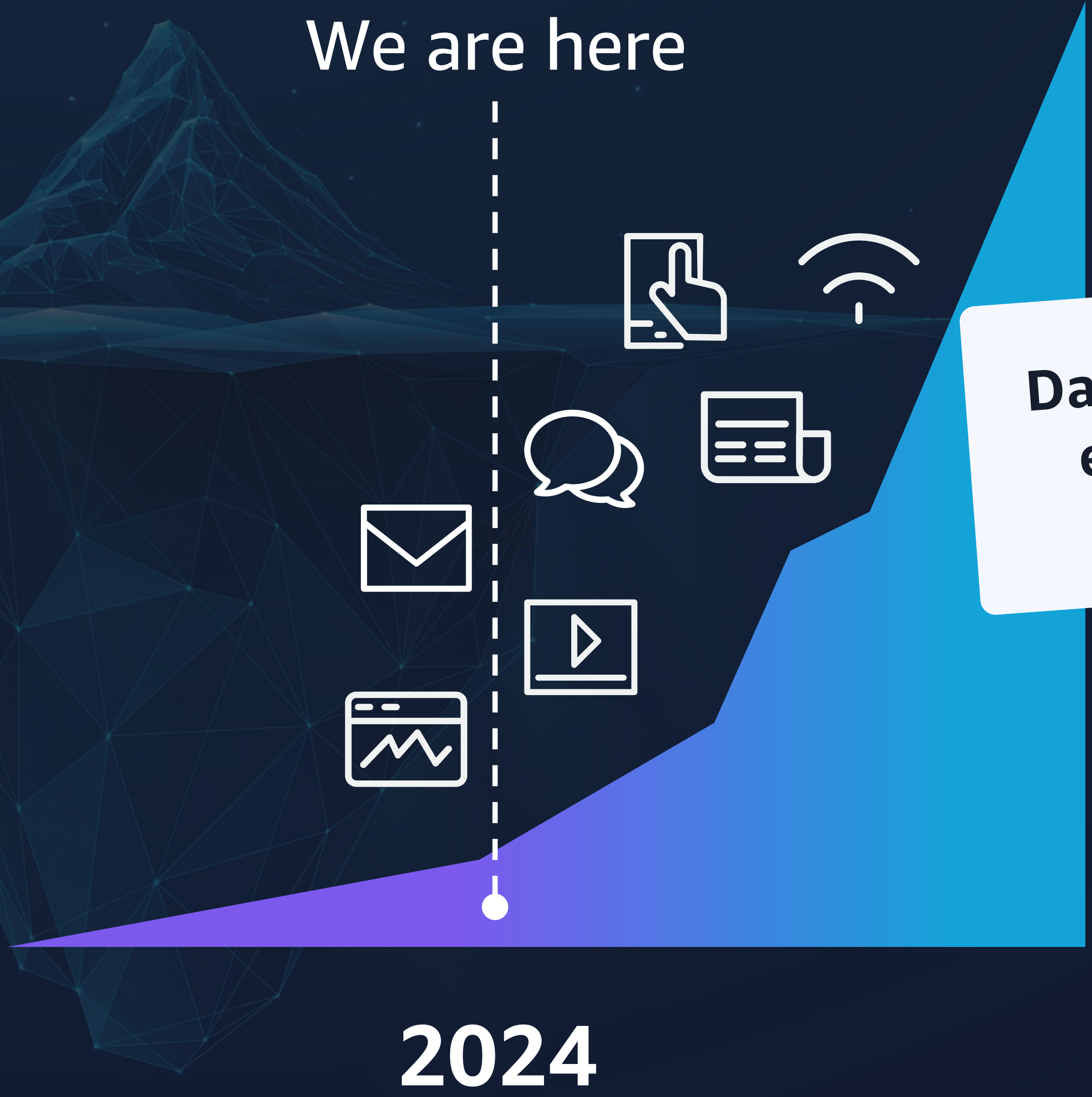
Real-time logistic optimization

Generative AI

Cool use cases 😎

Hidden (?) foundations

Architecture principles
Technology standards
Data management



Data grows >10x every 5 years

IDC, Data Age 2025

Norman Weisenburger

Techie by heart, connecting people and technology at AWS Retail/CPG customers in Germany as a Senior Solutions Architect at Amazon Web Services

in /in/normalerweise

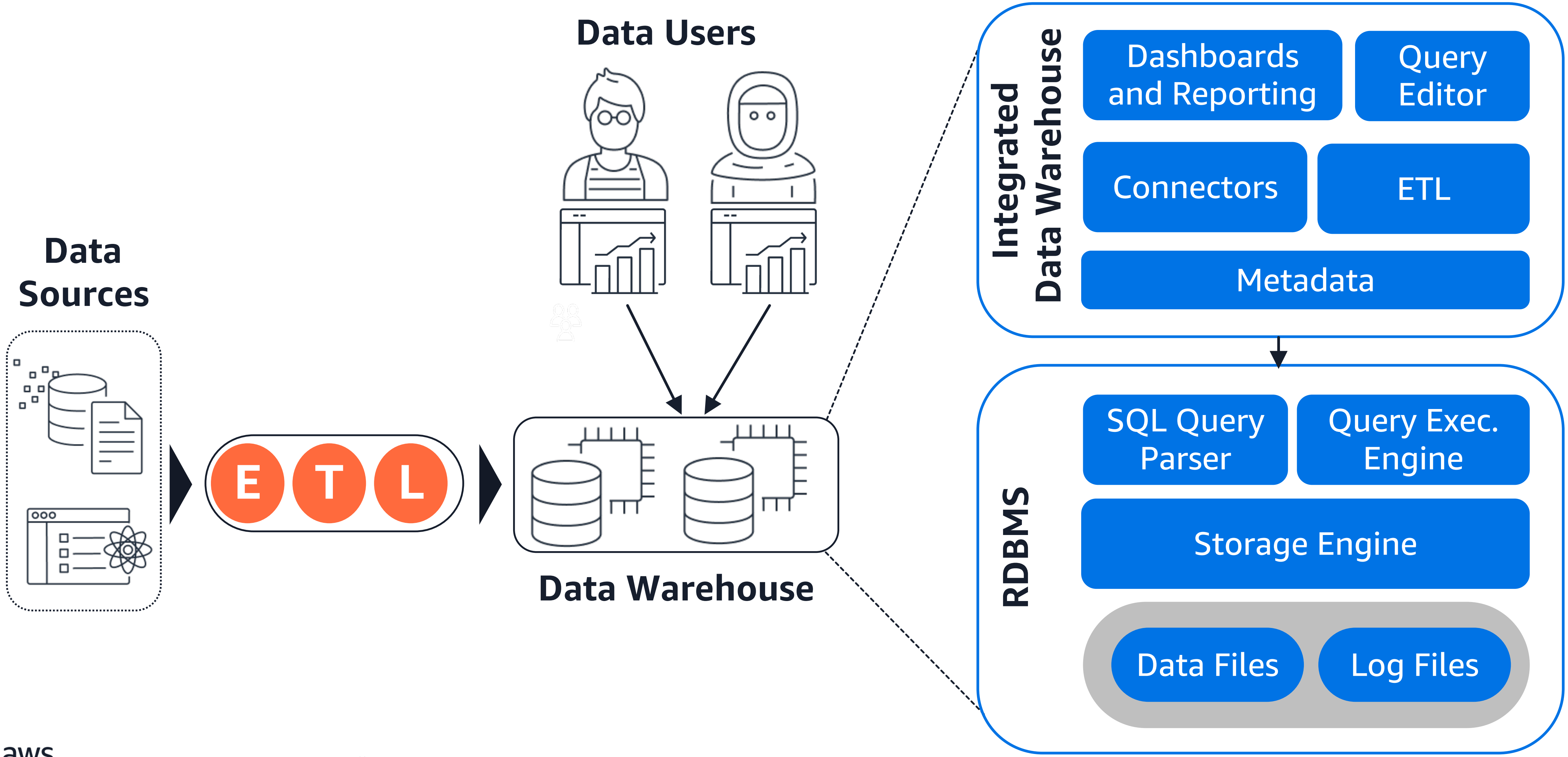
Beyond Warehouses: Transactional Data Lakes Power Modern Analytics

Norman Weisenburger

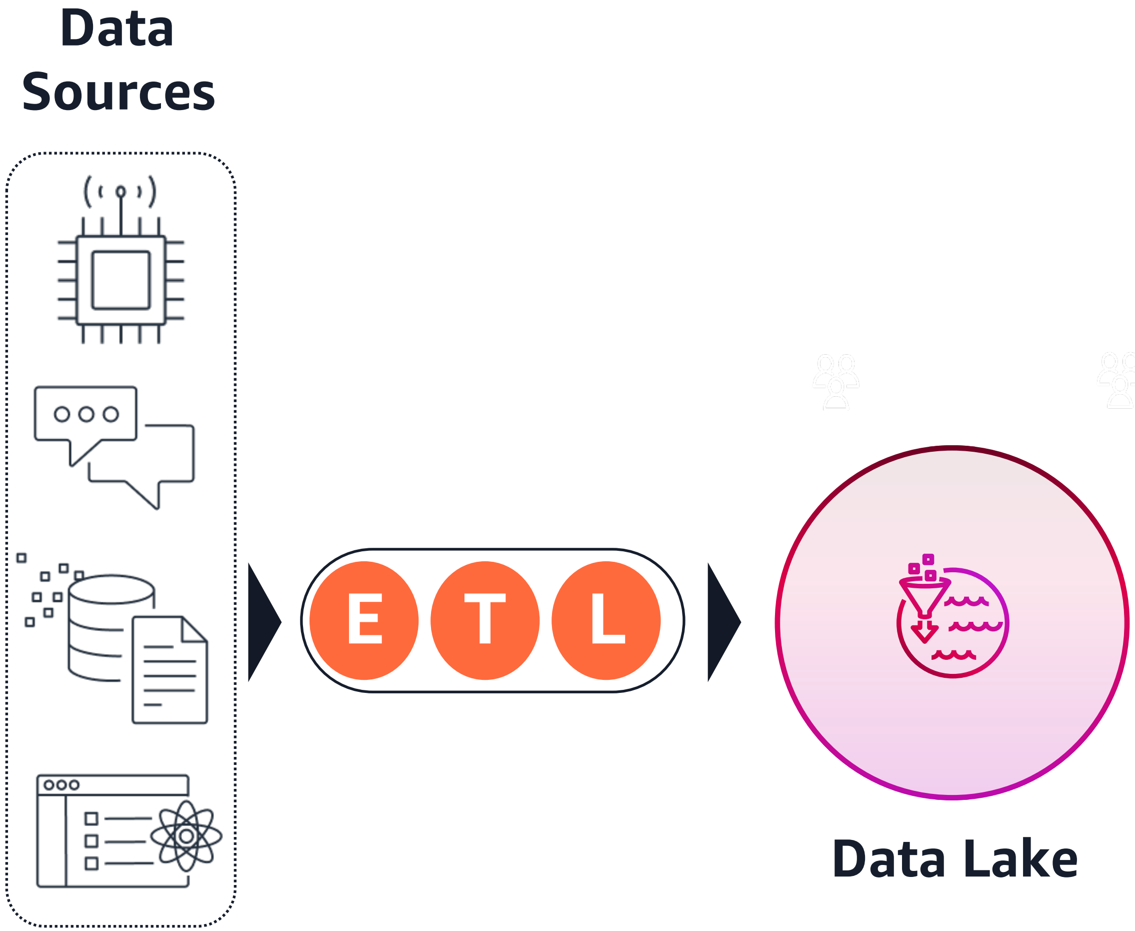
(he/him)

From Data Warehouses to Data Lakes

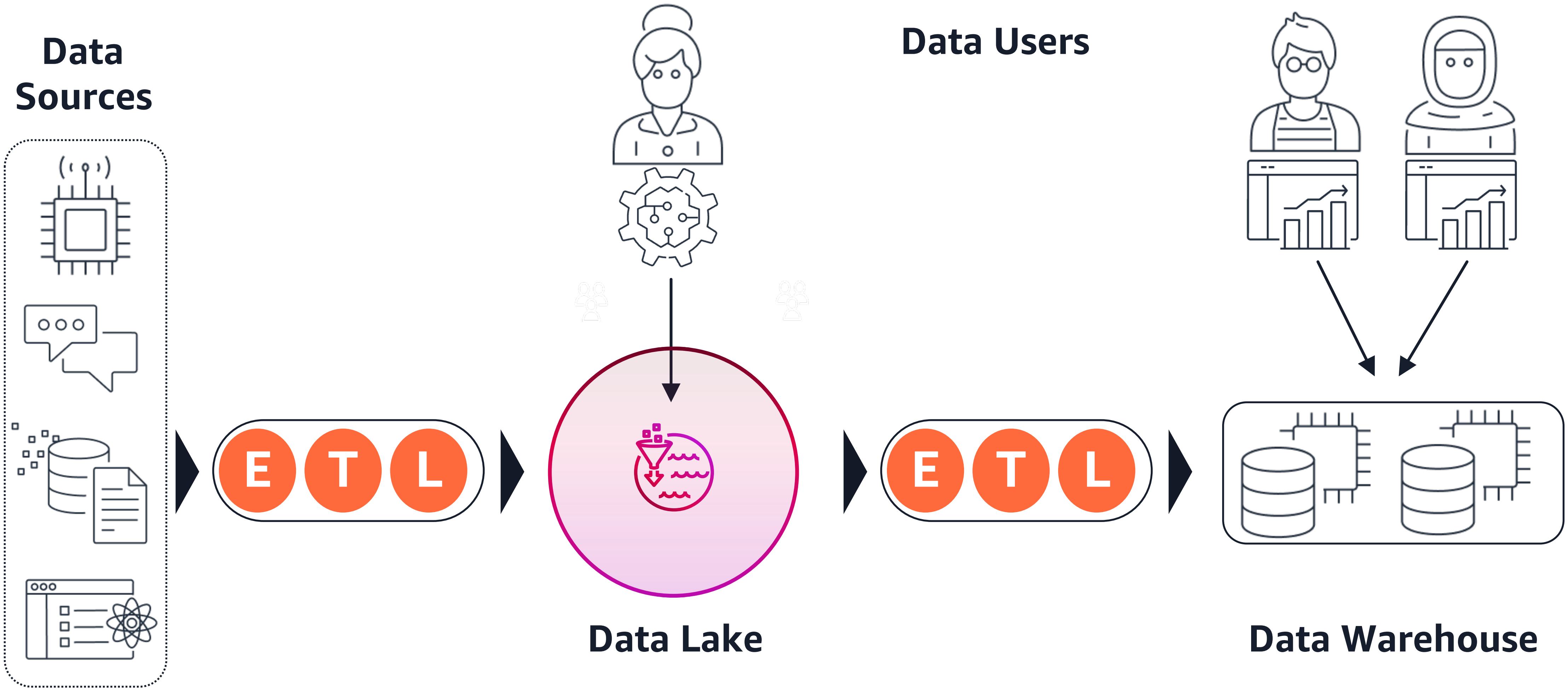
The Data Warehouse ~2006



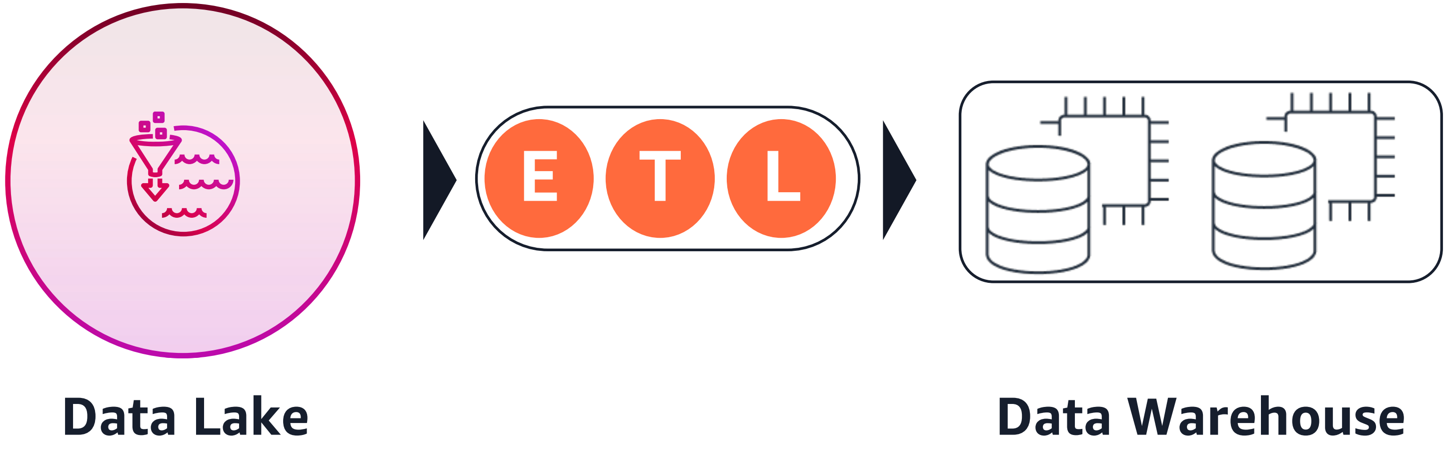
The rise of the Data Lake 2006+



The rise of the Data Lake 2006+



Challenges operating two independent systems



Data Lake

Data Warehouse

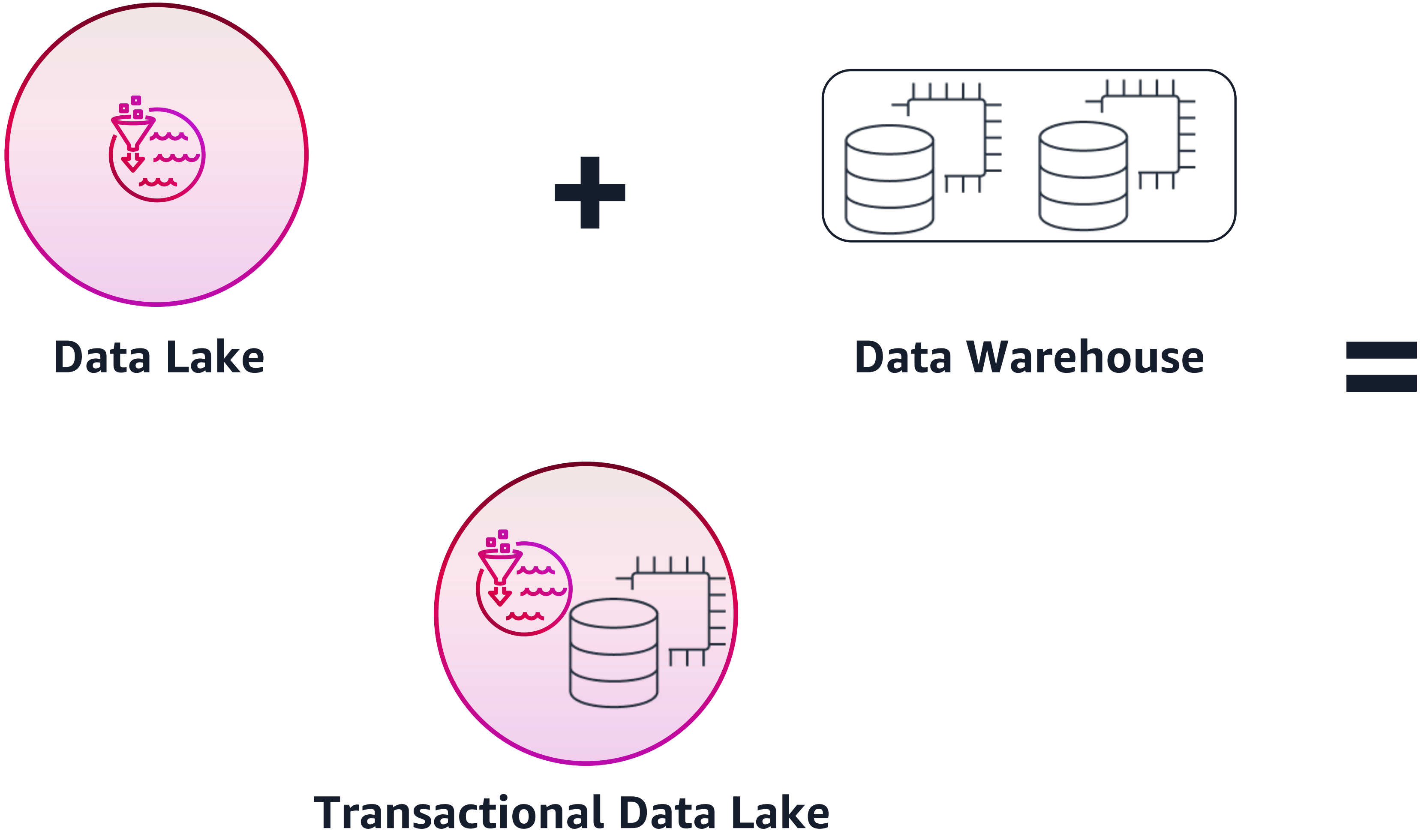
No single source of truth

Additional cost and complexity

Error prone security and governance

Transactional Data Lakes

1 + 1 = Transactional Data Lake



Data Lake

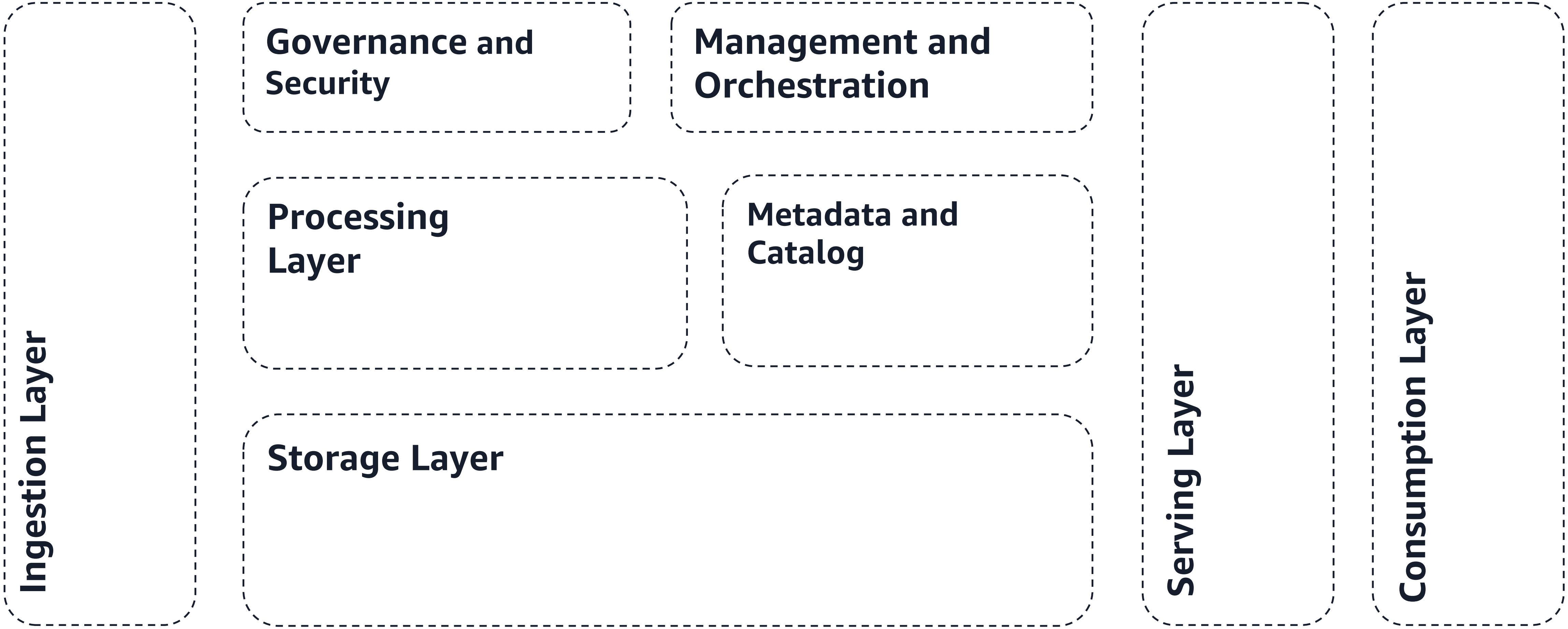
+

Data Warehouse

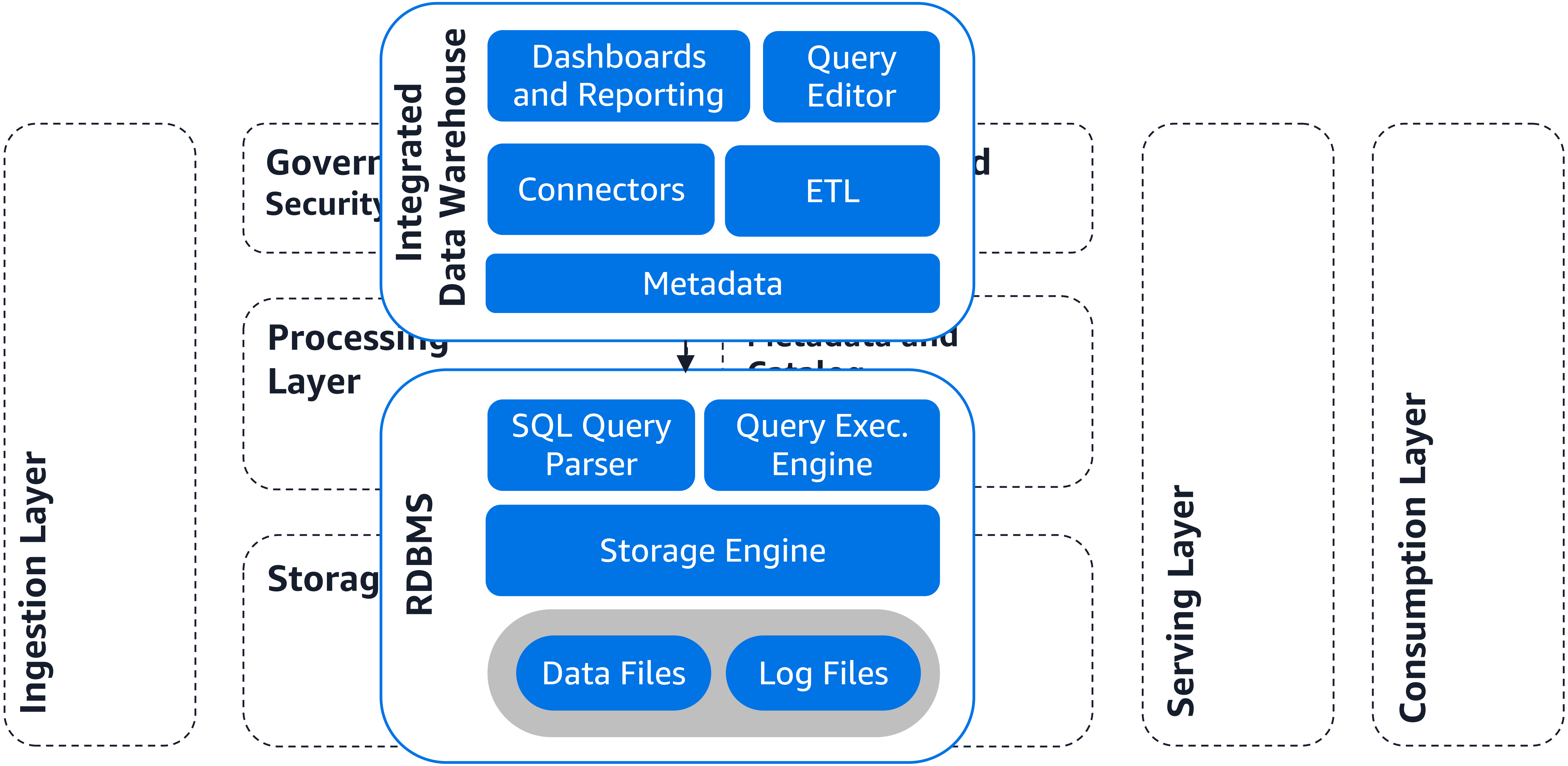
=

Transactional Data Lake

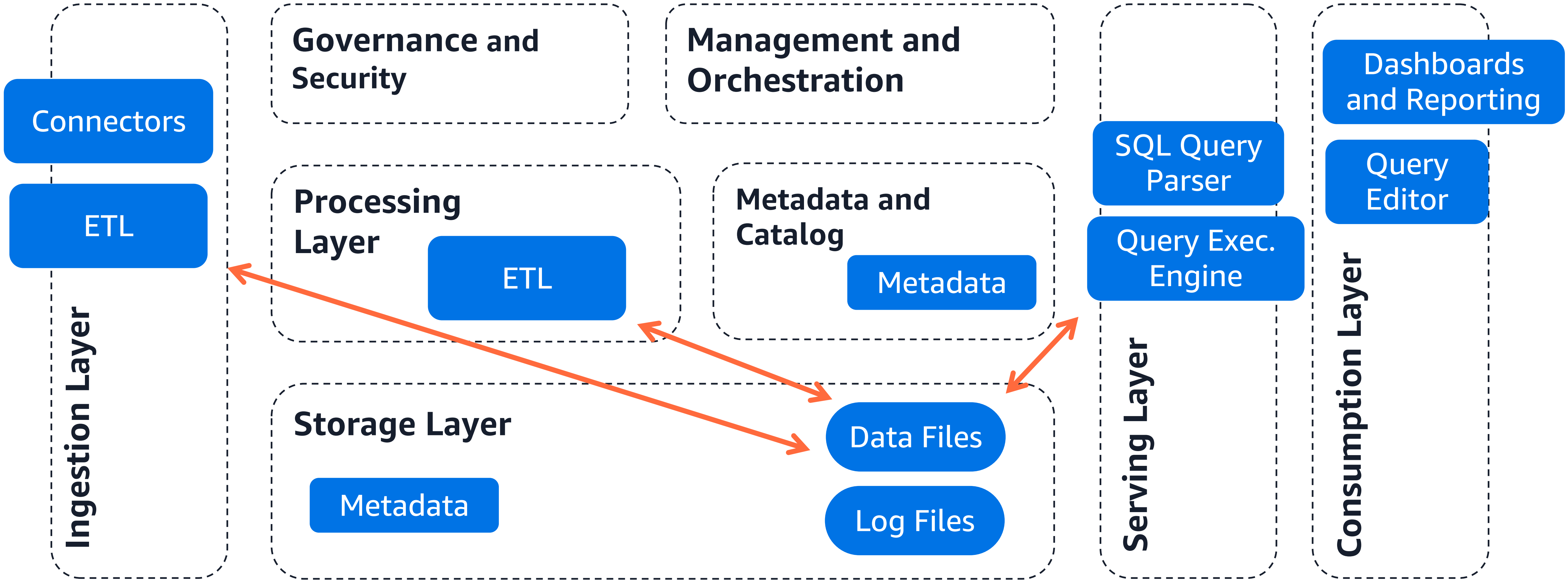
A typical data platform architecture



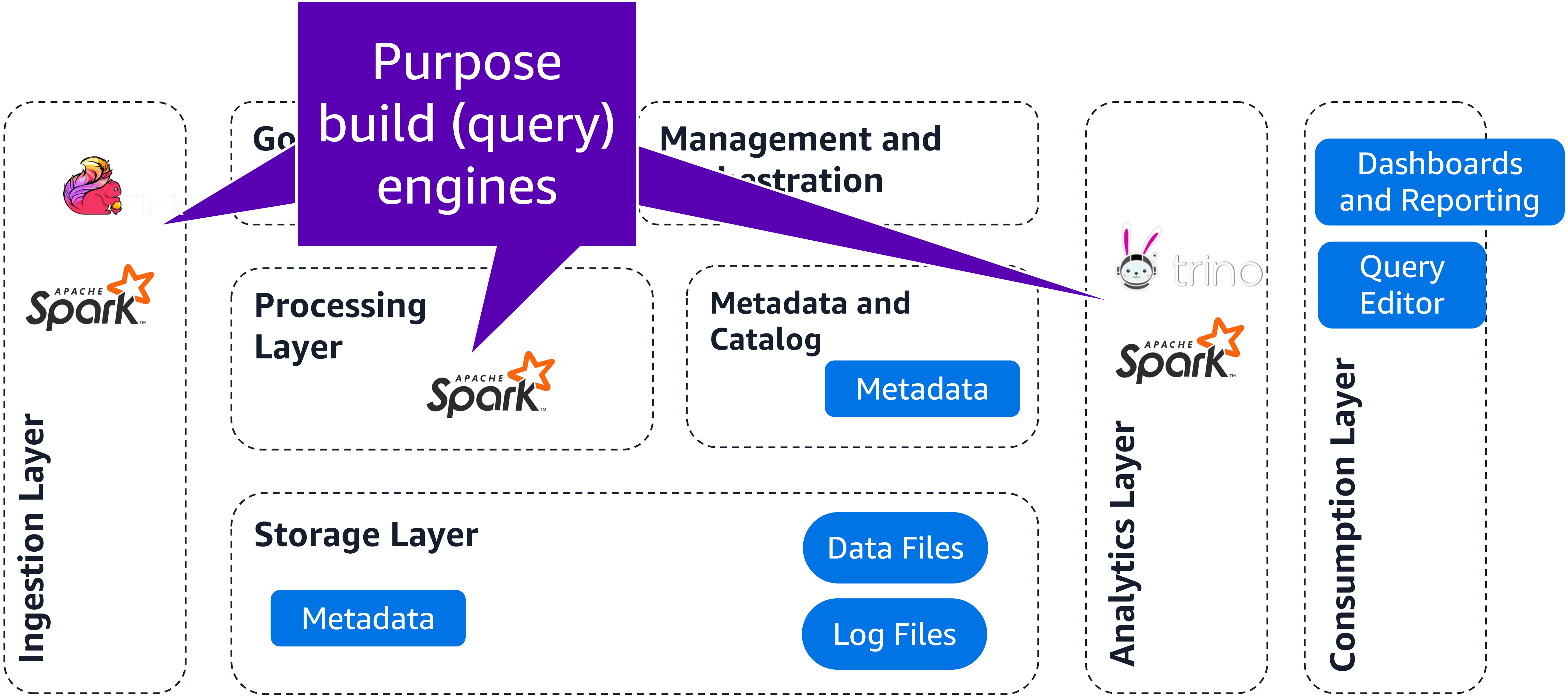
Turning the Data Warehouse inside out (1/2)



Turning the Data Warehouse inside out (2/2)



Powering the Transactional Data Lakes



Purpose build (query) engines

Ingestion Layer

Go

Management and
...estration

Processing Layer

Metadata and Catalog

Metadata

Storage Layer

Metadata

Data Files

Log Files

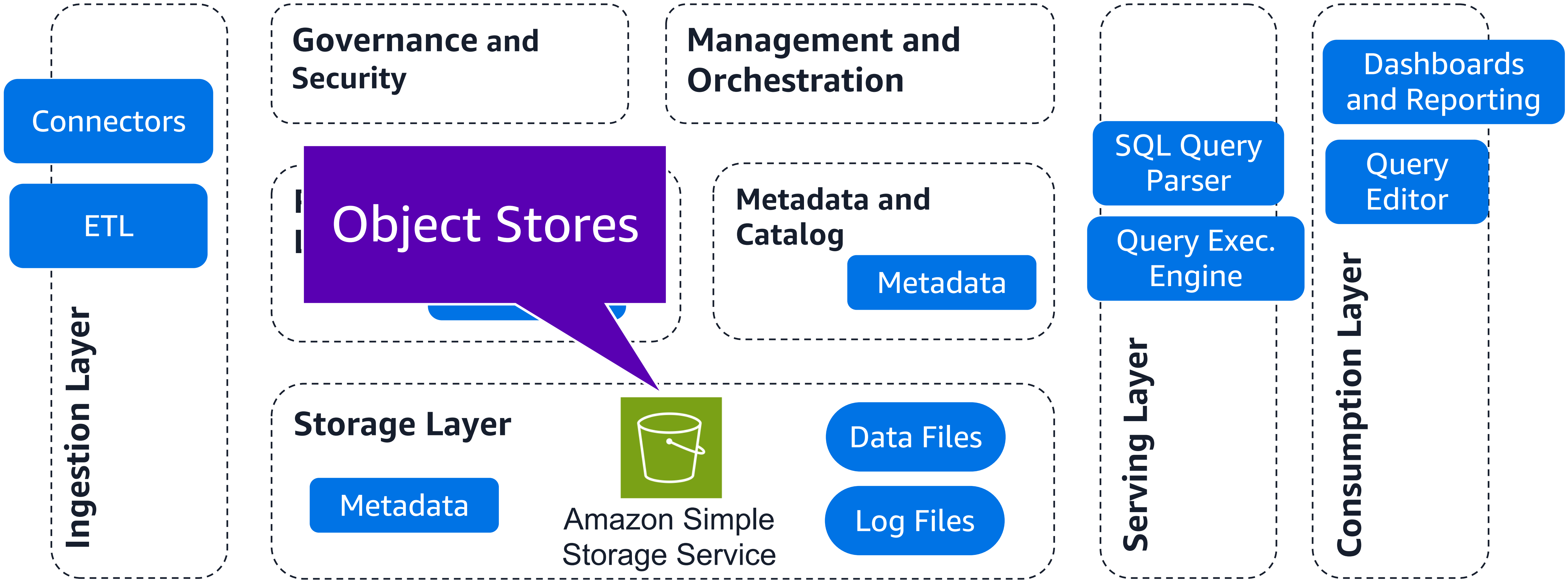
Analytics Layer

Consumption Layer

Dashboards and Reporting

Query Editor

Powering the Transactional Data Lake



Ingestion Layer

Connectors

ETL

Governance and Security

Management and Orchestration

Object Stores

Metadata and Catalog

Metadata

Storage Layer

Metadata

Amazon Simple Storage Service

Data Files

Log Files

Serving Layer

SQL Query Parser

Query Exec. Engine

Consumption Layer

Dashboards and Reporting

Query Editor

Powered by object stores

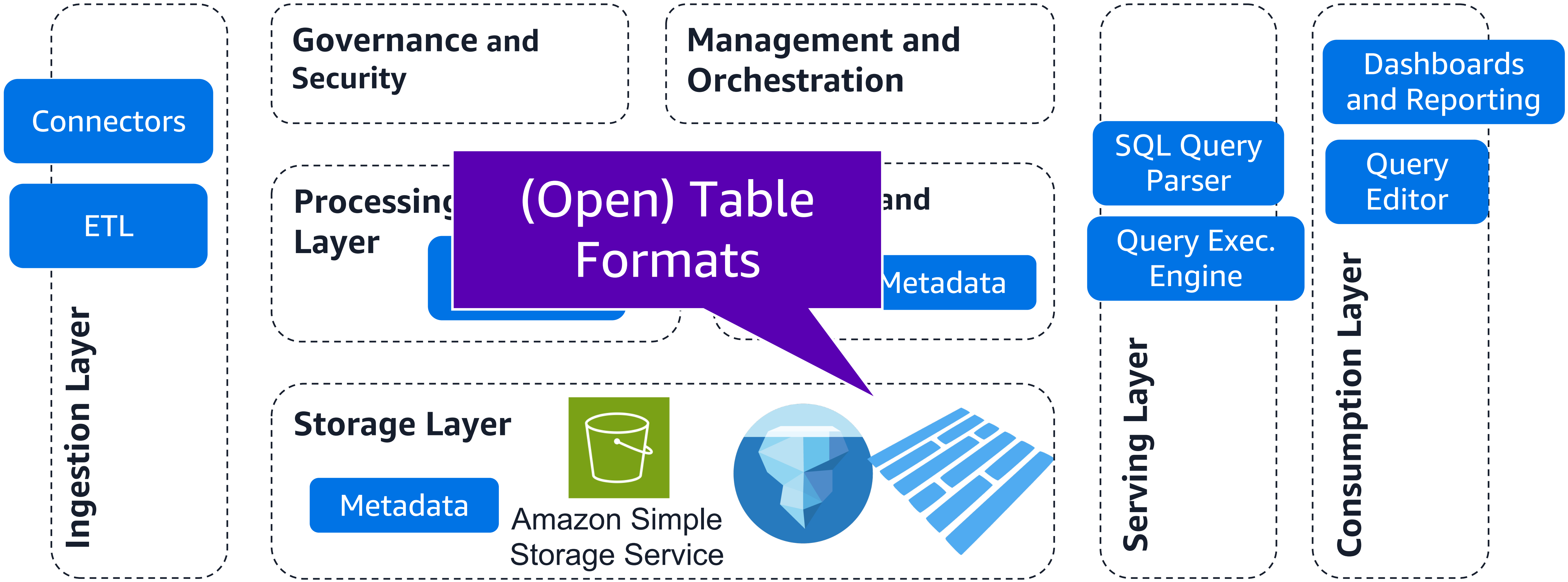
Amazon Simple Storage Service

infinite storage

11 nines of durability

99.99% availability

Powering the Transactional Data Lake



Ingestion Layer

Connectors

ETL

Governance and Security

Management and Orchestration

Processing Layer

and

Metadata

(Open) Table Formats

Storage Layer

Metadata

Amazon Simple Storage Service

Serving Layer

SQL Query Parser

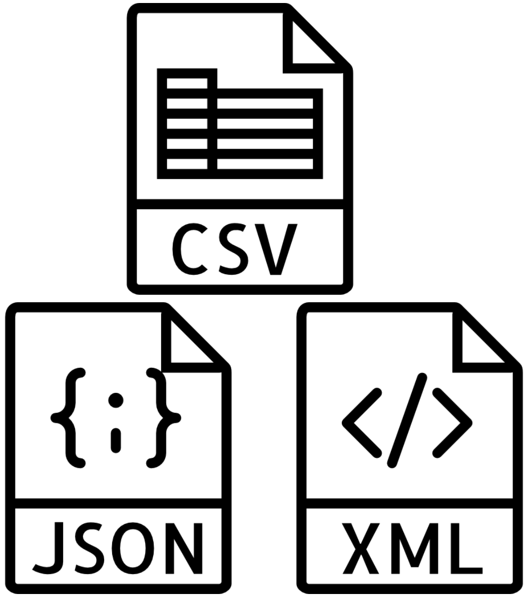
Query Exec. Engine

Consumption Layer

Dashboards and Reporting

Query Editor

Backed by a broad choice of file formats

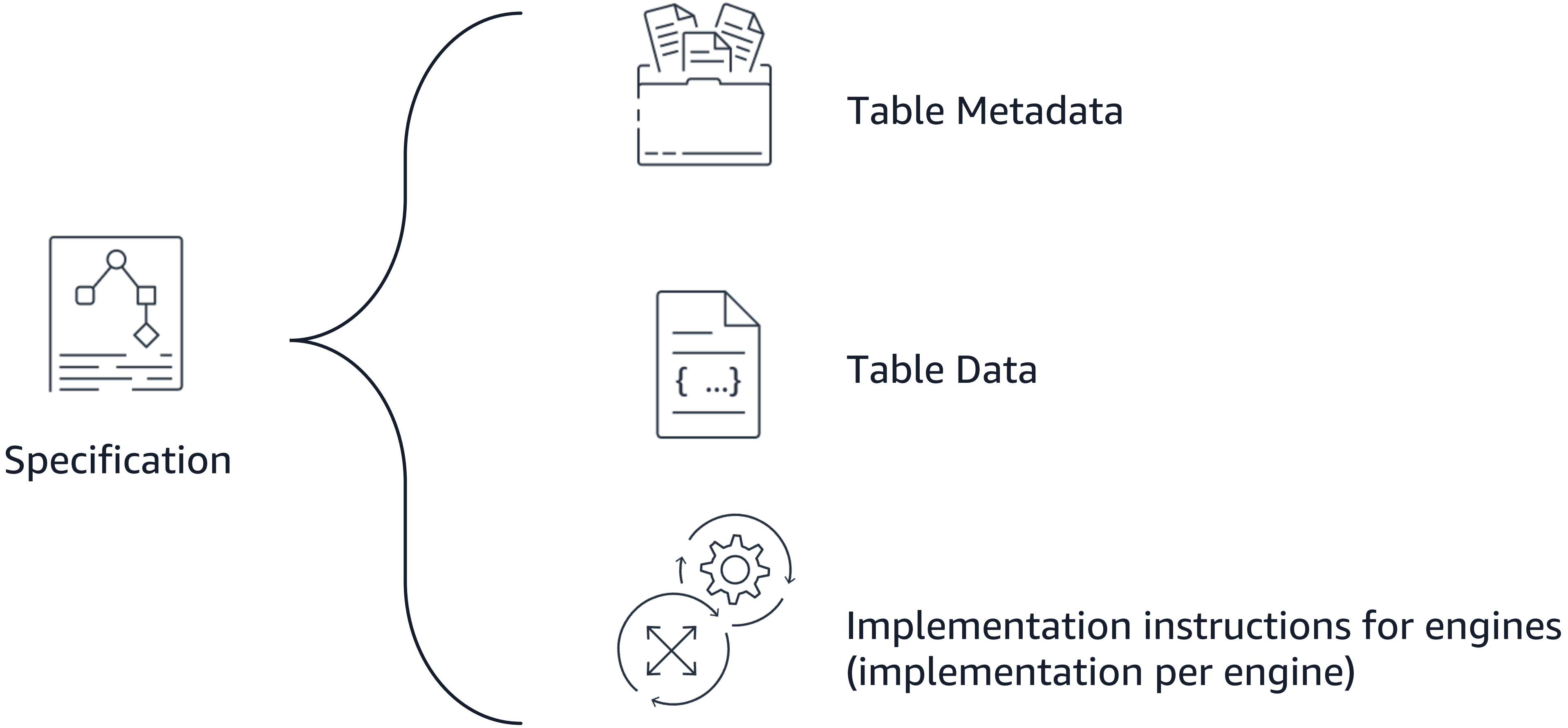


Common file formats

Optimized file formats

Table file formats

Open Table formats



Specification

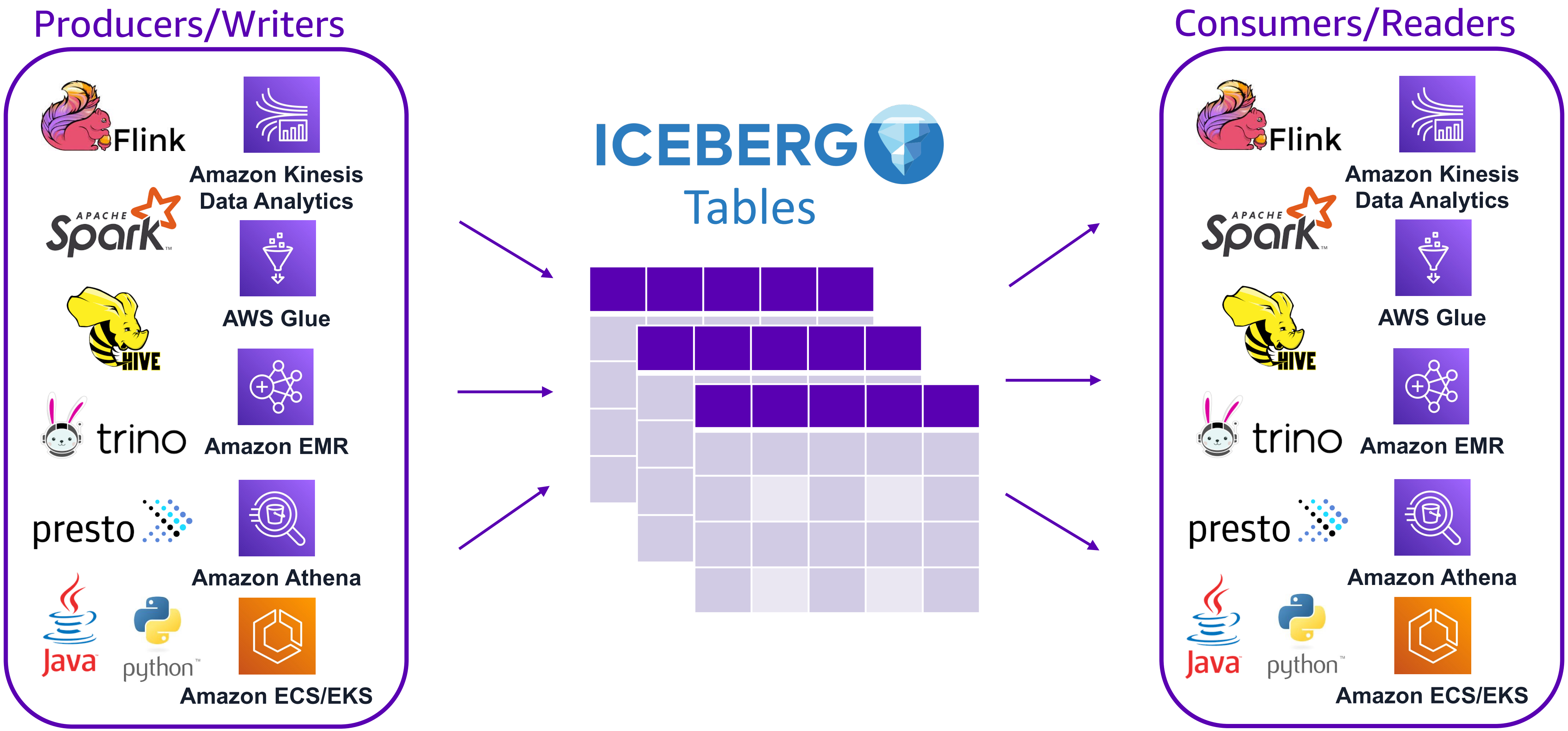
Table Metadata

Table Data

Implementation instructions for engines
(implementation per engine)

Introduction to Apache Iceberg

Iceberg Ecosystem



Iceberg Features

- **Delete, update, and upsert** support using copy-on-write or merge-on-read approaches
- **Fast scan planning & advanced filtering** using table metadata for with partition and column-level stats
- **Full schema evolution** supports add, drop, update, or rename, and has no side-effects
- **Partition layout evolution** can update the layout of a table as data volume or query patterns change
- **Time travel** enables reproducible queries that use exactly the same table snapshot, or lets users easily examine changes
- **Version rollback** allows users to quickly correct problems by resetting tables to a good state
- **Snapshot isolation** where table changes are atomic and readers never see partial or uncommitted changes
- **Multiple concurrent writers** use optimistic concurrency and will retry to ensure that compatible updates succeed, even when writes conflict
- …

Iceberg Table Format Overview

- Iceberg tables are composed by 3 layers:
- Iceberg Catalog
- Metadata Layer
- Data Layer

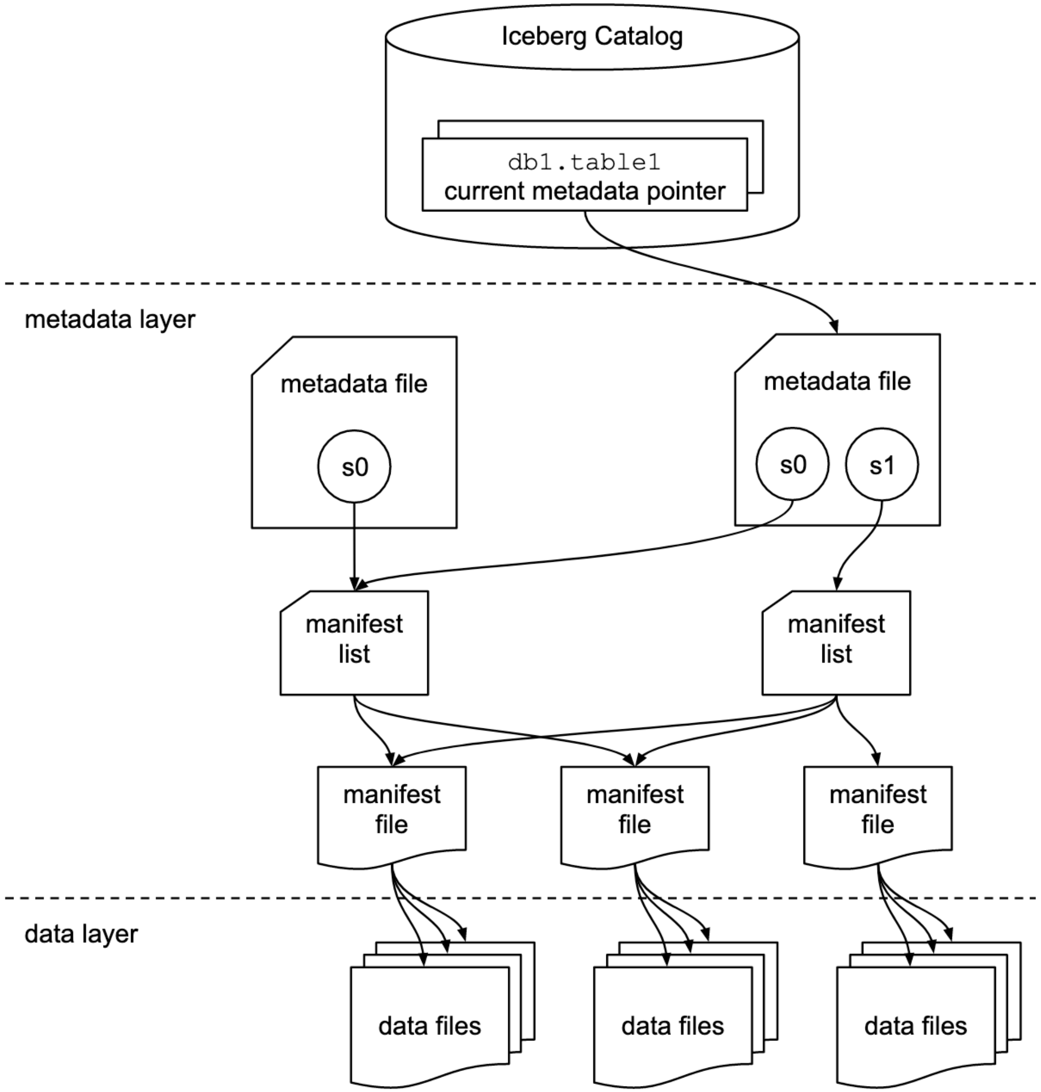


Table Layout - Catalog

- **Iceberg catalog** stores the metadata pointer for the current table metadata file and supports atomic updates

- **Metadata pointer** maps the table name to its current metadata file

- When a **SELECT query** is reading an Iceberg table, the query engine first goes to the Iceberg catalog, then retrieves the entry of the location of the current metadata file

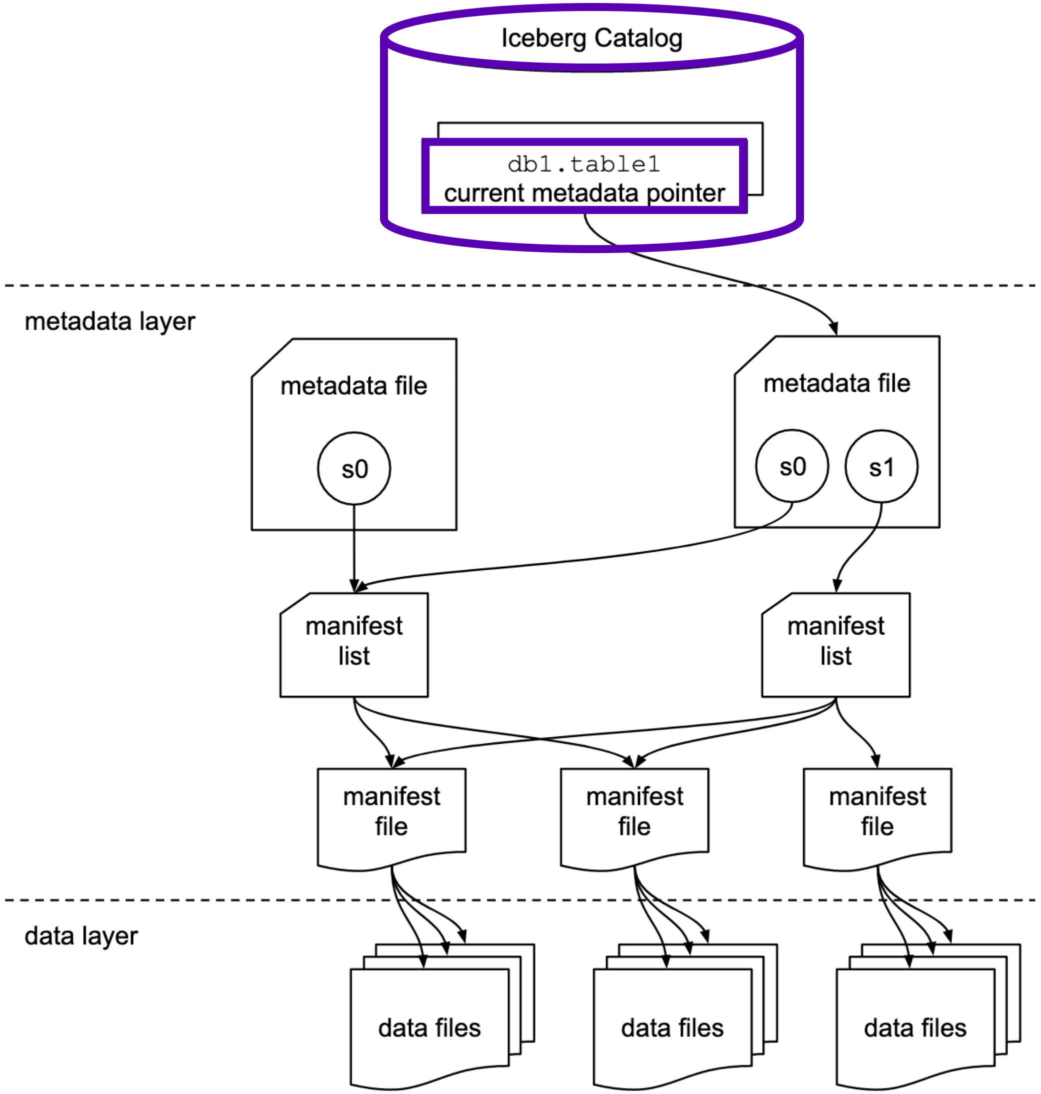


Table Layout - Metadata file

- **Metadata file** stores the table state. Any change creates a new metadata file replacing the old one with an **atomic swap**

- The table metadata file tracks **schema, partitioning**, **properties**, and **snapshots** of the table contents

- A **snapshot** represents the state of a table at some time and is used to access table's data files

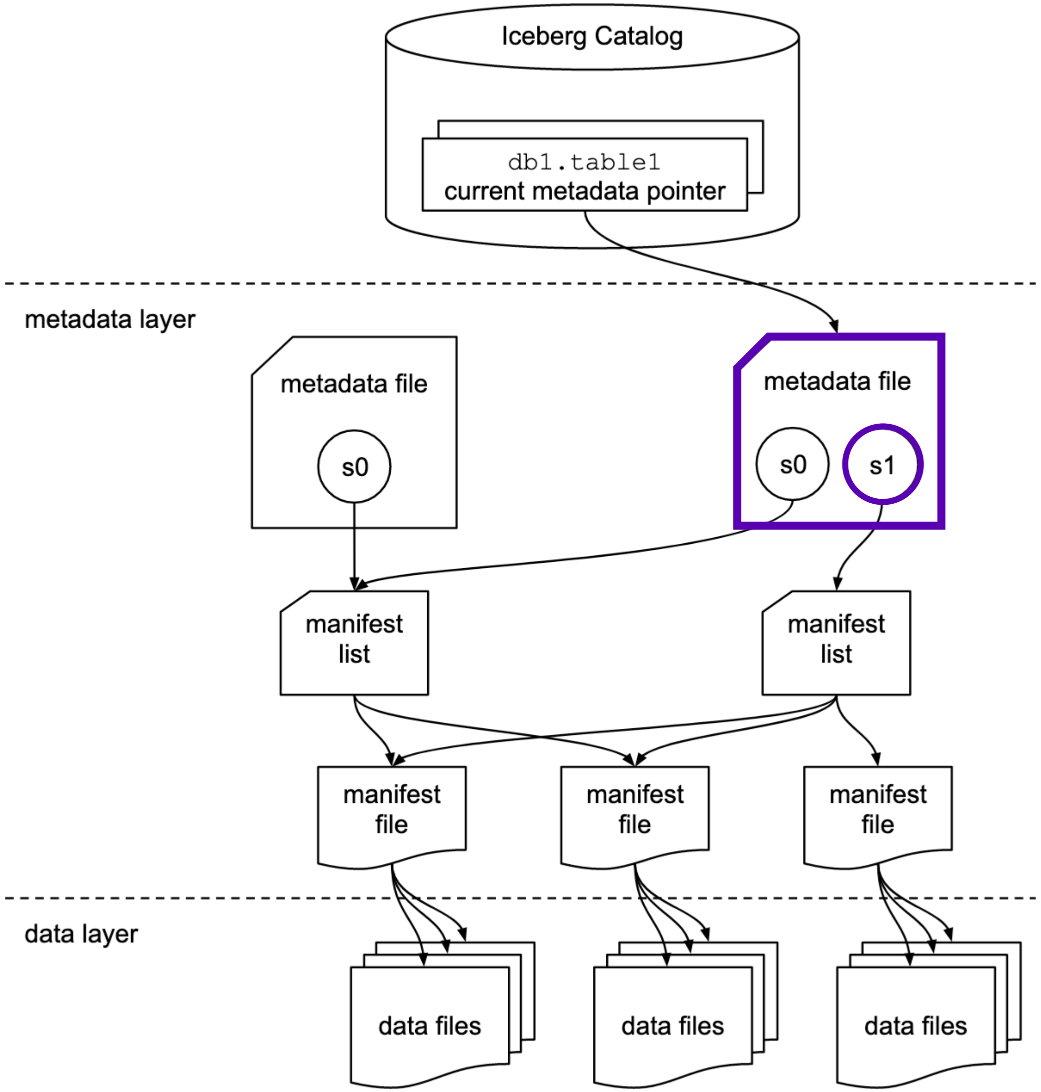
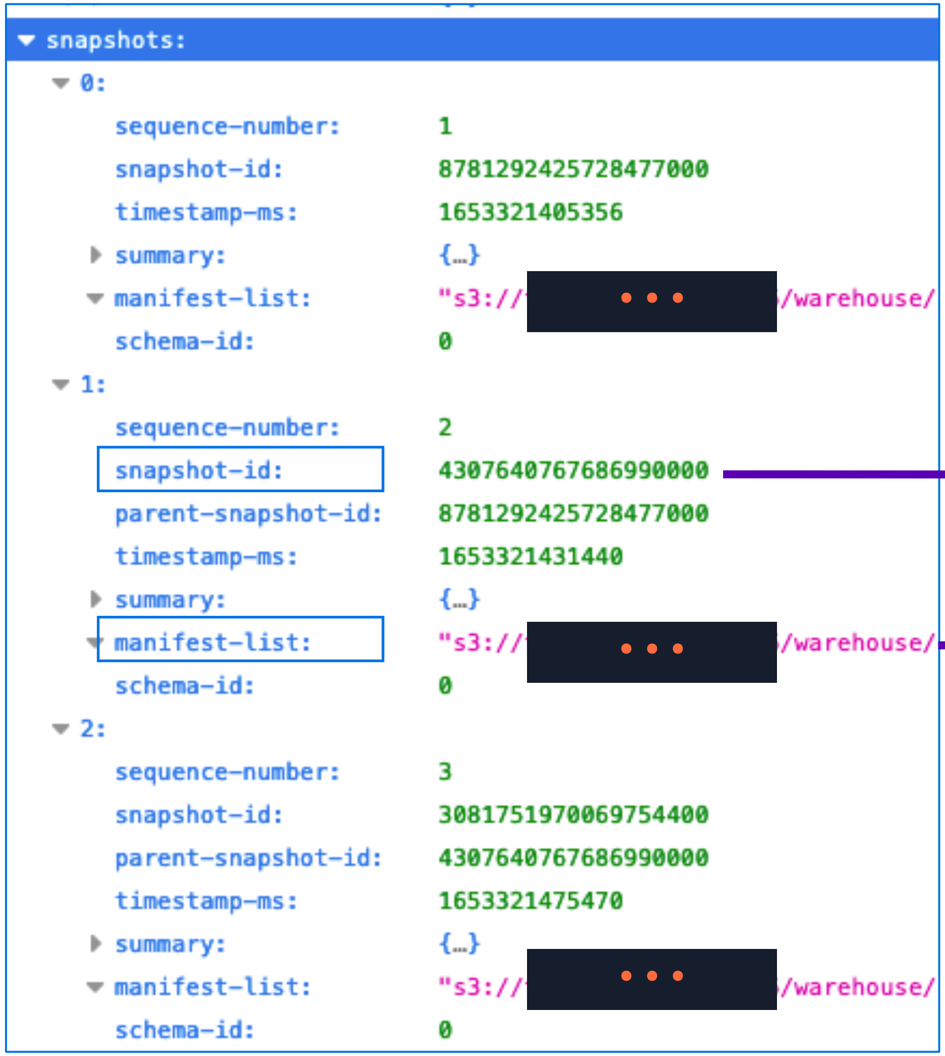


Table Layout - Metadata file



```
▼ snapshots:
  ▼ 0:
      sequence-number:       1
      snapshot-id:           8781292425728477000
      timestamp-ms:          1653321405356
    ▶ summary:               {…}
    ▼ manifest-list:         "s3://[redacted]/warehouse/
      schema-id:             0
  ▼ 1:
      sequence-number:       2
      snapshot-id:           4307640767686990000
      parent-snapshot-id:    8781292425728477000
      timestamp-ms:          1653321431440
    ▶ summary:               {…}
    ▼ manifest-list:         "s3://[redacted]/warehouse/
      schema-id:             0
  ▼ 2:
      sequence-number:       3
      snapshot-id:           3081751970069754400
      parent-snapshot-id:    4307640767686990000
      timestamp-ms:          1653321475470
    ▶ summary:               {…}
    ▼ manifest-list:         "s3://[redacted]/warehouse/
      schema-id:             0
```

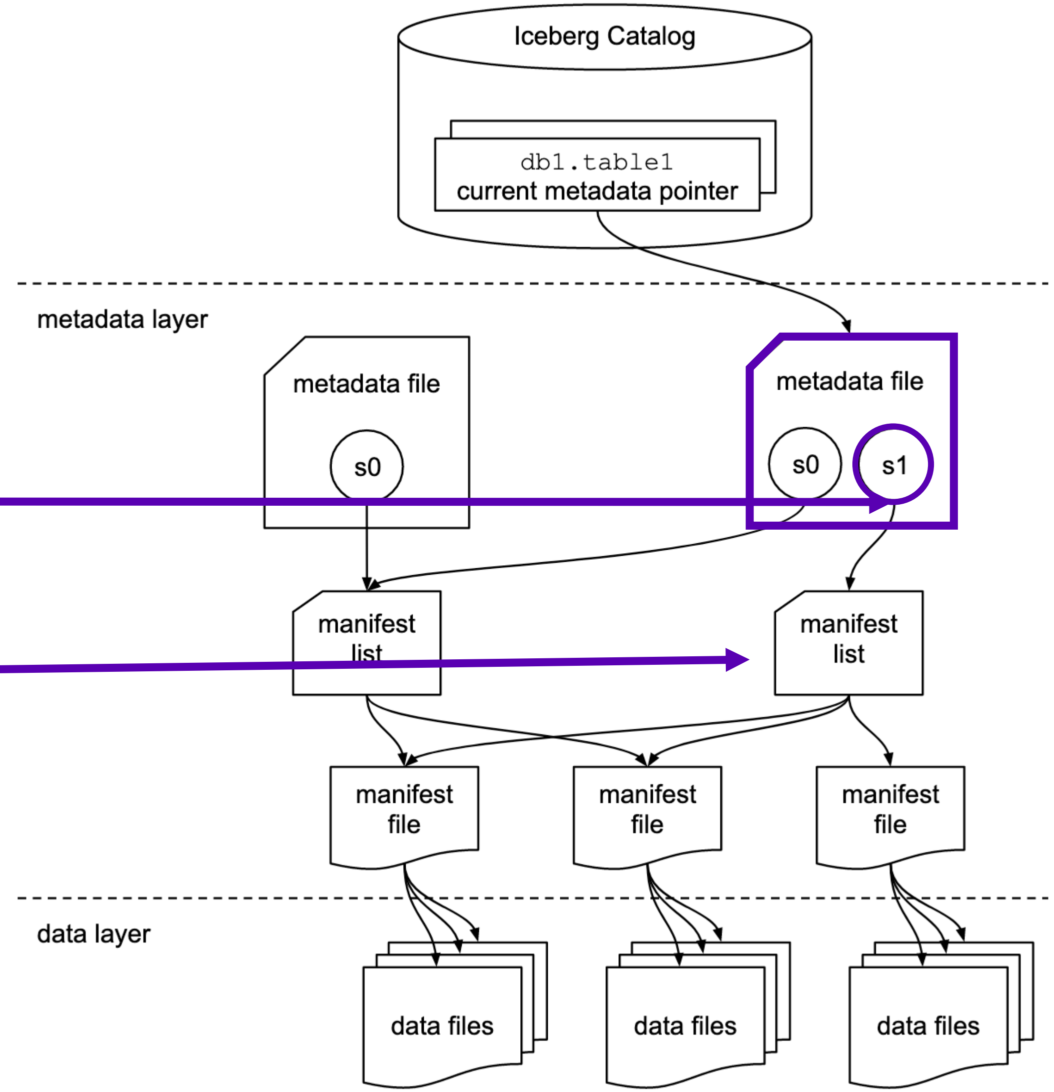


Table Layout - Manifest list file

```
{
    "manifest_path" : "s3://.../...6e8107509-m0.avro",
     "added_snapshot_id" : 8781292425728477000,
     (...)
},

{
    "manifest_path" : "s3://.../...6678f597-m0.avro",
    "added_snapshot_id" : 3081751970069754400,
    (...)
},

{
    "manifest_path" : "s3://.../.... ejebe38-m0 avro",
     "added_snapshot_id" : S1,
     (...)
},

{
    "manifest_path" : "s3://.../...c10be38-m1.avro",
    "added_snapshot_id" : S1,
    (...)
},
...
```

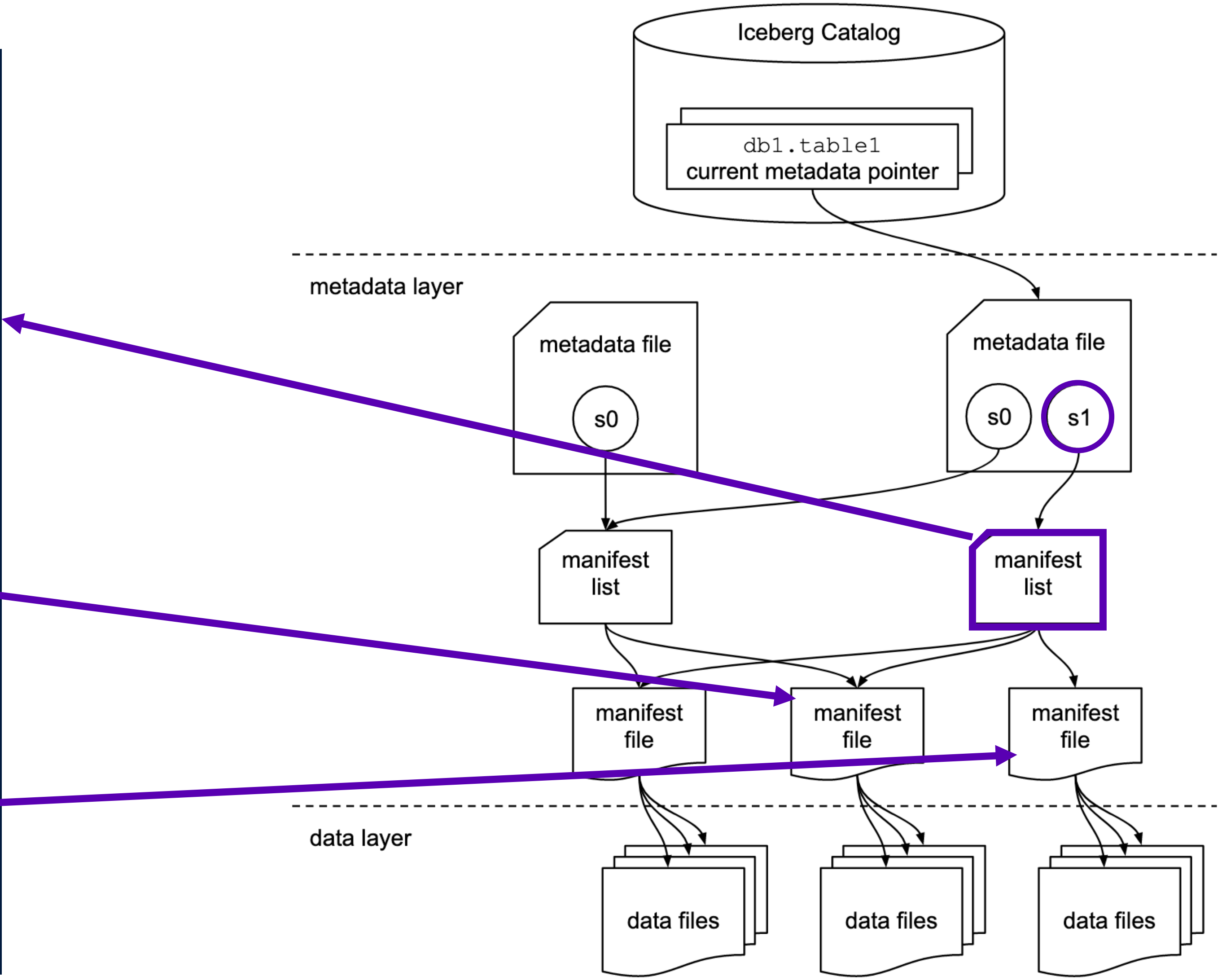


Table Maintenance

Recommended Maintenance

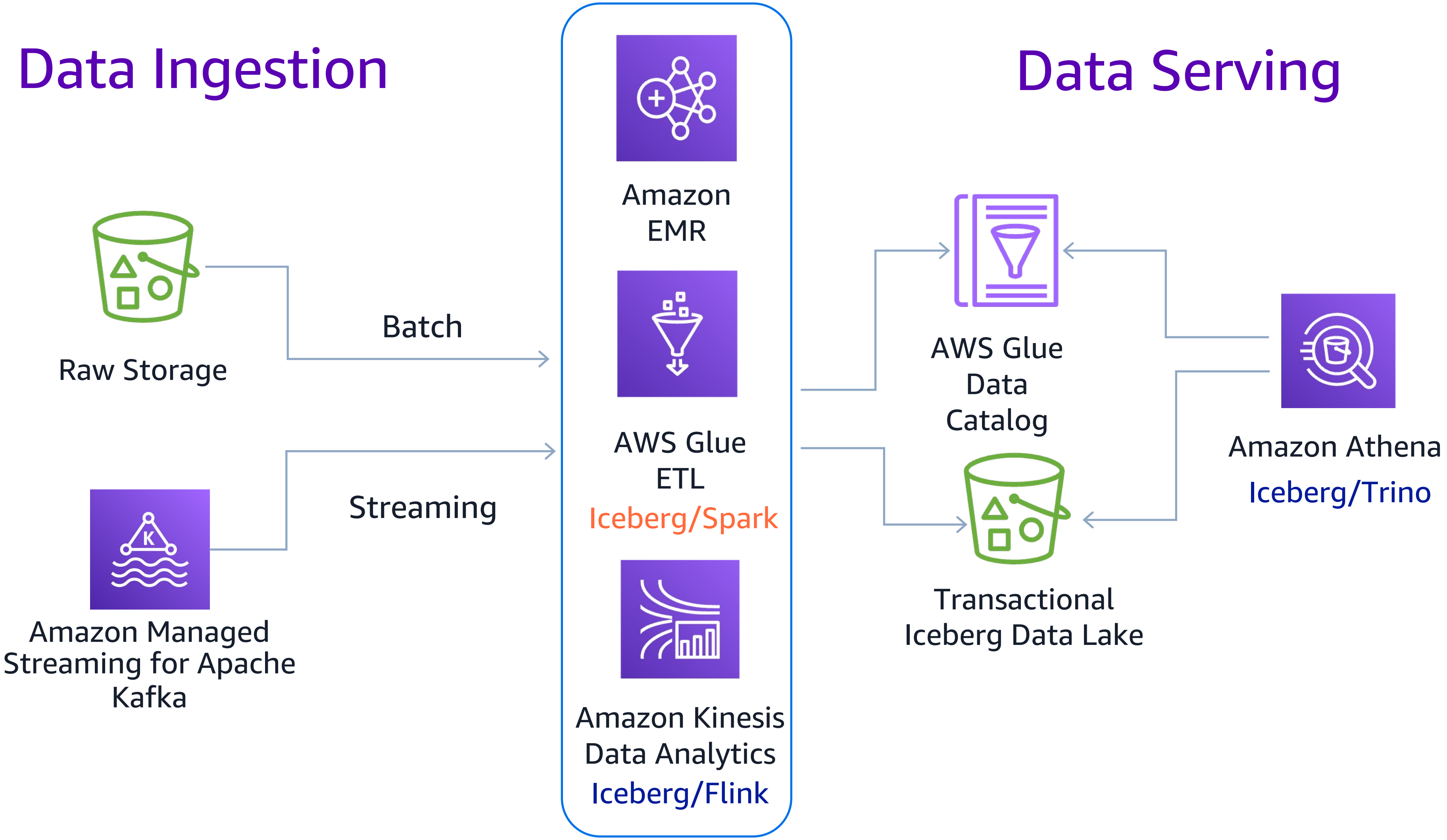
- Expire Snapshots
- Remove old metadata files
- Delete orphan files

Optional Maintenance

- Compaction of data files
- Compaction of manifest files

Transactional Data Lake on AWS

Transactional Data Lake reference architecture



Recap – What have we learned

Transactional Data Lakes combine the best of Data Lakes and Data Warehouses into one system, whilst avoiding their downsides.

Open Table Formats bring back ACID (Atomicity, Consistency, Isolation, Durability) transactions and enable advanced query optimization through extended metadata.

Apache Iceberg is an Open Table Format, widely supported inside and outside of AWS, keeping you in control of your data and giving you choice selecting the right engine for the job.

Thank you!

Norman Weisenburger

norwei@amazon.de

/in/normalerweise

Feedback

https://pulse.aws/survey/ARYOGOIM