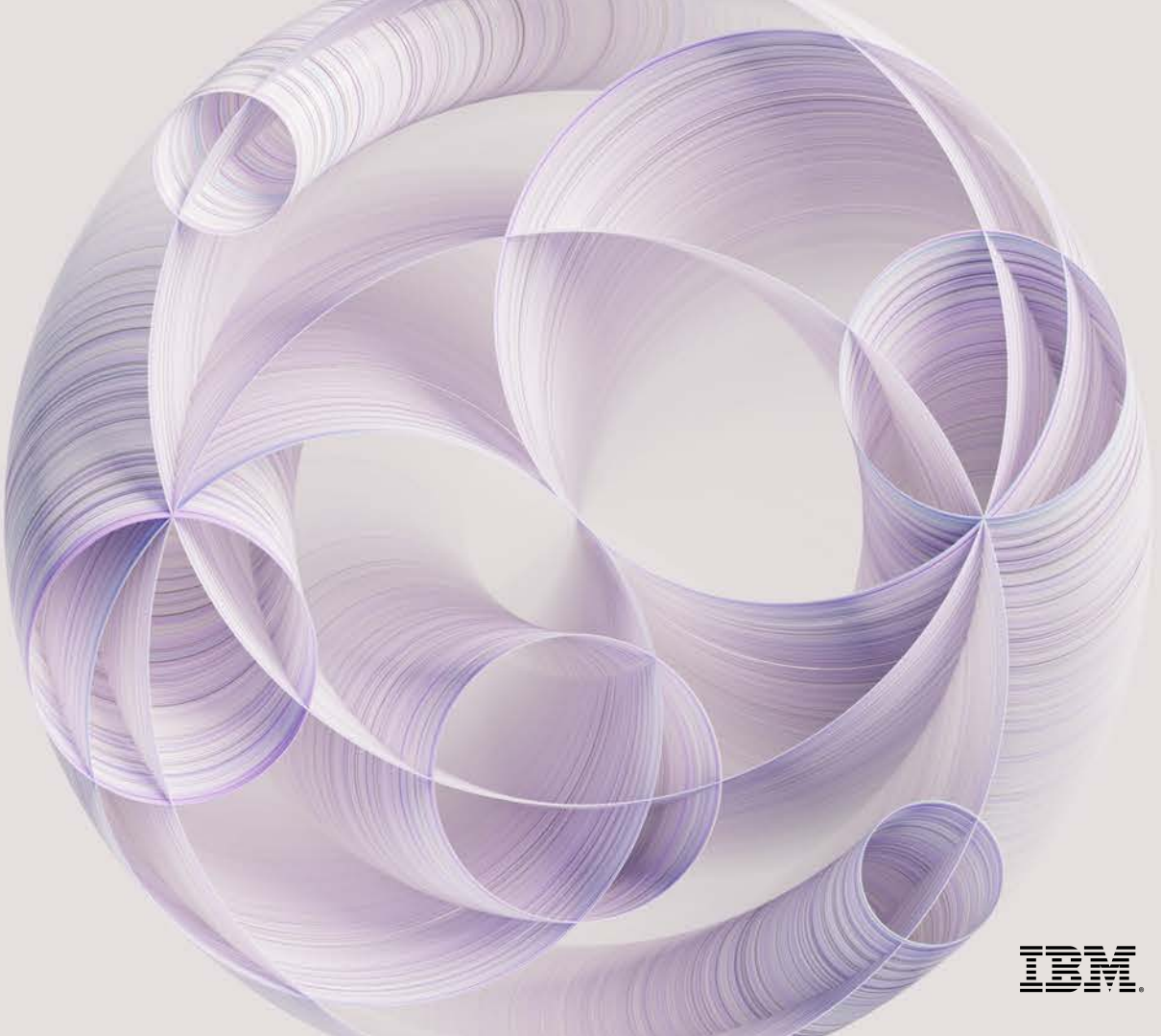
Watsonx.data

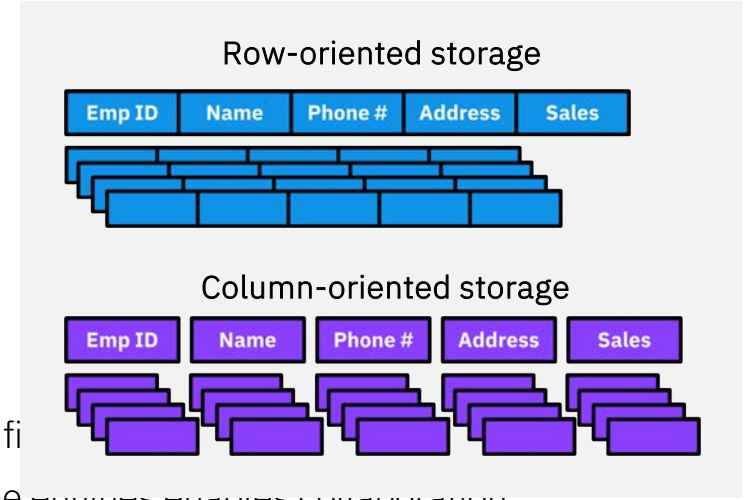
Day 2
Beyond
Fundamentals



Apache Parquet

Parquet is designed to support fast data processing for complex data

- Open-source
- Columnar storage
- Highly compressible with configurable compression options and extendable encoding schemas by data type
- Self-describing: schema and structure metadata is included
- Schema evolution with support for automatic schema merging

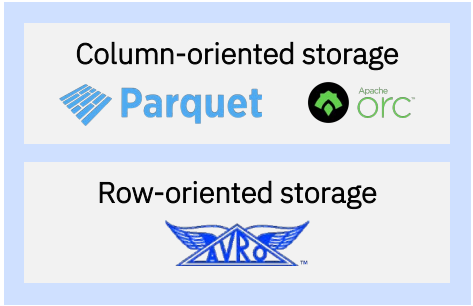


Why do these things matter in a lakehouse?

- Performance of queries directly impacted by size and amount of fi
- Ability to read/write data to an open format from multiple runtime engines enables collaboration
- Size of data stored, amount of data scanned, and amount of data transported affect the charges incurred in using a lakehouse (depending on the pricing model)

Apache ORC

- Open-source, **columnar storage** format
  - Similar in concept to Parquet, but different design
  - Parquet considered to be more widely used than ORC
- Highly compressible, with multiple compression options
  - Considered to have higher compression rates than Parquet
- Self-describing and type-aware
- Support for schema evolution
- Built-in indexes to enable skipping of data not relevant to a query
- Excellent performance for read-heavy workloads
  - ORC generally better for workloads involving frequent updates or appends
  - Parquet generally better for write-once, read-many analytics



Apache Avro

- Open-source, **row-based** storage and serialization format
  - Can be used for file storage or message passing
- Beneficial for write-intensive workloads
- Format contains a mix of text and binary
  - Data definition: Text-based JSON
  - Data blocks: Binary
- Robust support for schema evolution
  - Handles missing/added/changed fields
- Language-neutral data serialization
  - APIs included for Java, Python, Ruby, C, C++, and more

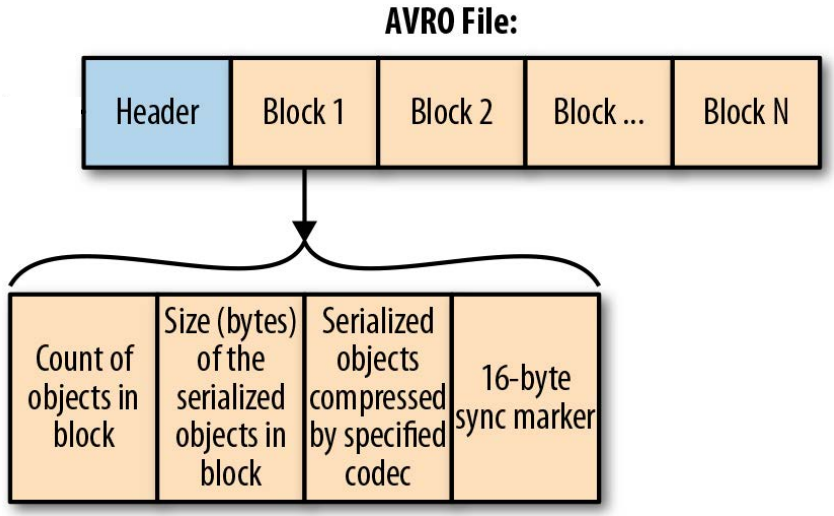


Table management and formats

Sits “above” the data file layer

Organizes and manages table metadata and data

Typically supports multiple underlying disk file formats (Parquet, Avro, ORC, etc.)

May offer transactional concurrency, I/U/D, indexing, time-based queries, and other capabilities

- Open-source
- Designed for large, petabyte (PB)-scale tables
- ACID-compliant transaction support
- Capabilities not traditionally available with other table formats, including schema evolution, partition evolution, and table version rollback – all without re-writing data
- Advanced data filtering
- Time-travel queries let you see data at points in the past

- Open-source
- Manages the storage of large datasets on HDFS and cloud object storage
- Includes support for tables, ACID transactions, upserts/ deletes, advanced indexes, streaming ingestion services, concurrency, data clustering, and asynchronous compaction
- Multiple query options: snapshot, incremental, and read-optimized

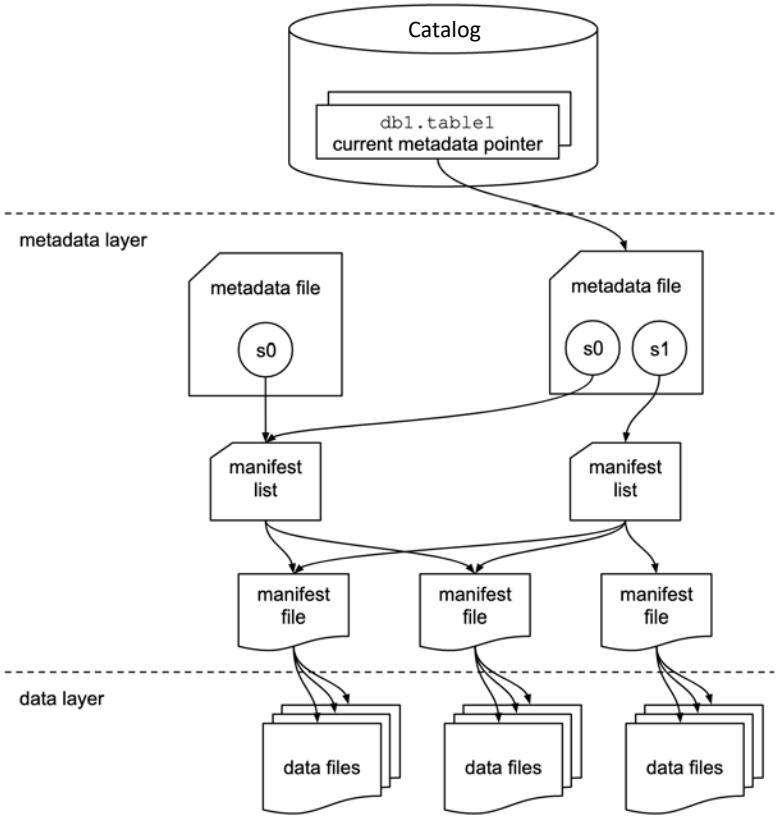
- Open-source, but Databricks is primary contributor and user, and controls all commits to the project – so “closed”
- Foundation for storing data in the Databricks Lakehouse Platform
- Extends Parquet data files with a file-based transaction log for ACID transactions and scalable metadata handling
- Capabilities include indexing, data skipping, compression, caching, and time-travel queries
- Designed to handle batch as well as streaming data

Why Apache Iceberg for data lakehouses?

Open-source data table format that helps simplify data processing on large dataset stored in data lakes

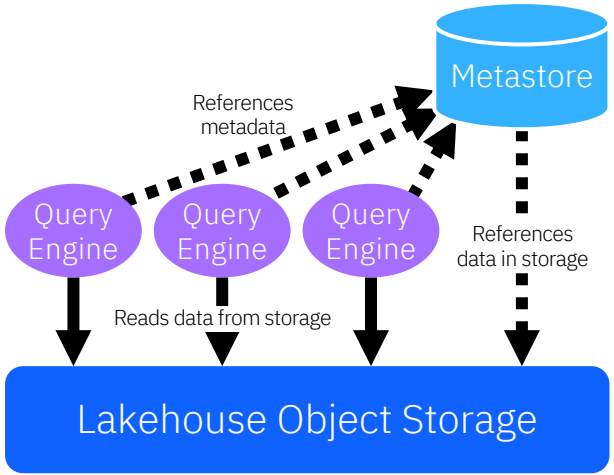
People love it because it has:

- SQL — Use it to build the data lake and perform most operations without learning a new language
- Data Consistency — ACID compliance (not just append data operations to tables)
- Schema Evolution — Add/remove columns without distributing underlying table structure
- Data Versioning — Time travel support that lets you analyze data changes between update and deletes
- Cross Platform Support — Supports variety of storage systems and query engines (Spark, Presto, Hive, +++)



Hive Metastore (HMS)

- Open-source **Apache Hive** was built to provide an SQL-like query interface for data stored in Hadoop
- **Hive Metastore (HMS)** is a component of Hive that stores metadata for tables, including schema and location
- HMS can be deployed standalone, without the rest of Hive (often needed for lakehouses, like watsonx.data)
- Query engines use the metadata in HMS to optimize query execution plans
- The metadata is stored in a traditional relational database (PostgreSQL in the case of watsonx.data)
- In watsonx.data, Watson Knowledge Catalog integrates with HMS to provide policy-based access and governance



Presto

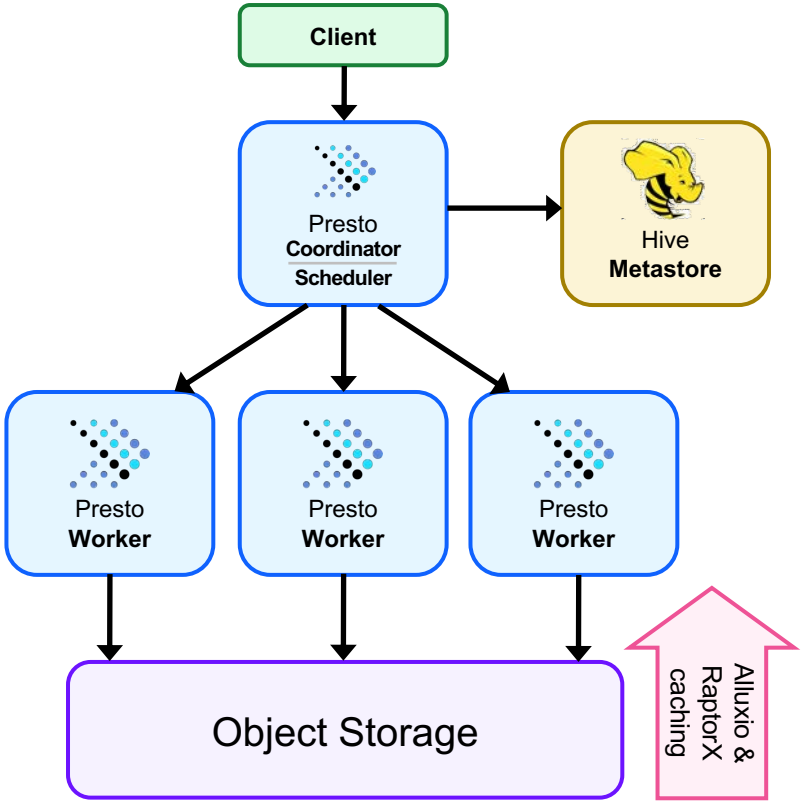
- Presto is an open-source distributed SQL engine suitable for querying large amounts of data
- Supports both relational and non-relational sources
- Easy to use with data analytics and business intelligence tools
- Supports both interactive and batch workloads
- In watsonx.data, spin up one or more Presto compute engines of various sizes – cost effective, in that engines are ephemeral and can be spun up and shut down as needed

- Presto connectors allow access to data in-place, allowing for **no-copy data access and federated querying**
- Consumers are abstracted from the physical location of data
- A wide variety of data sources are supported, including:

Presto architecture

The structure of Presto is similar to that of classical MPP database management systems.

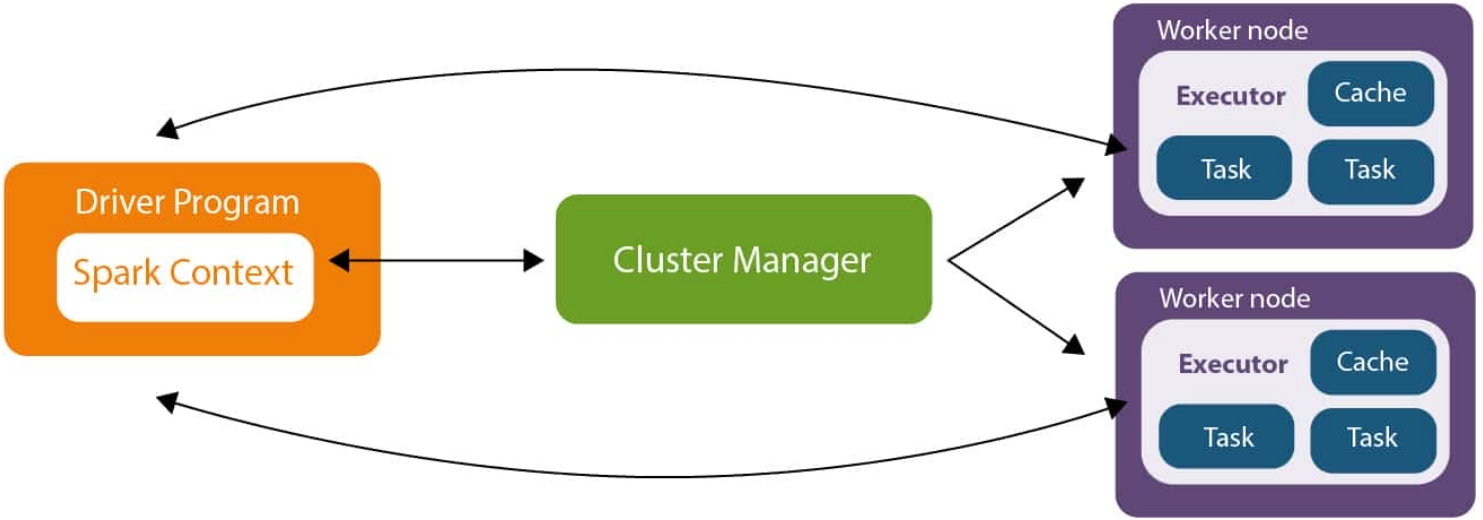
- **Client:** Issues user query and receives final result.
- **Coordinator:** Parses statement, plans query execution, and manages worker nodes. Gets results from workers and returns final result to client.
- **Workers:** Execute tasks and process data.
- **Connectors:** Integrate Presto with external data sources like object stores, relational databases, or Hive.
- **Caching:** Accelerated query execution through metadata and data caching (provided by Alluxio and RaptorX).



Apache Spark

Apache Spark is an open-source data-processing engine for large data sets. It is designed to deliver the computational speed, scalability, and programmability required for *big data*, specifically for streaming data, graph data, ML, and AI applications.

The basic Apache Spark architecture diagram:



Apache Spark and machine learning

Spark has libraries that extend the capabilities to ML, AI, and stream processing.

- Apache Spark MLlib
- Spark Streaming
- Spark SQL
- Spark GraphX