watsonx.data™

watsonx.data™

Agenda

- Introduction to WatsonX
- watsonx.data
- Architecture,
- Value proposition
- Demonstration of watsonx.data and related use cases (retail/ Mortgage)
- Use Cases, Competitive Insights , Roadmap
- Sales calculator overview and how to size, licensing details
- Environment set up for Hands on labs
- Watsonx.data hands on lab - Watsonx.data for Retail
- Training recap followed by quiz

Agenda

- What is this watsonx.data?
- Architecture
- Deployment options
- Interface/tools
- TechZone watsonx.data Lab

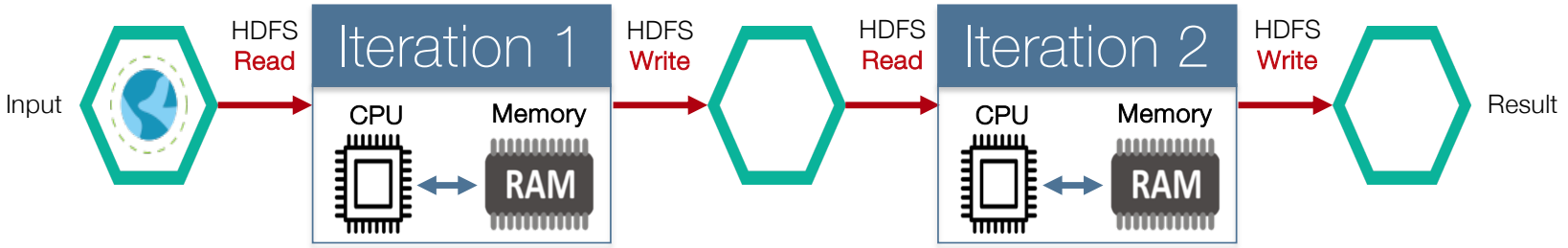
Challenges of Today's Monolithic Architecture

Key Data Terms

Data Warehouse	Centralized repository that stores large volumes of structured, semi-structured, and unstructured data from various sources within an organization. It is designed to support business intelligence (BI) activities, including reporting, analysis, and decision-making.
Data Mart	Subset or a specialized version of a data warehouse that focuses on a specific department, function, or subject area within an organization. It contains a curated and tailored collection of data that is relevant to a particular group of users.
Data Lake	Storage architecture that allows organizations to store large volumes of structured, semi-structured, and unstructured data in its raw format. It is a central repository that holds diverse data types and formats without the need for upfront data transformation or schema definition.
Data Lakehouse	Data lakehouse are designed to be one place for all workloads, providing support for reporting, data science, AI, and machine learning on the same data, at the same time, all in one place.
Data Pipeline	Series of processes and steps that extract, transform, and load (ETL) data from various sources into a target destination for storage, analysis, or consumption. It is a structured flow of data that ensures the reliable and efficient movement of data throughout the stages of the pipeline.
Data Mesh	Conceptual architectural approach that aims to address the challenges of data management and scalability in modern organizations. It suggests a decentralized and domain-oriented approach to data, where data ownership and governance are distributed across different teams or domains within an organization.

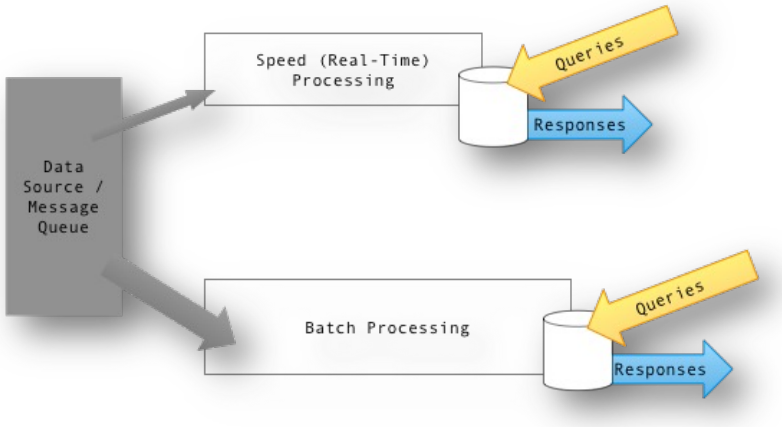
Some context – Big Data, MapReduce challenges

With the advent of Big Data a few years ago, we saw the rise of Hadoop and commodity hardware as a means to collect and analyze large volumes of diverse data in a cost-effective manner. Map Reduce was the programming model within the Hadoop framework used to access big data stored in the Hadoop File System(HDFS). The Map & Reduce functions operate in concert to accomplish the task, the intermediate results for these functions are written to disk and read from disk.



Great, but what about getting the most accurate near-real time data ?

Lambda Architecture



Batch layer

- Use distributed processing system to pre-compute results.
- There is always a lag in data.

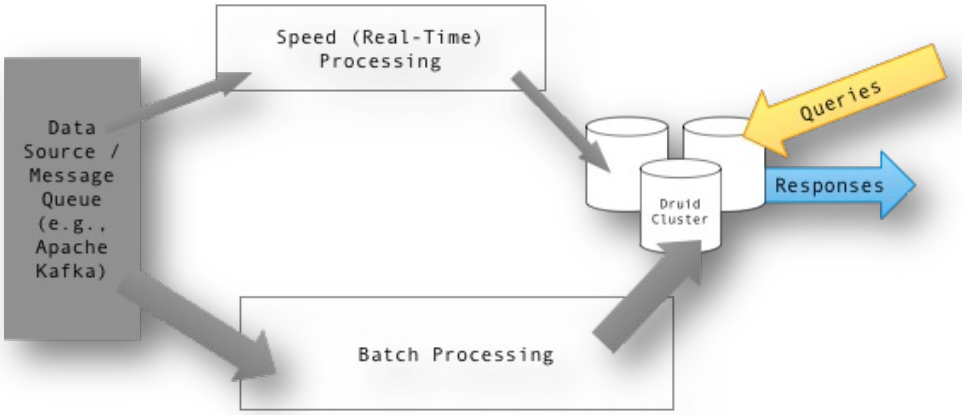
Speed layer

- Uses streaming mechanisms to process data streams.
- Fills the gap in the batch layer's lag.

Serving layer

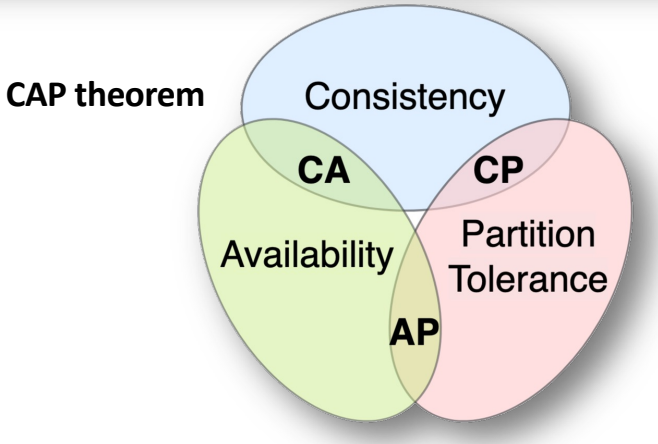
- Unifies batch and speed layers to address adhoc queries
- Inherent **complexities** to address delays in streams and reconciliation of data.

Why: mitigate the latencies of map-reduce

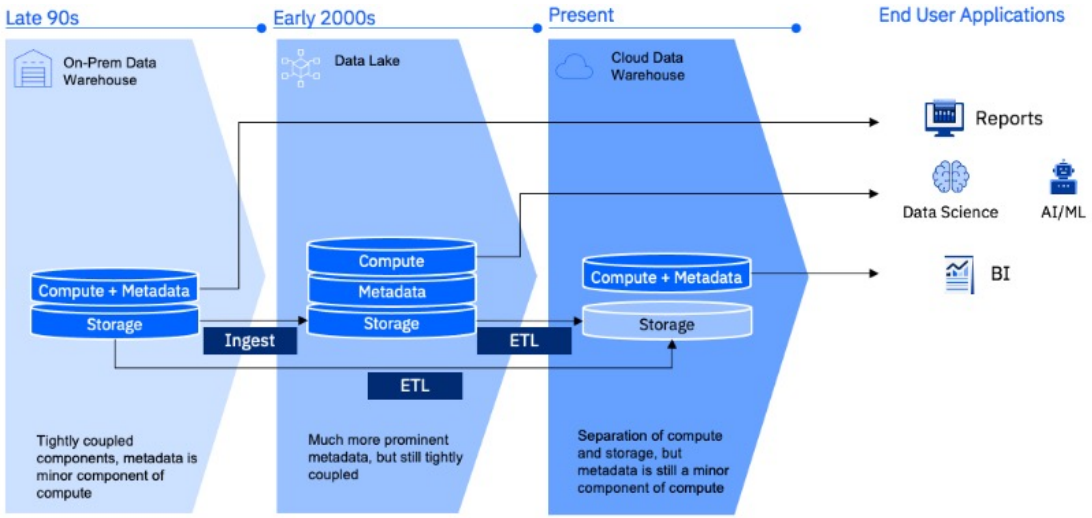


Images courtesy : Wikipedia

Problem & Opportunity !



Typical Data Management Architecture – Multi Tier



- This architecture is dominant for most customers. An on-premises data warehouse, will flow data to and from the Data Lake which acts as the raw data store. Then smaller independent Data Marts or Cloud Data Warehouses will be used to do the real work.
- Some of the operational reporting might still point to the EDW or ODS. Data science and AI are traditionally done on the Data Lake and the independent data marts are usually intended for analytics. The result is that this is a very segregated or siloed system that requires a customer to move data or copy data multiple times.
- In a traditional data warehouse, you had tight coupling of compute and storage. Metadata was a minor component which one rarely interacted with because it was an inherent part of the engine.
- In a Data Lake, metadata became more prominent, but compute, metadata and storage continued to be very tightly coupled.
- In the Cloud Data Warehouse, compute and storage were separated, but metadata was still not that prominent.

Why Lakehouse

Current Challenges and Opportunity

Data Warehouse Challenges

- Proprietary data formats
- Vendor lock in
- SQL-centric
- Less flexible
- Elasticity scale limitations
- Expensive
- Loosing next gen applications

Hadoop* Data Lake Challenges

- Poor in place Performance
- ACID/DML lacking
- Failure to address real time requirements
- Rigid pipelines – unable to handle evolution
- Narrow user focus mainly Data Science & ML
- Expensive to expand to generic BI and Analytical use cases
- High Skill to maintain and operate

* bulk of the market w.r.t Data Gravity

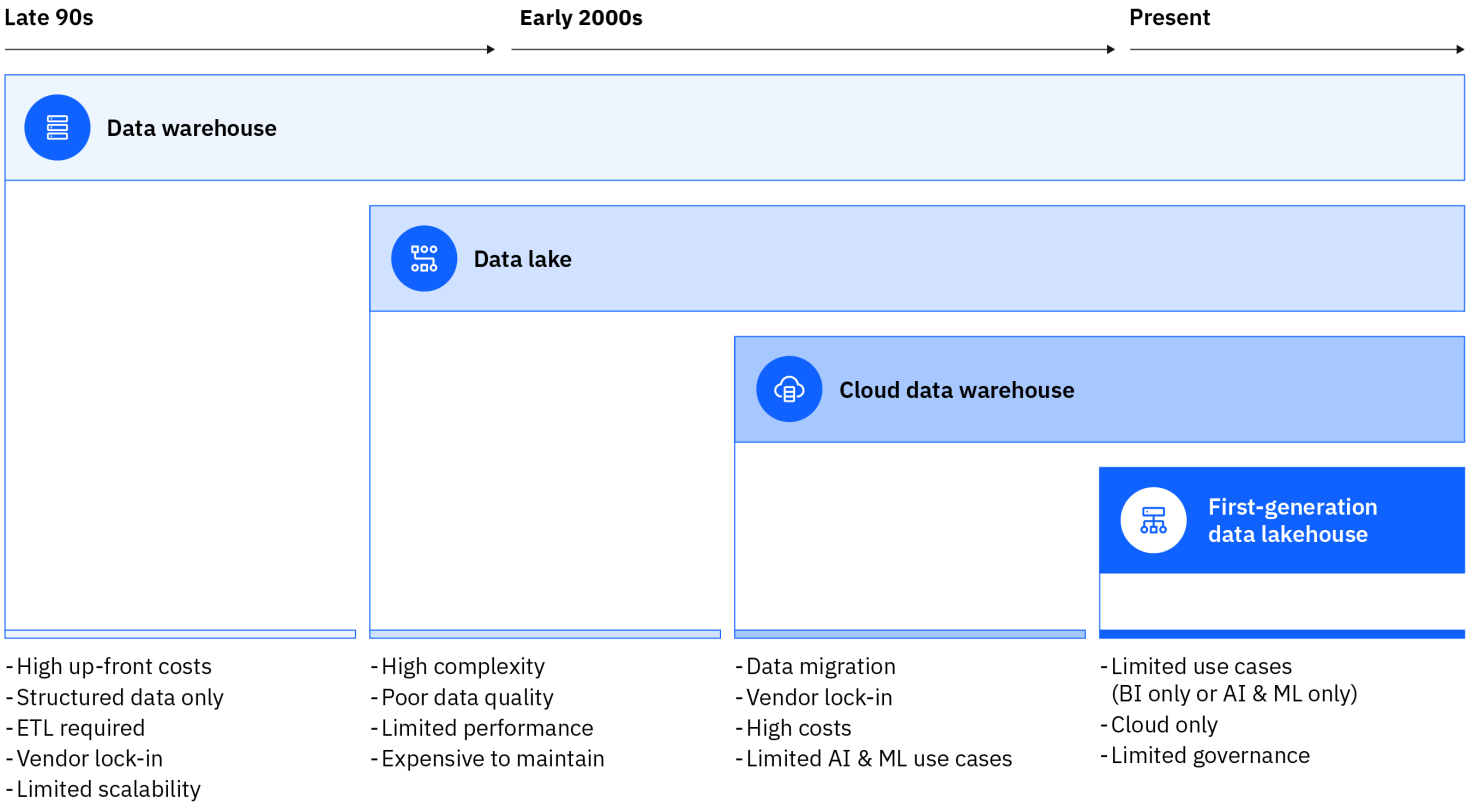
Data *Lake*
+
Ware*house*

Lakehouse

- ACID-compliancy - ensure consistency for multiple engines to concurrently read/write data
- Compute Storage separation (and low cost, highly elastic storage anywhere)
- Data Versioning
- Regulatory compliance
- In-built Governance with policy enforcement
- Open Data for applications – shared across vendors and technology

– Key use cases to support:
  - OLAP
  - ETL
  - ML

Traditional approaches to addressing these challenges have created more overall complexity and cost, which has led to the emergence of data lakehouse architectures



Today, leaders at most large enterprises manage their data and workloads using a mix of data repositories and data stores in hybrid environments.

The overall cost across all these repositories remains high.

It's difficult for leaders to effectively leverage and govern the data across multiple environments and use enterprise data for analytics and AI.

Enterprise leaders require a data architecture that can provide quick access to data, centralized governance and fit-for-purpose use.

1

Ability to scale AI while supporting compliance with lineage and reproducibility of data

---

2

Real-time analytics and BI that can connect to existing data in minutes without expensive duplicating or moving of data

---

3

Data sharing and self-service access for more users and more data while strengthening governance and security

Introducing ...

watsonx

IBM® watsonx.data™

an open, hybrid, and governed
fit-for-purpose data
store optimized to scale all data,
analytics and
AI workloads

The platform for AI and data

watsonx

Scale and accelerate the impact of AI with trusted data.

watsonx.ai

- Train, validate, tune and
- deploy AI models

A next generation enterprise studio for AI builders to train, validate, tune, and deploy both traditional machine learning and new generative AI capabilities powered by foundation models. It enables clients to build AI applications in a fraction of the time with a fraction of the data.

watsonx.data

Scale AI workloads, for all your data, anywhere

Fit-for-purpose data store, built on an open lakehouse architecture, supported by querying, governance and open data formats to access and share data.

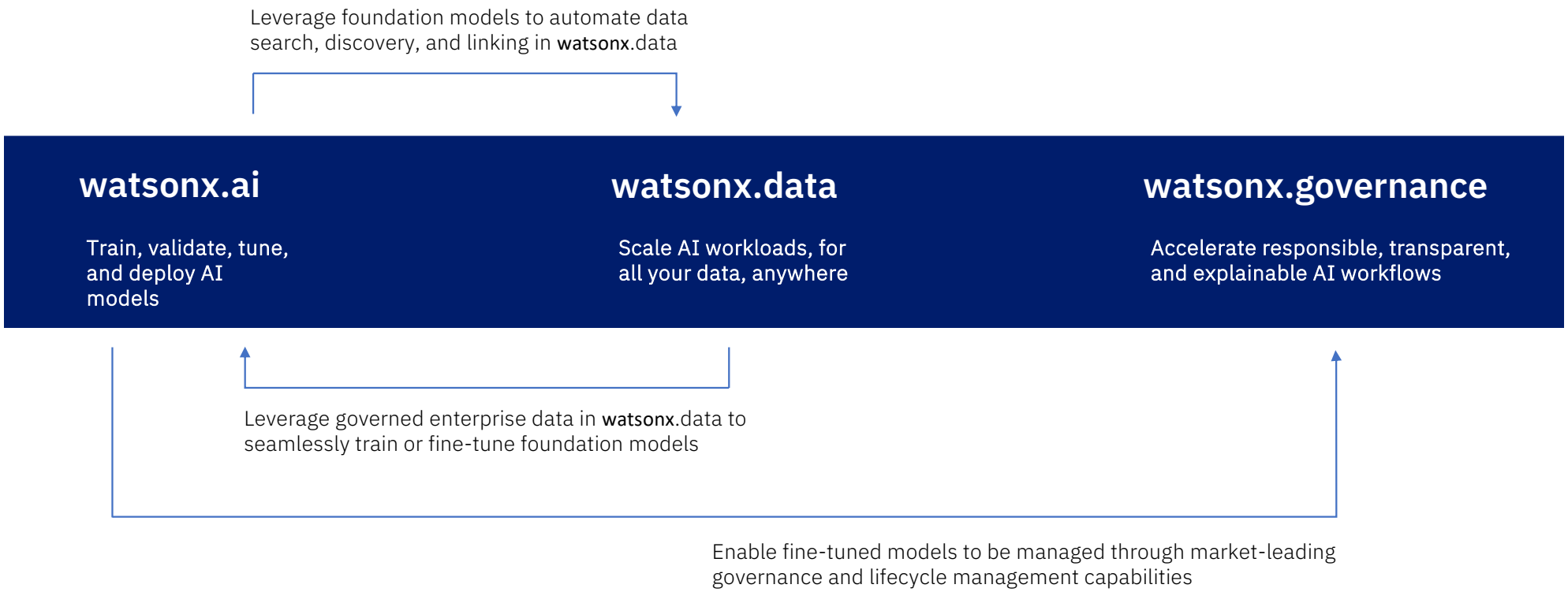
watsonx.governance

Accelerate responsible, transparent and explainable AI workflows

End-to-end toolkit for AI governance across the entire model lifecycle to accelerate responsible, transparent, and explainable AI workflows.

Put AI to work with watsonx

Scale and accelerate the impact of AI with trusted data



watsonx.data

Scale AI workloads, for all your data, anywhere

A fit-for-purpose data store, based on an open lakehouse architecture, supported by querying, governance and open data formats to access and share data

Access all your data through a single point of entry across all clouds and on-premises environments.

Get started in minutes with built-in governance, security and automation.

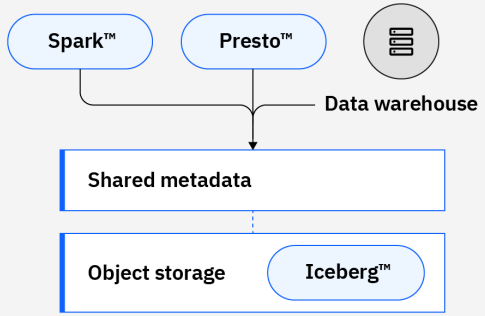
Reduce the cost of a data warehouse by up to 50%* through workload optimization across multiple query engines and storage tiers.

*When comparing published 2023 list prices normalized for VPC hours of IBM watsonx.data to several major cloud data warehouse vendors. Savings may vary depending on configurations, workloads and vendors.

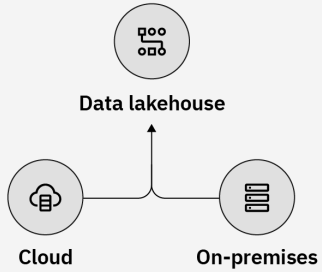
Access all your data across hybrid cloud through a single point of entry

An open data store, based on an open lakehouse architecture built for hybrid deployment of your data, analytics, and AI workloads

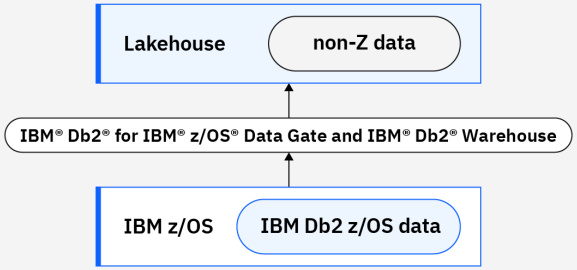
1 Share a single copy of data with tools that can read open data formats to minimize data duplication



2 Connect to and access data remotely across hybrid cloud with the ability to cache remote sources



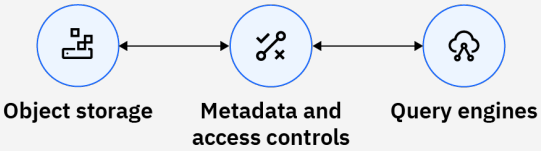
3 Synchronize and incorporate Db2 for z/OS data for lakehouse analytics



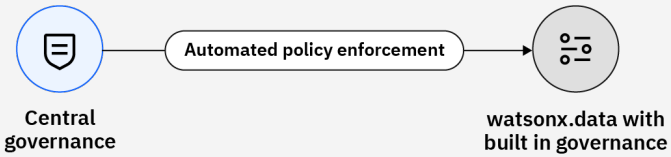
Get started in minutes with built-in governance, security and automation

Accelerate time to trusted analytics and AI

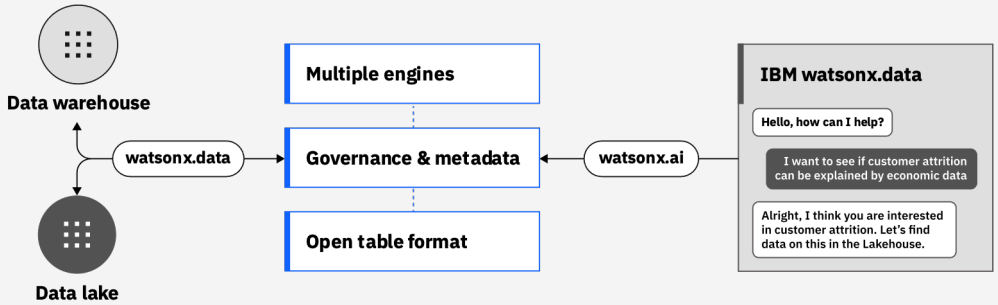
Connect to your existing analytics data and deploy fit-for-purpose query engines in minutes



Address enterprise compliance and security using built-in centralized governance across your data ecosystem



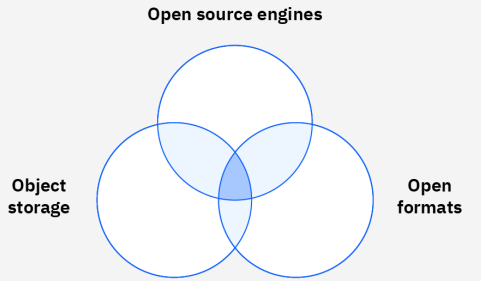
Use foundation models to discover, augment, refine and visualize **watsonx.data** data and metadata



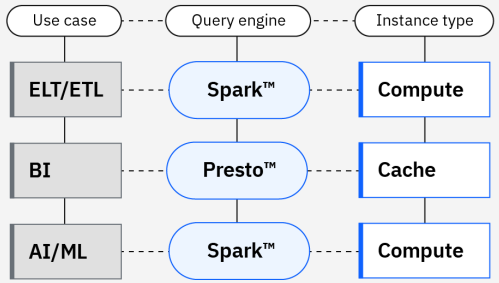
Reduce your data warehouse costs by up to 50%* by optimizing workloads

Optimize workloads from your data warehouse when you take advantage of low-cost object storage and fit-for-purpose query engines

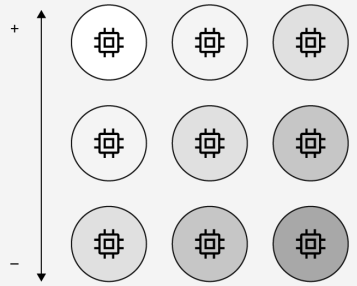
1 Share data between multiple analytics engines



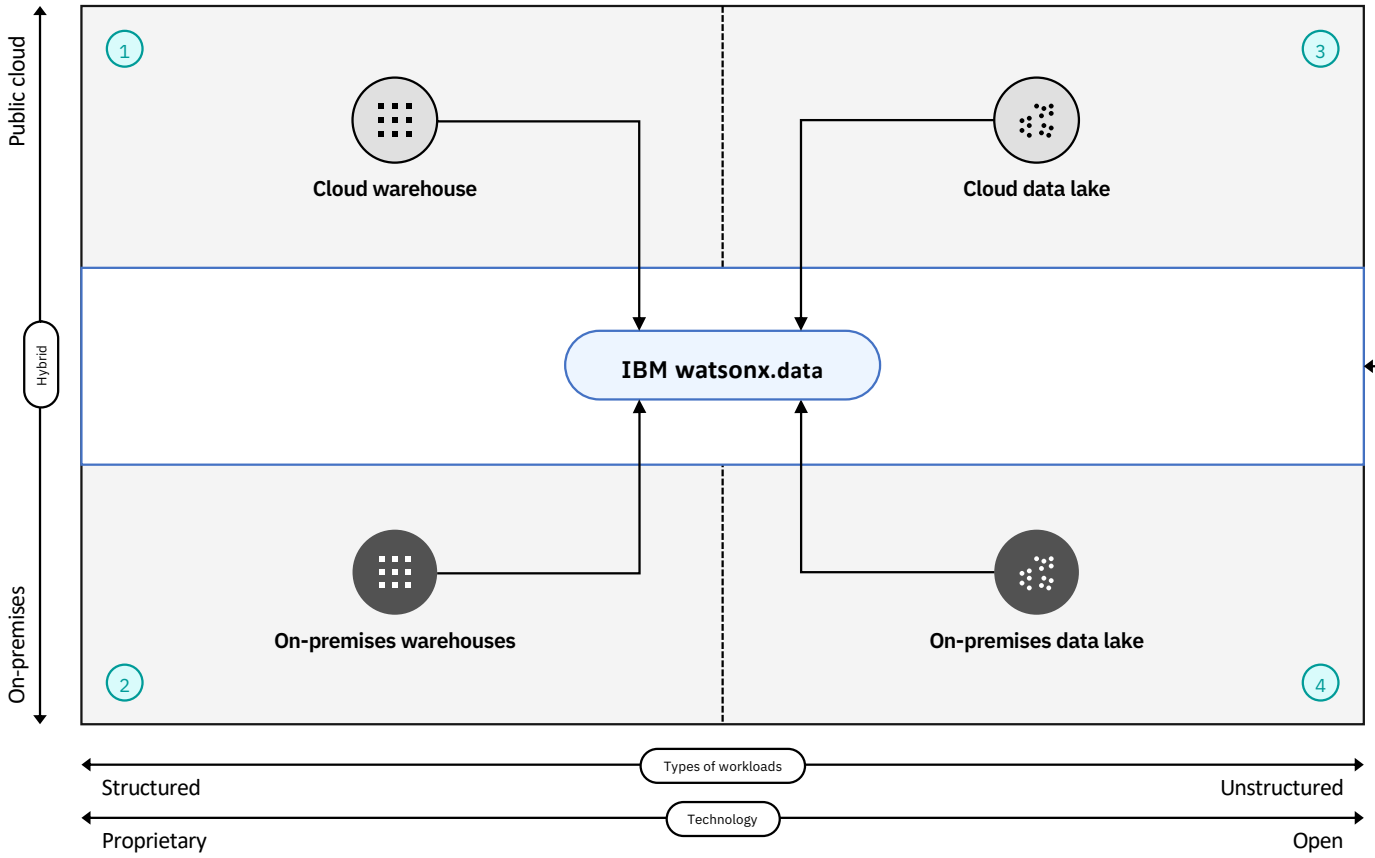
2 Use fit-for-purpose compute and cache-optimized instances



3 Scale up and scale down automatically



Access all your data, quickly and optimize your data architecture with multi-engine support and hybrid deployment of analytics and AI workloads



1
Optimize costly cloud warehouses
Make the most of fit-for-purpose query engines and compute resources

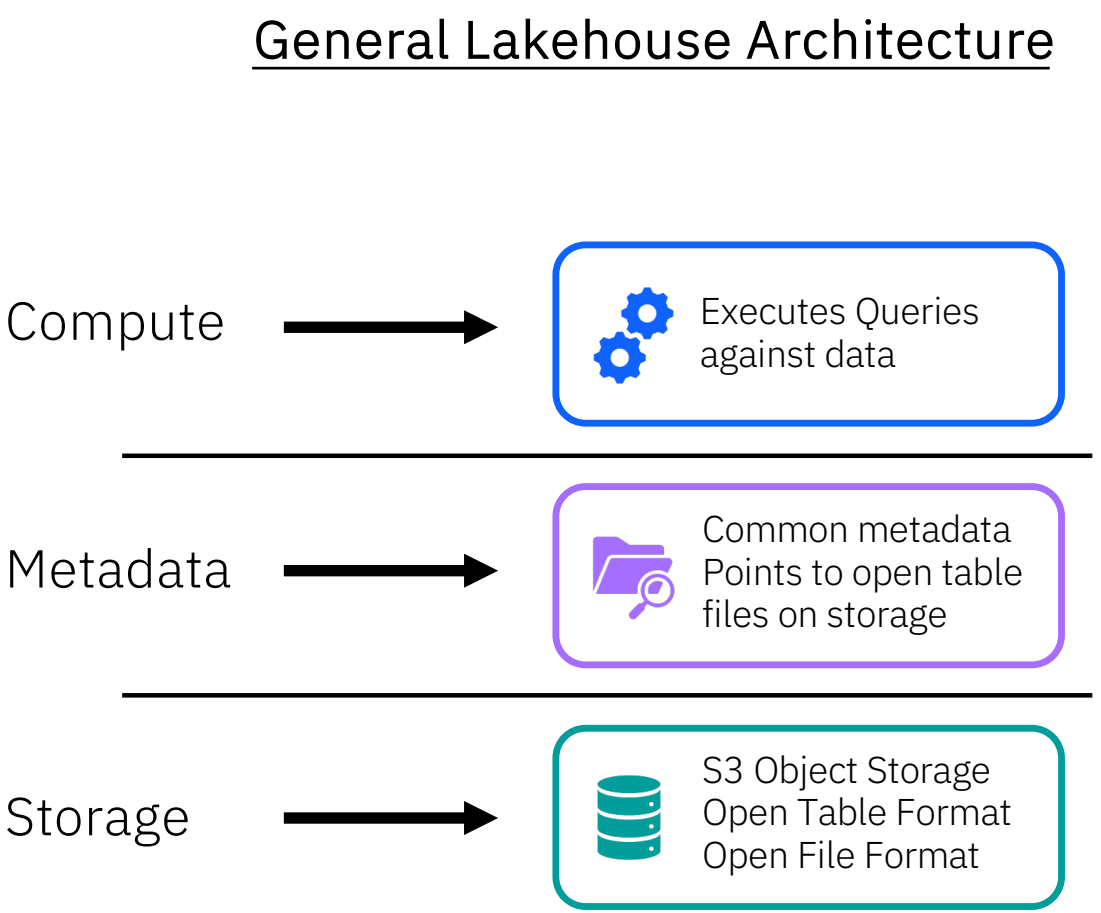
2
Optimize & access on-premises warehouses
Use low-cost object storage and fit-for-purpose engines

3 4
Modernize data lakes
Run existing reporting and enable new AI workloads without the cost and complexity of Hadoop

1 2 3 4
Deploy across hybrid cloud and multicloud
Seamlessly deploy to both the public cloud and to your existing on-premises investment

watsonx.data

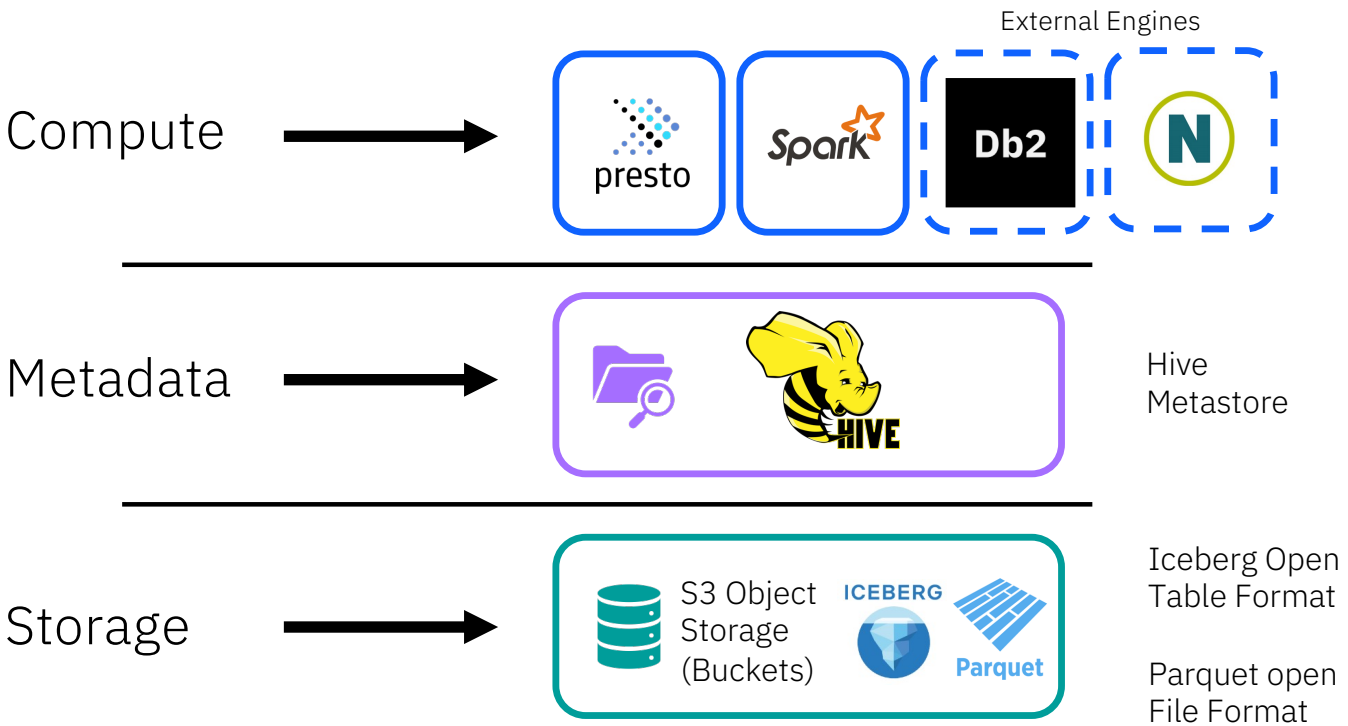
What is a lakehouse architecture?



watsonx.data

What is the watsonx.data architecture?

watsonx.data Architecture



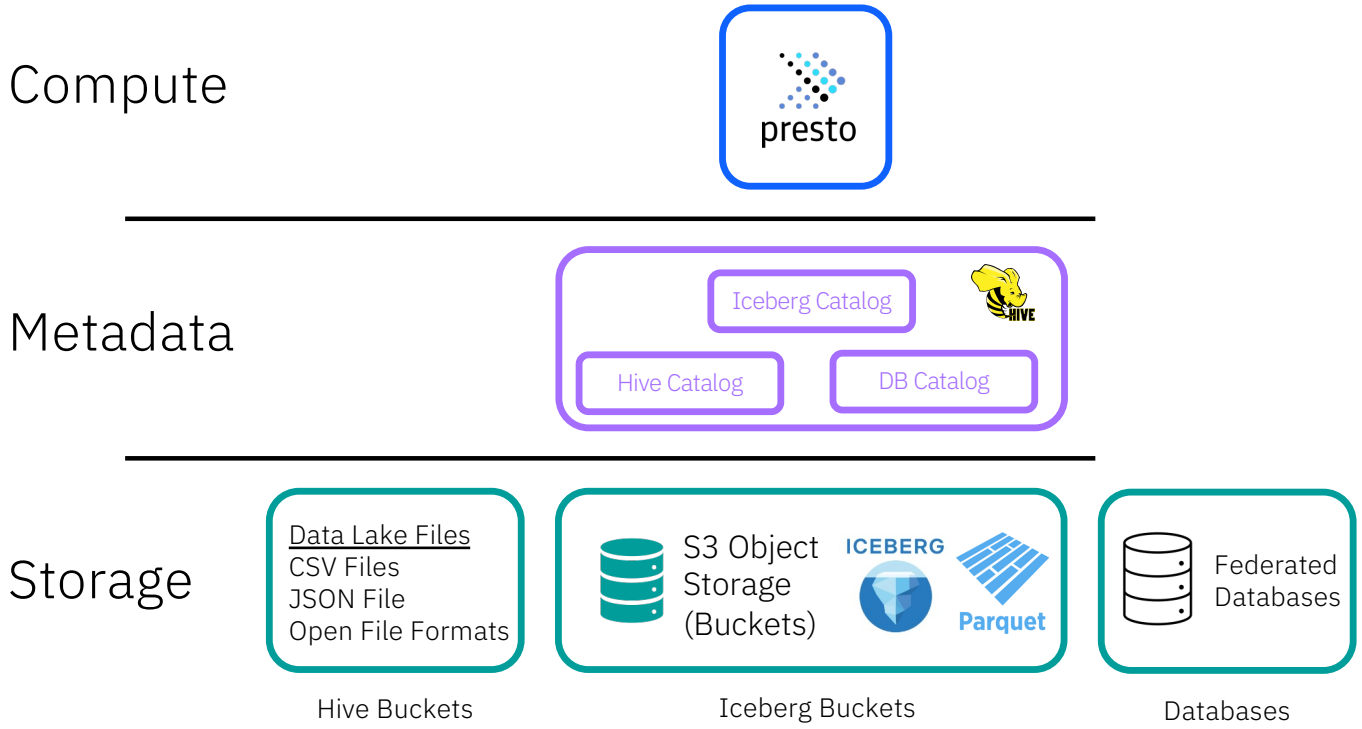
watsonx.data

The Presto Engine adds functionality:

Connectors to non-iceberg data files

Connectors to databases (some allowing CRUD operations)

watsonx.data Architecture



watsonx.data

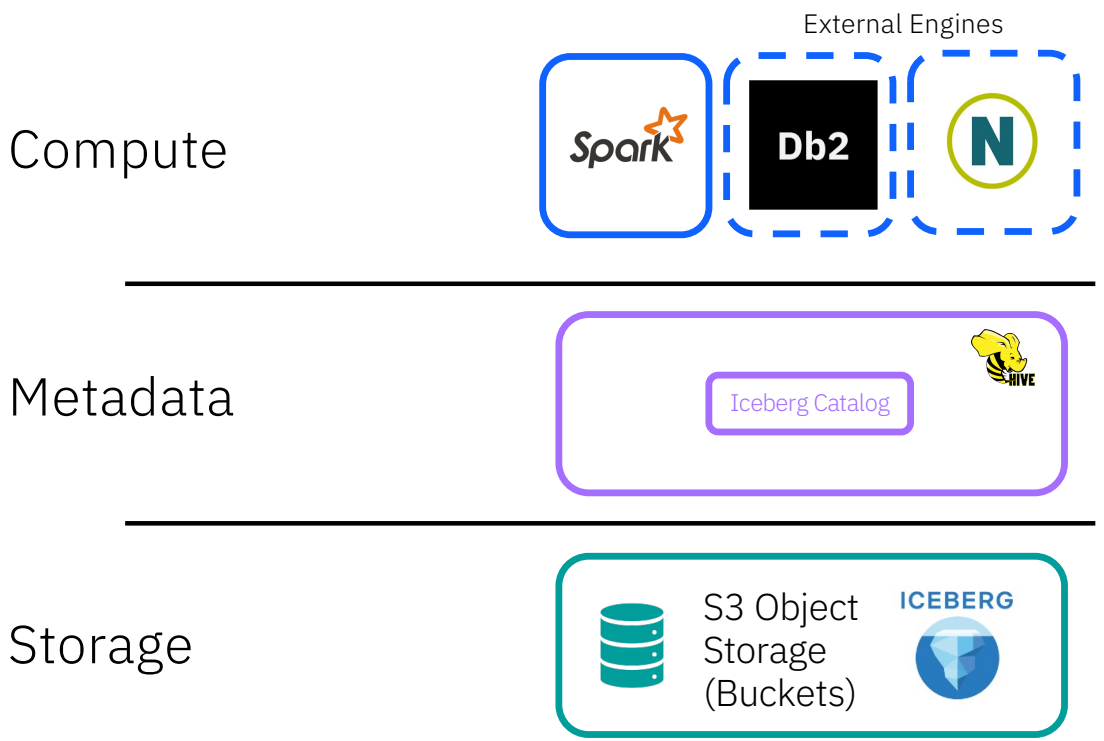
Spark, Db2, Netezza Engines

Can connect to Iceberg tables

Can access their own data objects/table within their environments

Cannot access Presto connectors

watsonx.data Architecture



Agenda

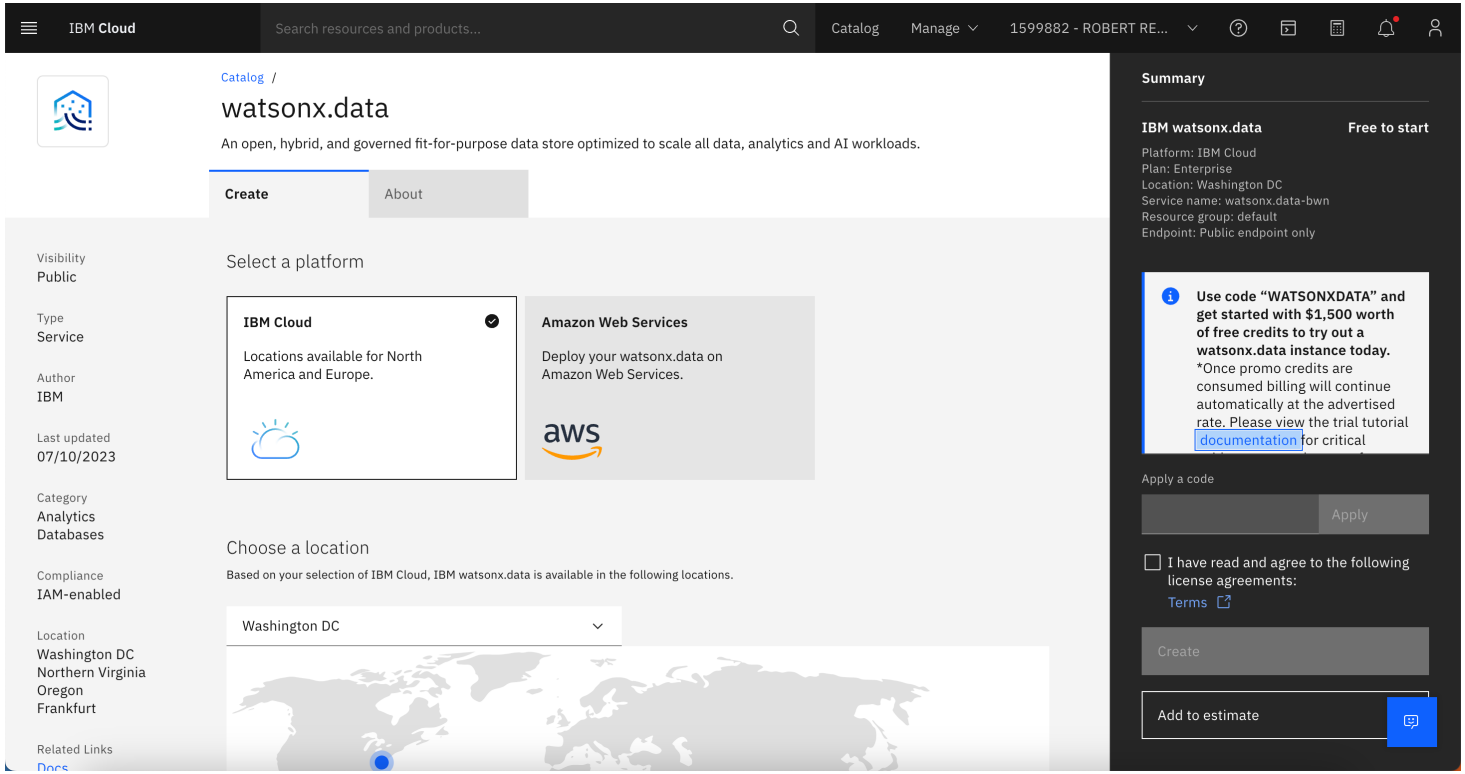
- What is this watsonx.data?
- Architecture
- **Deployment options**
- Interface/tools
- Techzone watsonx.data Lab

watsonx.data

https://cloud.ibm.com/lakehouse

Deployment options

SaaS → IBM Cloud
AWS



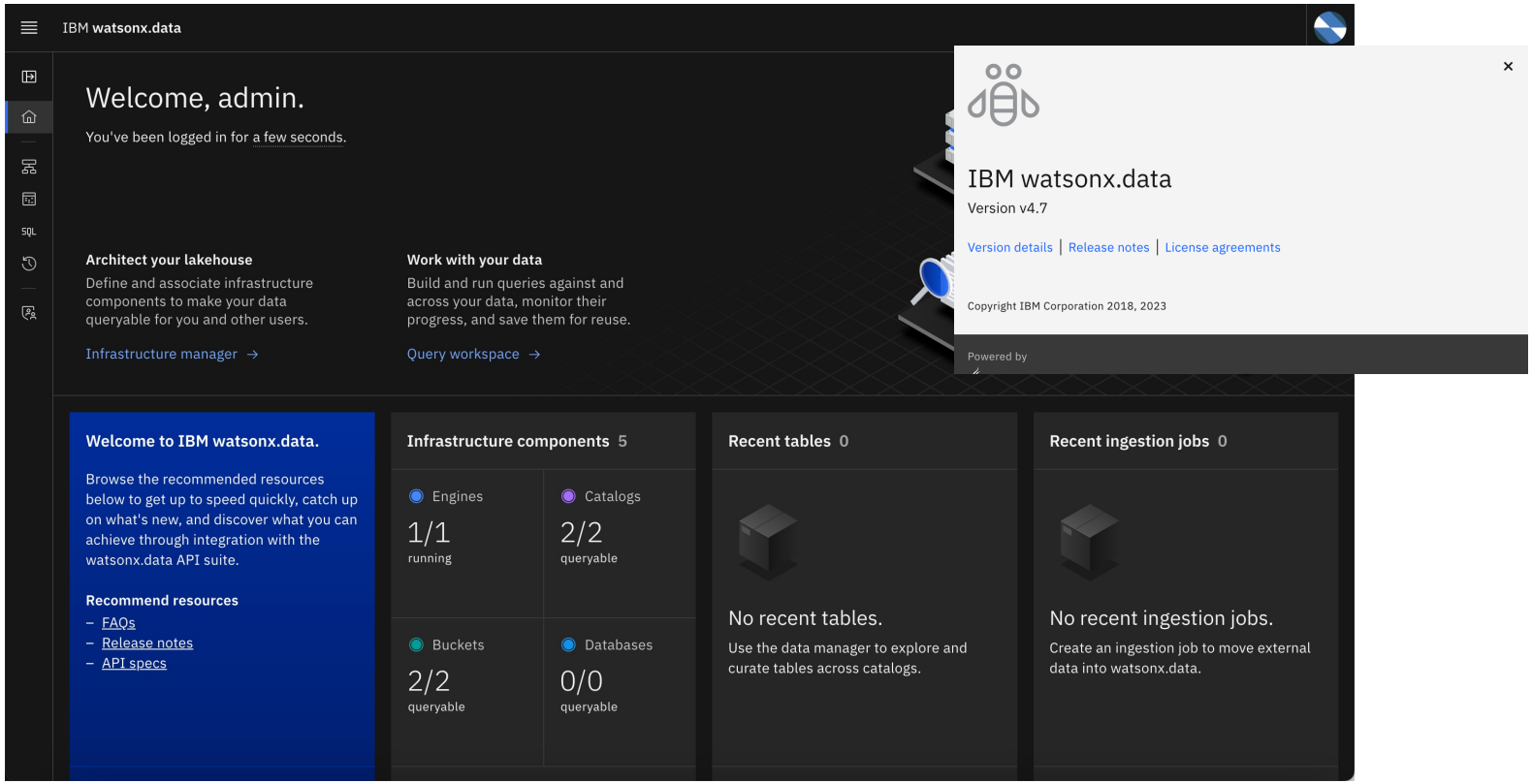
watsonx.data

- Engine Scaling
- Multi-Engine Support
- Caching Support
- For Enterprise Workloads

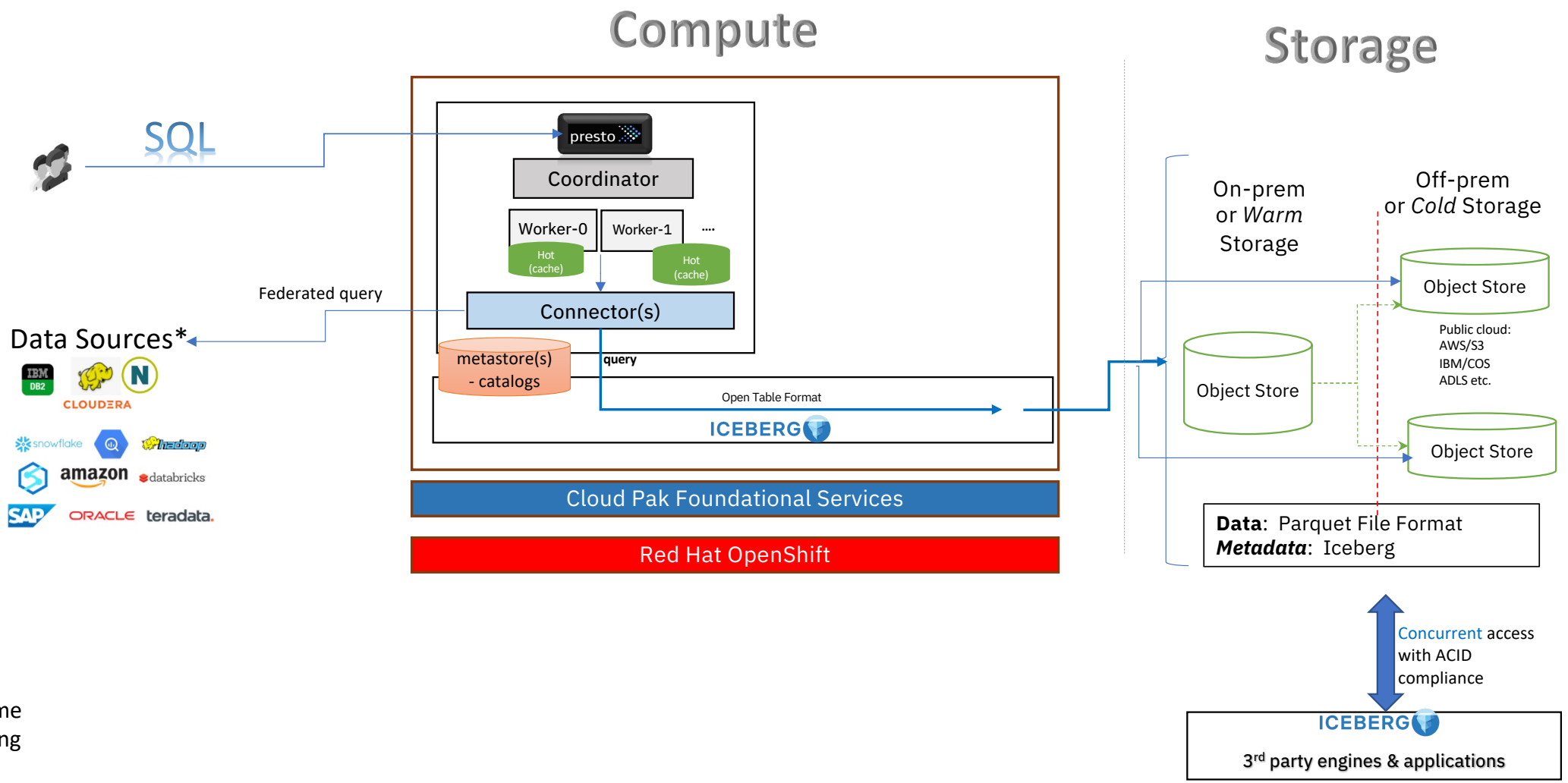
- Runs on Red Hat Openshift

Deployment options

Software (on-premises) → Stand alone or CP4D Cartridge

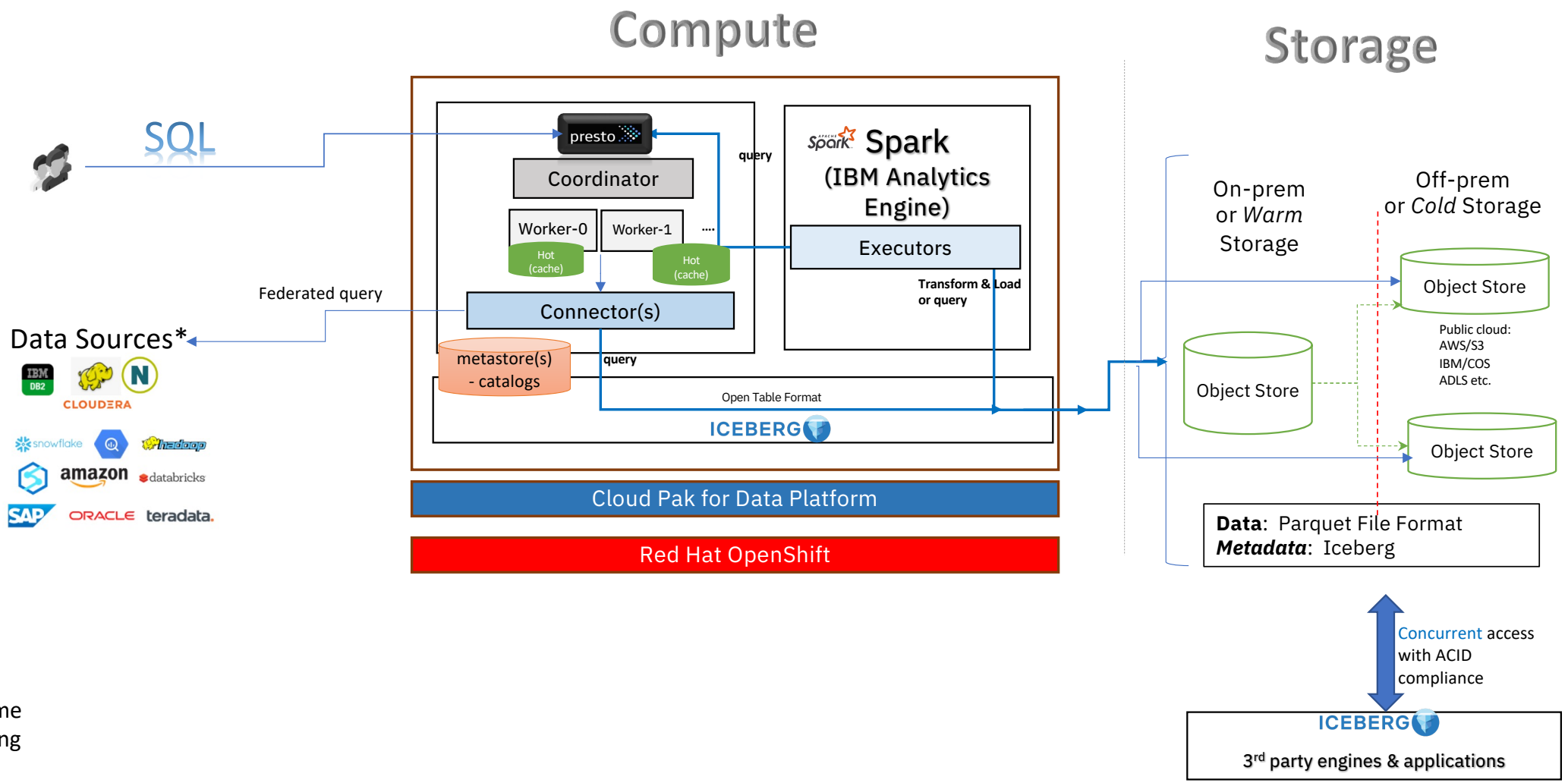


watsonx.data on Openshift – stand alone



* Some coming soon

watsonx.data on Cloud Pak for Data



* Some coming soon

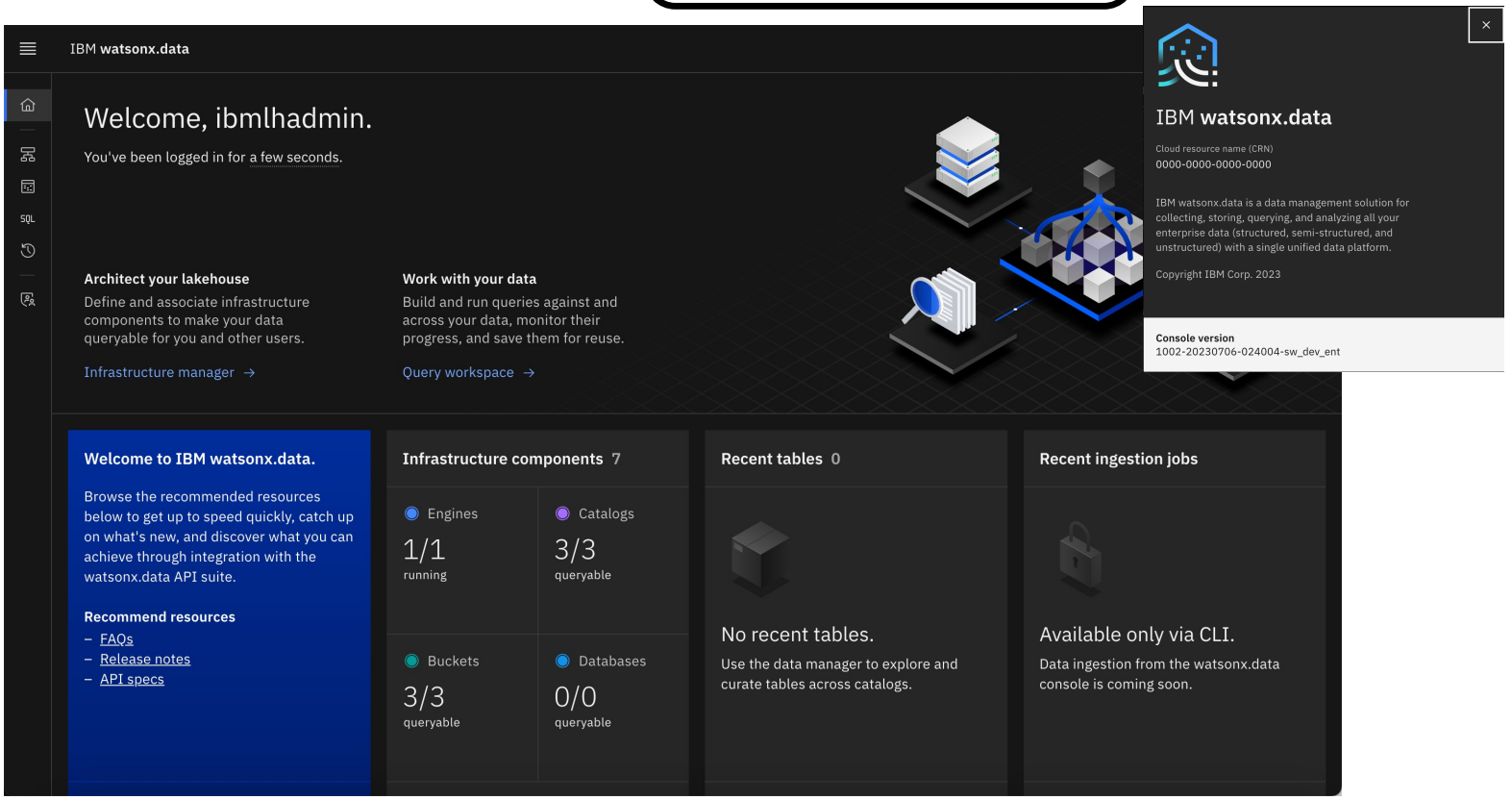
watsonx.data

- Single Presto Engine
- Cannot Scale
- Single User
- No Caching
- Used to test/play
- Not for Production

- Install on Linux (Recommended)
- Runs on Docker/Podman

Deployment options

Developer Edition → Runs on Podman or Docker



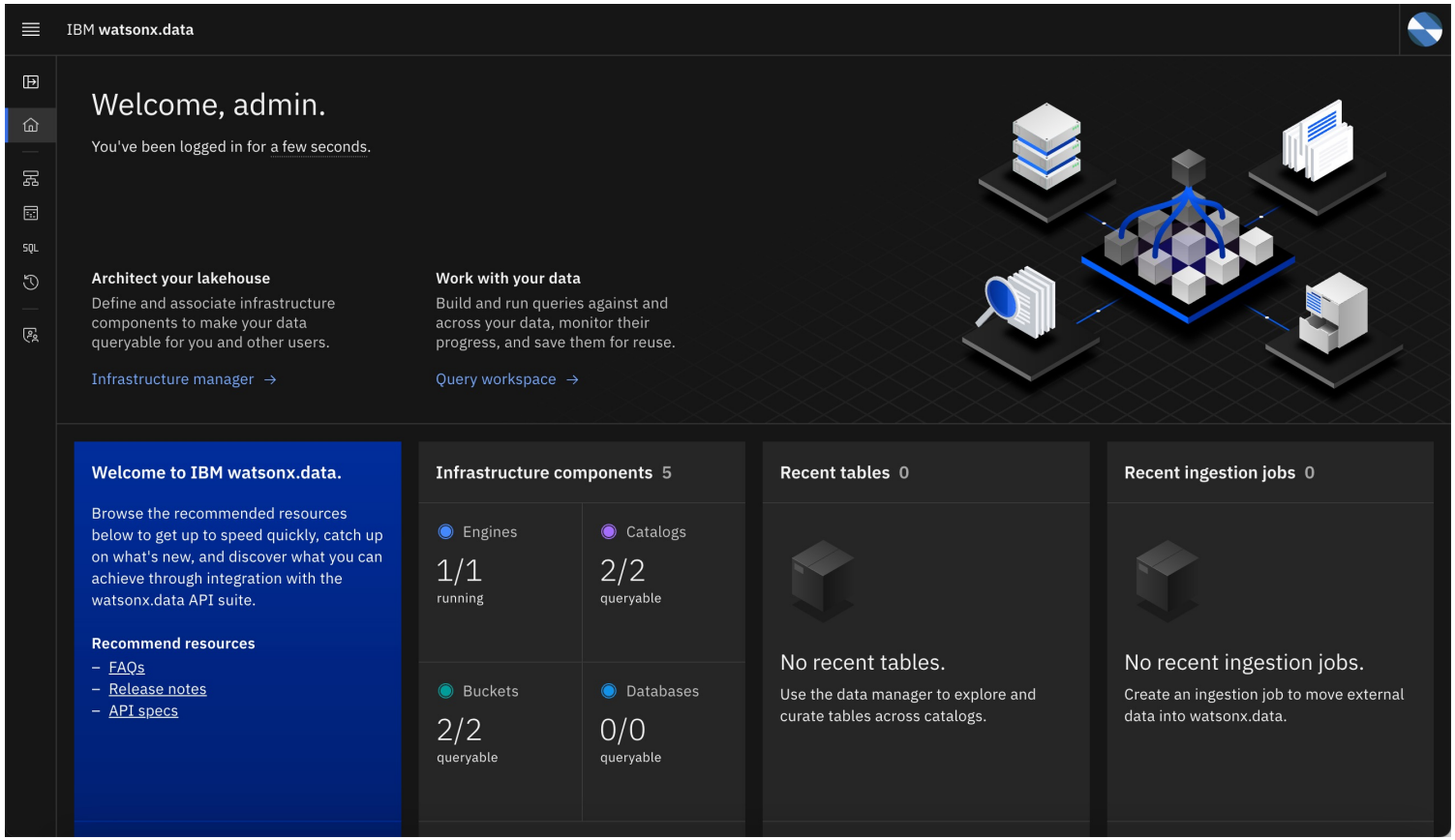
Agenda

- What is this watsonx.data session?
- Architecture
- Deployment options
- **Interface/tools**
- Techzone watsonx.data Lab

watsonx.data

Web Console

- Main UI entry to watsonx.data
- Browse engines, catalogs, buckets, schemas, tables
- Run SQL Queries
- Access Control
- Ingestion (small data)



watsonx.data

- Utilities to access and manage watsonx.data
- Browse engines, catalogs, buckets
- Includes presto-cli client
- dev-sandbox container for python code with libraries installed
- ibm-lh to manage instance and load data
- Docker or podman required

- https://www.ibm.com/docs/en/watsonxdata/1.0.x?topic=package-installing-lh-client

Client Package (Tools)

ibm-lh commands and usage

Last Updated: 2023-08-30

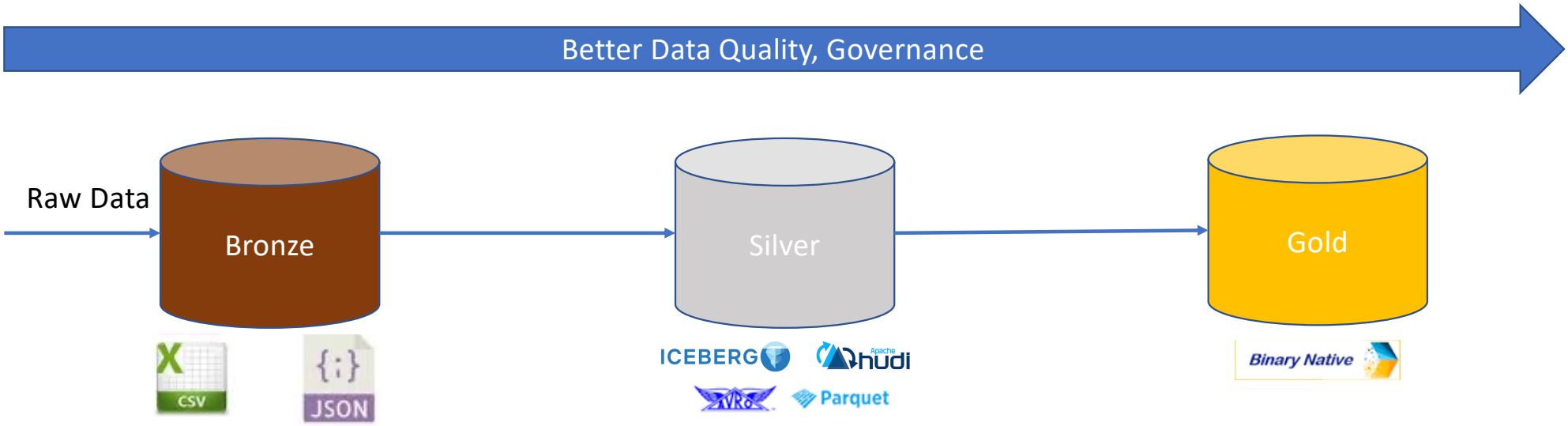
The `ibm-lh` command-line utility is a terminal-based interface that is designed to facilitate interaction with watsonx.data resources.

The `ibm-lh` CLI utility supports the following commands:

- `ibm-lh engine`
- `ibm-lh config`
- `ibm-lh database`
- `ibm-lh bucket`
- `ibm-lh data-copy`
- `ibm-lh table-maint`

Operating system	x86-64	Docker / Pod man / Colima install instructions
Linux	✓	Docker Podman
Windows	✓	Docker Podman
Mac OS x86	✓	Docker Podman
Mac with Apple Silicon with Rosetta Emulation		Docker Colima

Data foundation with Medallion Architecture



Ingestion/Staging Tables	Refined Tables	Ingestion Tables
• No business rules or transformations of any kind. • Should be easy to get new data onto this layer. • Not so high performance.	• Prioritize speed to market and write performance. • Just enough (low volume) transactions. • Quality data expected. • Interoperable format. • Flexible schema. • Medium performance.	• Prioritize business use cases and user experience. • Precalculated, business-specific transformations/aggregations. • Well defined schema. • Highest performance • High volume transactions.

Agenda

- What is this watsonx.data session?
- Architecture
- Deployment options
- Interface/tools
- **Lakehouse Components**

Lakehouse Components

Components for a modern Lakehouse solution

Engines

- Engines enables querying, analytics and transformations for the lakehouse. The expectation is to enable the use of the **right engine for the right workloads & use cases.**

Metadata repository

A repository maintains schema and table metadata so that all engines and users can **consistently** locate and query against their data in a structured manner.

Data Storage

The data storage is where the data is physically stored. Customers would expect to use their own storage such as S3 or HDFS and locate them wherever they need. With **compute engines separated from storage**, concurrent accesses by multiple engines must be enabled in the Lakehouse

Data Governance

A lakehouse needs to be able to **enforce Data policies** for access, privacy and safe harbor or sovereignty regulations

Cloud & on-premise Service

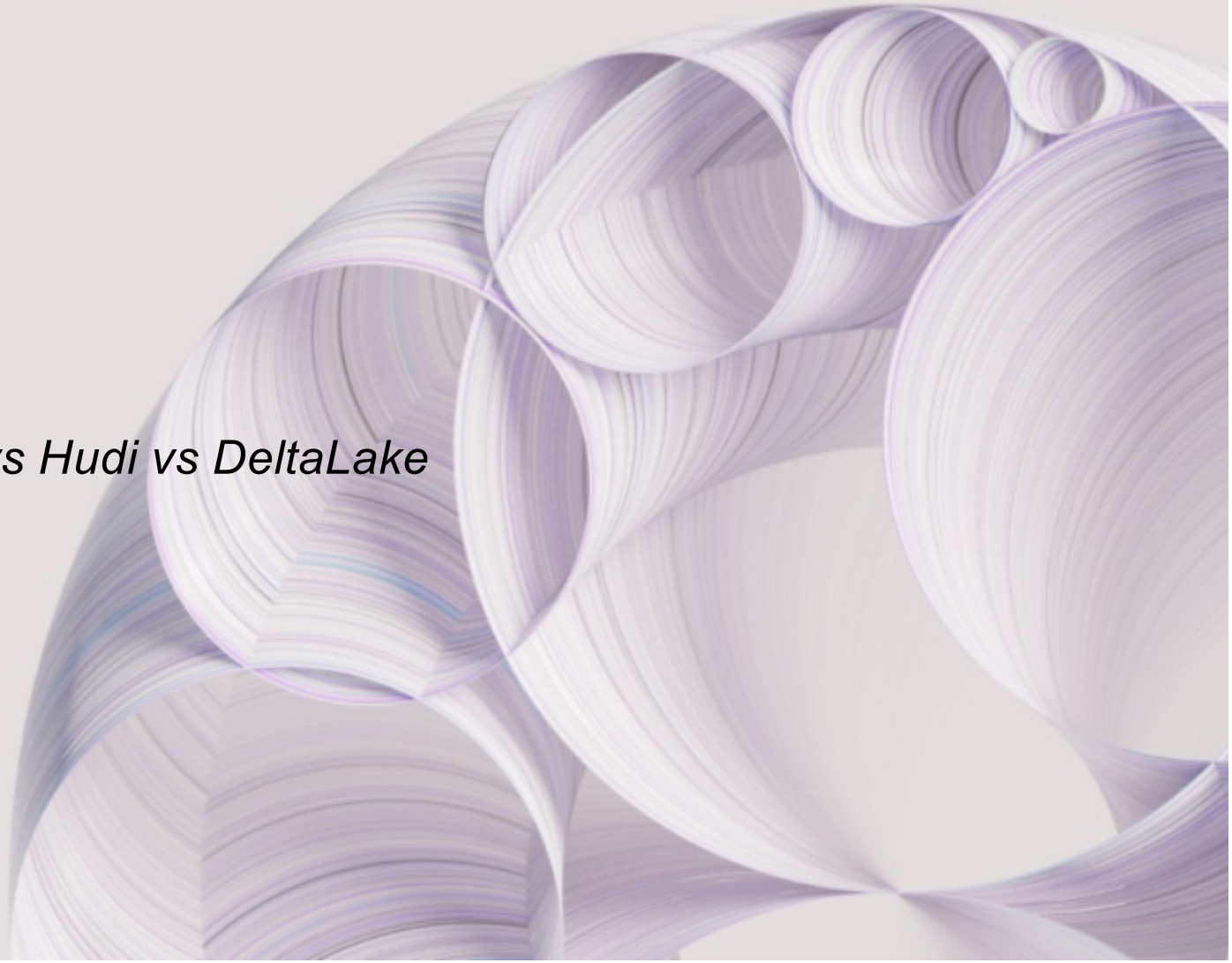
Lakehouses need to be deployable or burstable **anywhere** and even span clouds in a hybrid fashion. Applications and end users would need to access lake houses engines from anywhere

PrestoDB

Hive Metastore

File Formats

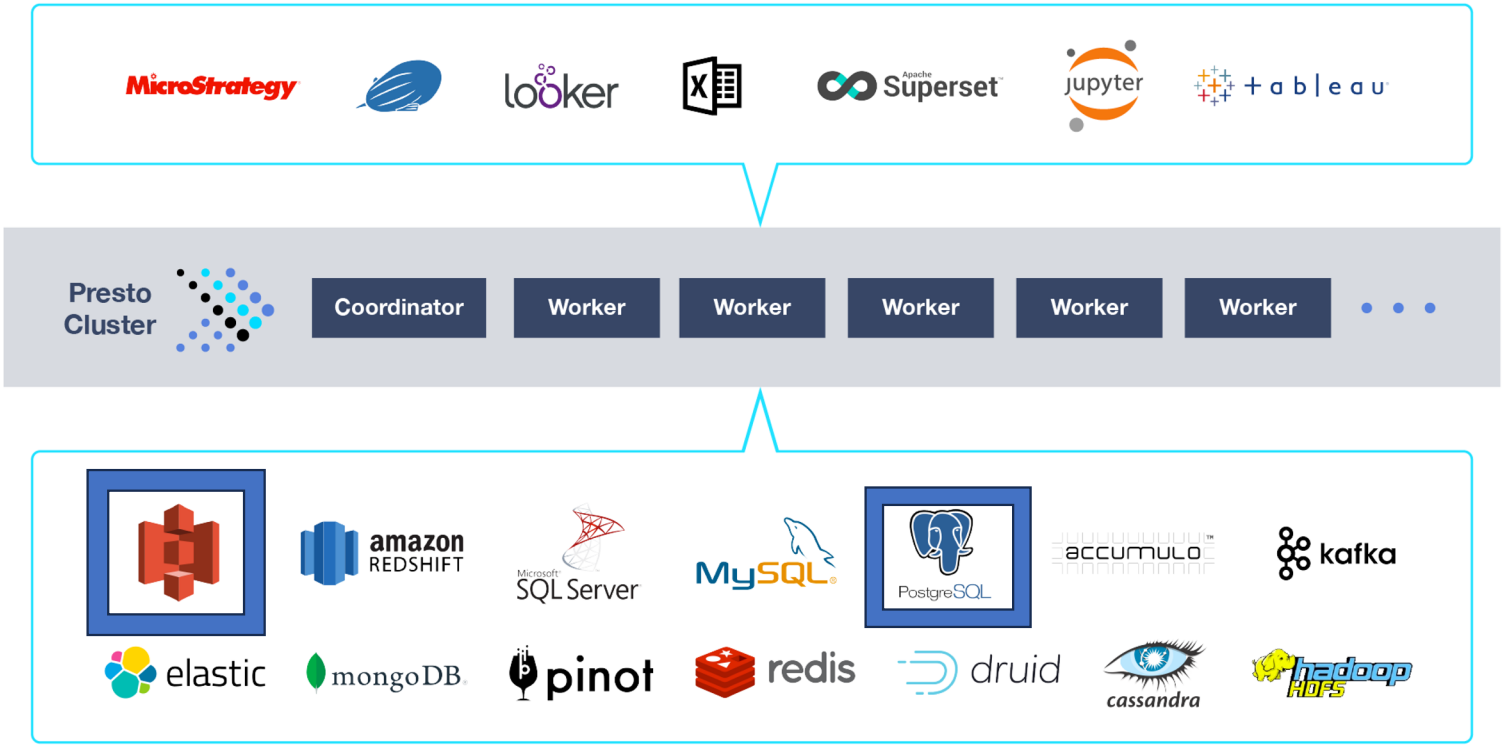
Table Formats *Iceberg vs Hudi vs DeltaLake*



presto

1. **Scalable**
   Support Big Data
2. **Low-latency**
   Interactive experience, great for ad hoc use cases
3. **Pluggable architecture**
   Support multiple data sources and federated query
4. **ANSI SQL**
   Lingua franca of data literate population
5. **Open source and governance**
   Diverse innovation, no vendor lock-in

Distributed query engine



Presto runs reliably at massive scale

See how some the largest internet-scale companies are using Presto today. It doesn't matter if you're operating at Meta-like scale or at just a few nodes - Presto is for everyone!

300PB data lakehouse
1K daily active users
30K queries/day

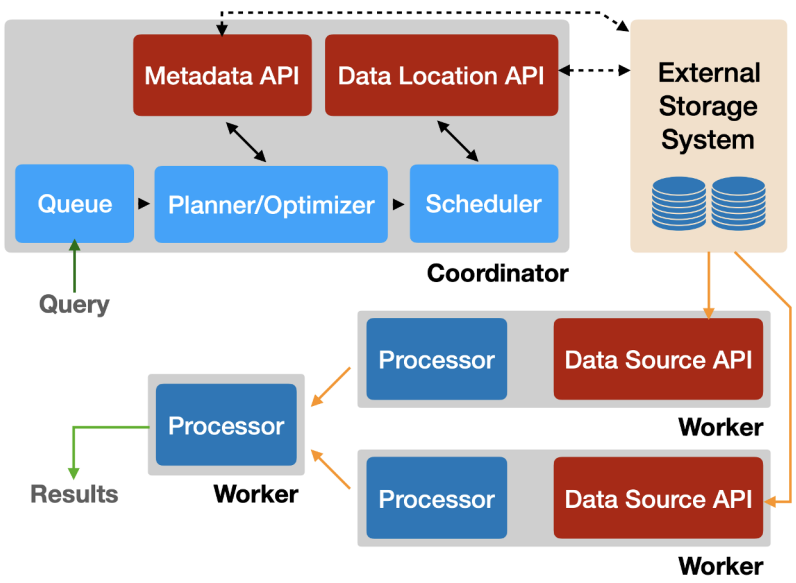
2 regions
20 clusters
8K nodes

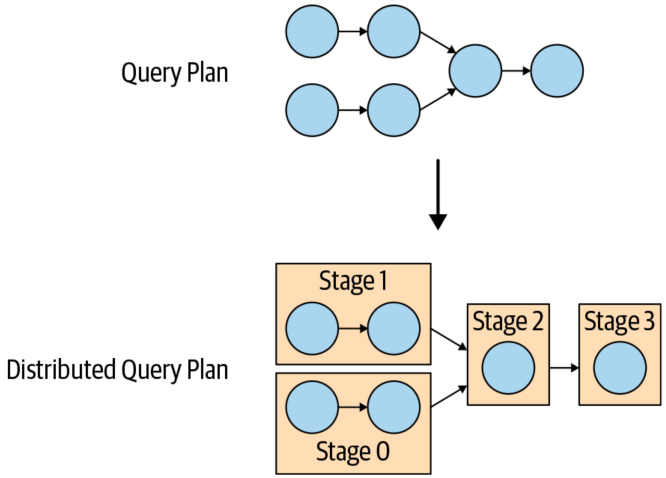
7K weekly active users
100M+ queries/day
50PB HDFS bytes read/day

10K+ compute cores
1M queries/day

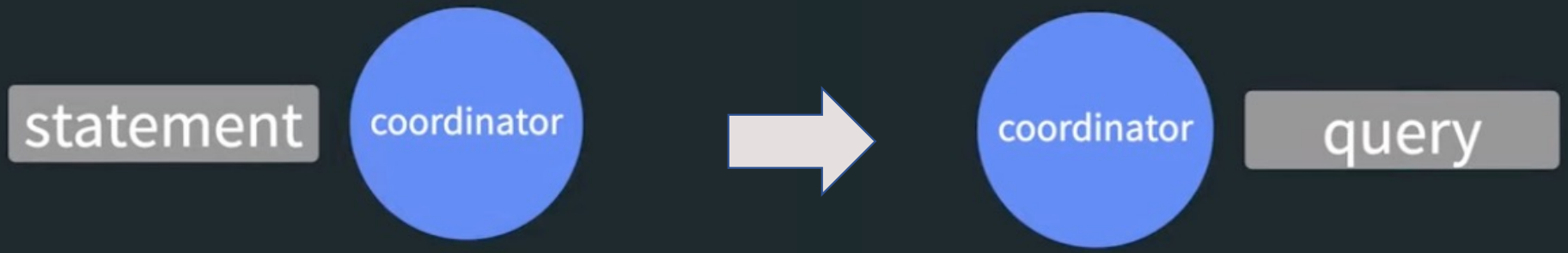
presto

1. SQL client connects (e.g. authenticates) to Coordinator
2. Coordinator queues query, if necessary
3. Coordinator creates a *query plan* for execution
   * Simple (logical) plan
   * Distributed query plan
   * Split into plan fragments -> Stage
4. Schedule *tasks* across Workers
5. Tasks works on a unit of data called *splits*

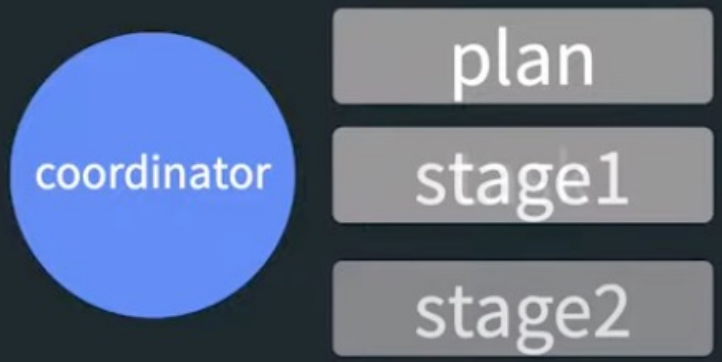




Coordinator takes the statement and parses it into a query



Internal planner generates an optimized plan as a series of stages

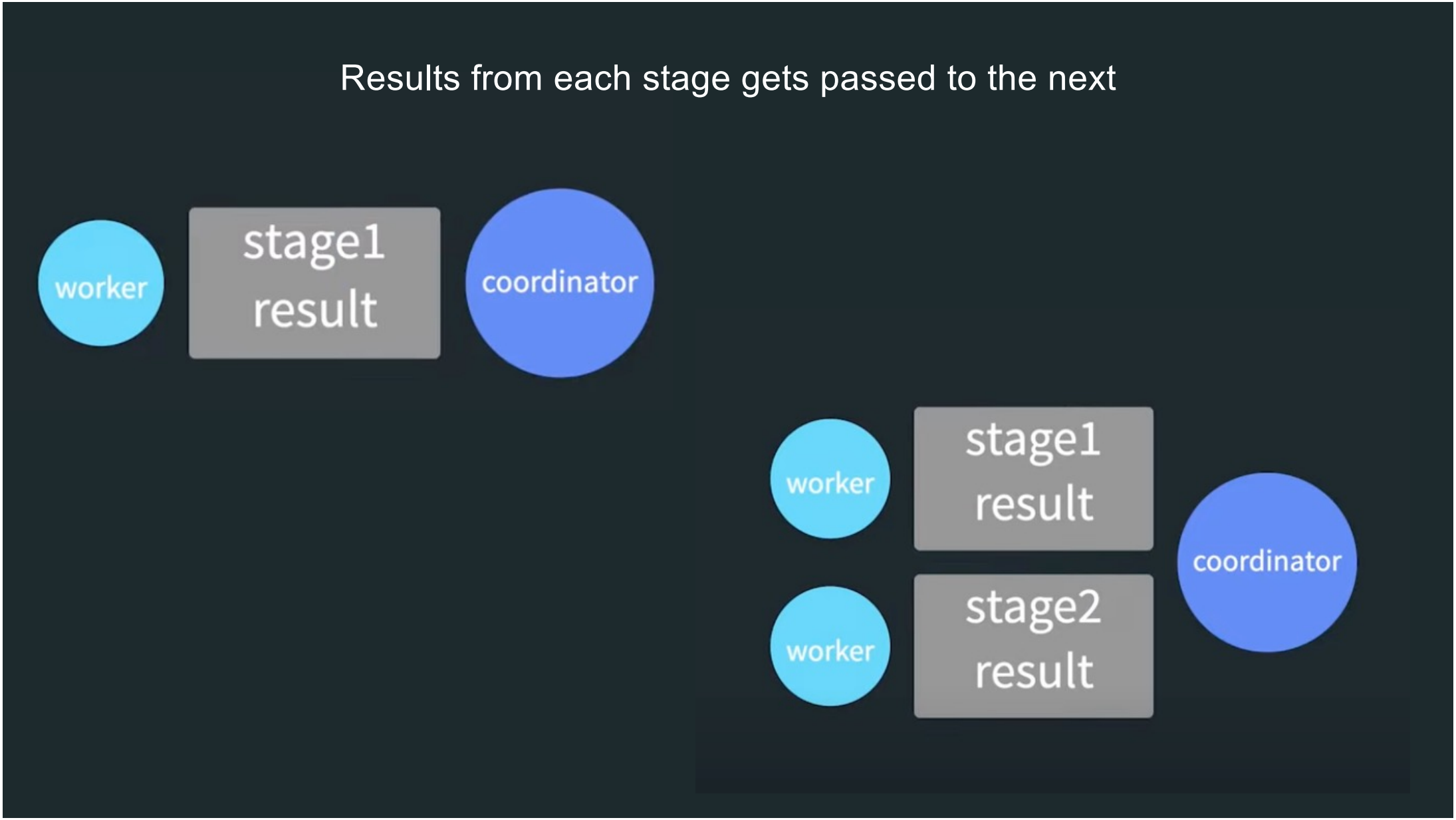


Which are then separated into tasks

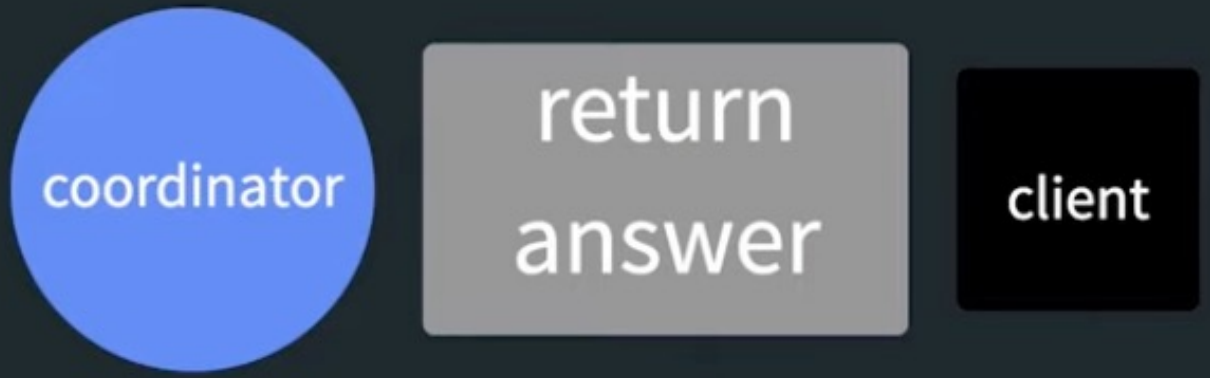
Tasks are assigned to workers to process in parallel where workers find and use the relevant connector for the appropriate data source

Output is returned by the workers until each task and stage is complete

Results from each stage gets passed to the next

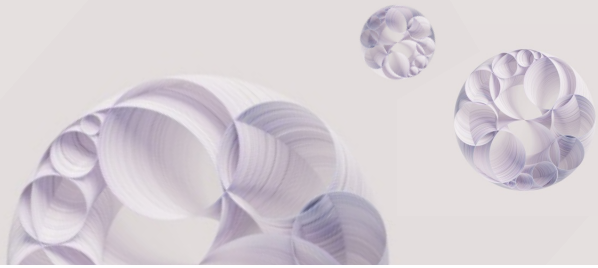


Until the result is calculated and returned to the client



Who is presto for?

- Presto allows watsonx.data to be a preferred tool for data engineers
- Helps to manage multiple query languages and interfaces from siloed databases and storage
- Ease of use with data analytics and business intelligence tools



takeaways

One Language
- One familiar ANSI SQL Language and one engine for your data analytics
- Interactive and batch workloads
- Small and large amounts of data

One Interface
- One place to reach all of your data regardless of where it is
- Separates storage and compute allows you to run queries where your data lives

Scalability
- Price-performance and optimized to meet growing data sizes and workloads
- Scales from a few to thousands of users

presto takeaways

Ad-hoc Query
- Removes the need for traditional ETL processes
- Ready to query, wherever and whenever you want

Reporting and Dashboarding
- Enhances business intelligence
- Create thorough visualizations and reports with all of your data in one

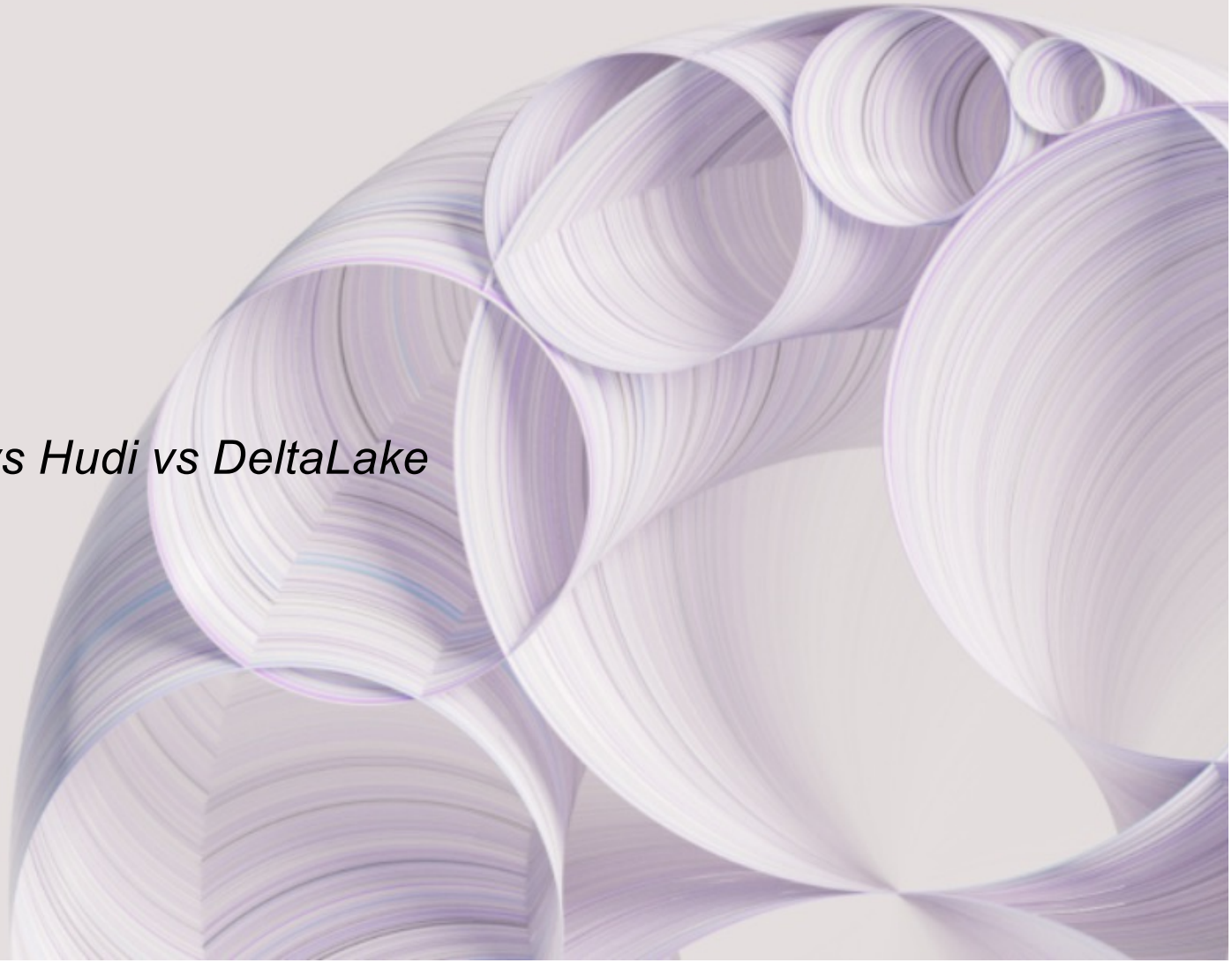
Contributes to ease of Open Data Lakehouse

PrestoDB

Hive Metastore

File Formats

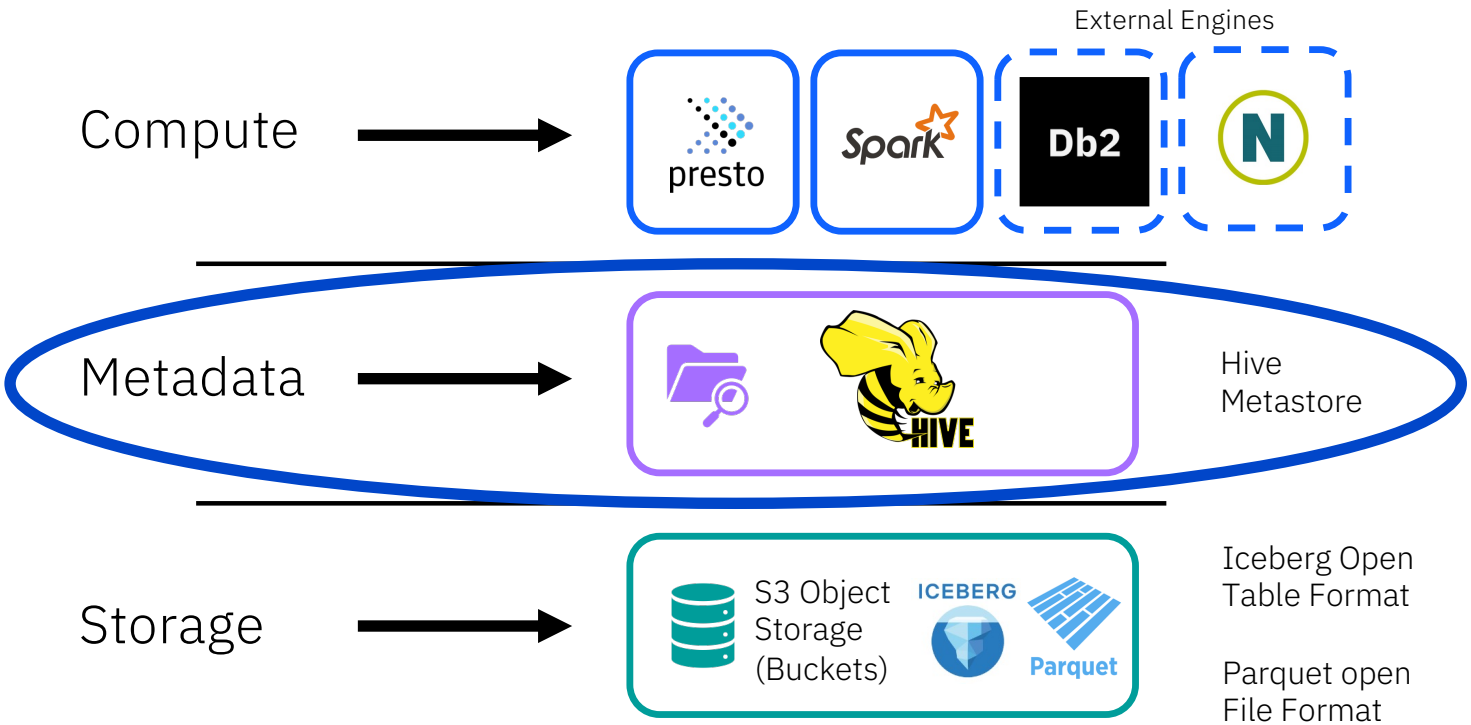
Table Formats *Iceberg vs Hudi vs DeltaLake*



watsonx.data

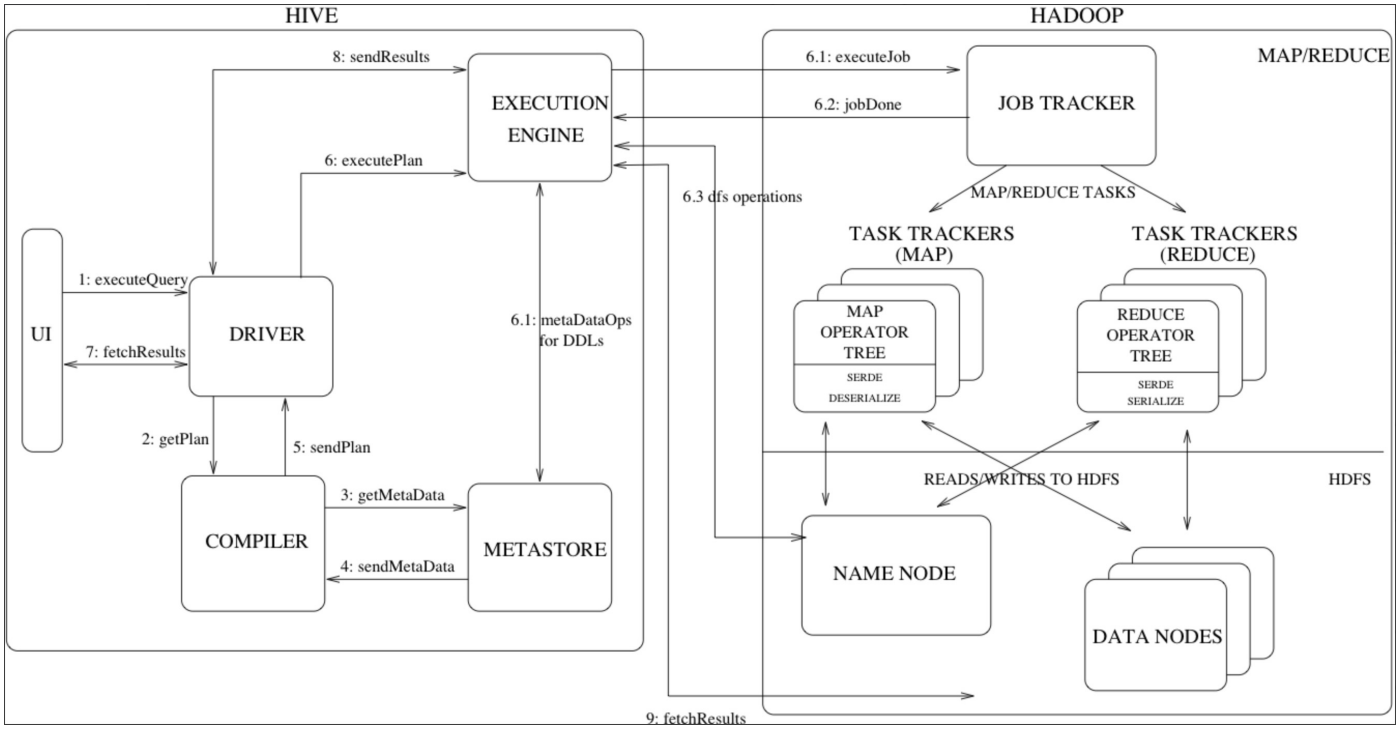
What is the watsonx.data architecture?

watsonx.data Architecture



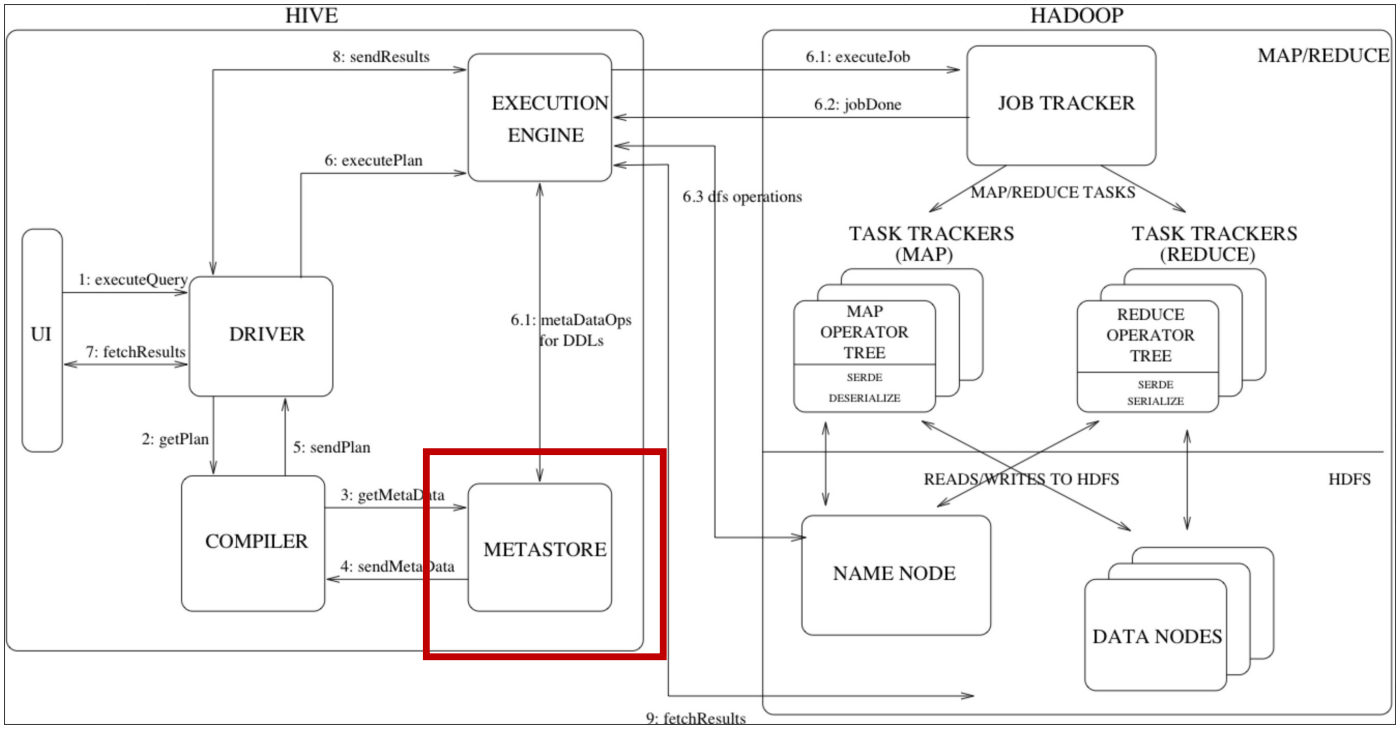
Hive vs Hive Metastore

- Runs over the Hadoop framework
- provides SQL like interface for processing/querying the data
- runs its query using HQL (Hive query language)
- Same structure as RDBMS and almost the same commands can be used in Hive
- Hive can store the data in external tables so it's not mandatory to use HDFS
- Also supports file formats such as ORC, Avro files, Sequence File and Text files, etc.



Hive vs Hive Metastore

- Runs over the Hadoop framework
- provides SQL like interface for processing/querying the data
- runs its query using HQL (Hive query language)
- Same structure as RDBMS and almost the same commands can be used in Hive
- Hive can store the data in external tables so it's not mandatory to use HDFS
- Also supports file formats such as ORC, Avro files, Sequence File and Text files, etc.



Hive Metastore

Central storage point for all the meta-information about your data storages

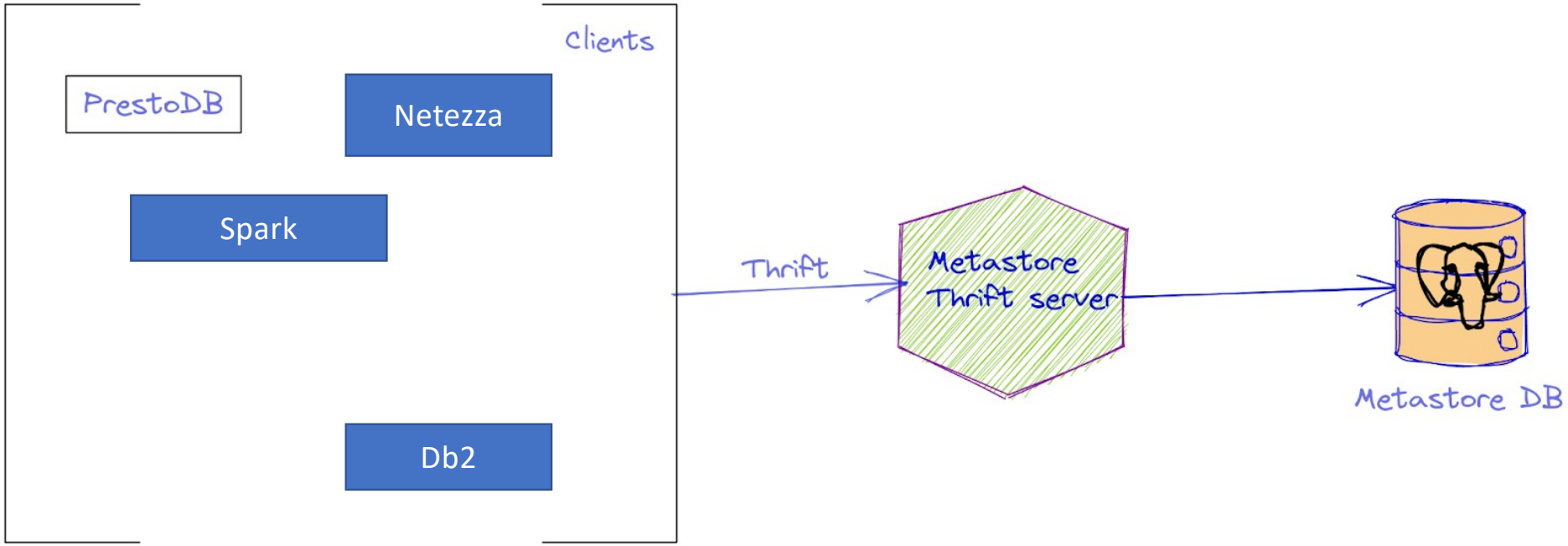
- Central repository for lakehouse query engines
- Stores metadata information about connected tables, views, partitions, columns, and their respective schemas
- Stores information such as the schema of tables, their column names, types, and partitioning information
    - This information is used by the query engines to optimize query execution and improve performance

Hive Metastore

Central storage point for all the meta-information about your data storages

- Central repository for lakehouse query engines
- Stores metadata information about connected tables, views, partitions, columns, and their respective schemas
- Stores information such as the schema of tables, their column names, types, and partitioning information
    - This information is used by the query engines to optimize query execution and improve performance

- Tracks the location of data stored in the storage systems, making it easier for the query engine to access and process the data
- Typically implemented as a relational database, such as MySQL, PostgreSQL, or Oracle
- Handles concurrent access and provides high availability and fault tolerance


Clients
PrestoDB
Netezza
Spark
Db2
Thrift
Metastore
Thrift server
Metastore DB

Key Benefits of Hive Metastore

Data Abstraction

- Typically, a user has to provide information about data formats, extractors and loaders along with the query
- Now, this information is given during table creation and reused every time the table is referenced
- Similar to traditional warehousing systems

Data Discovery

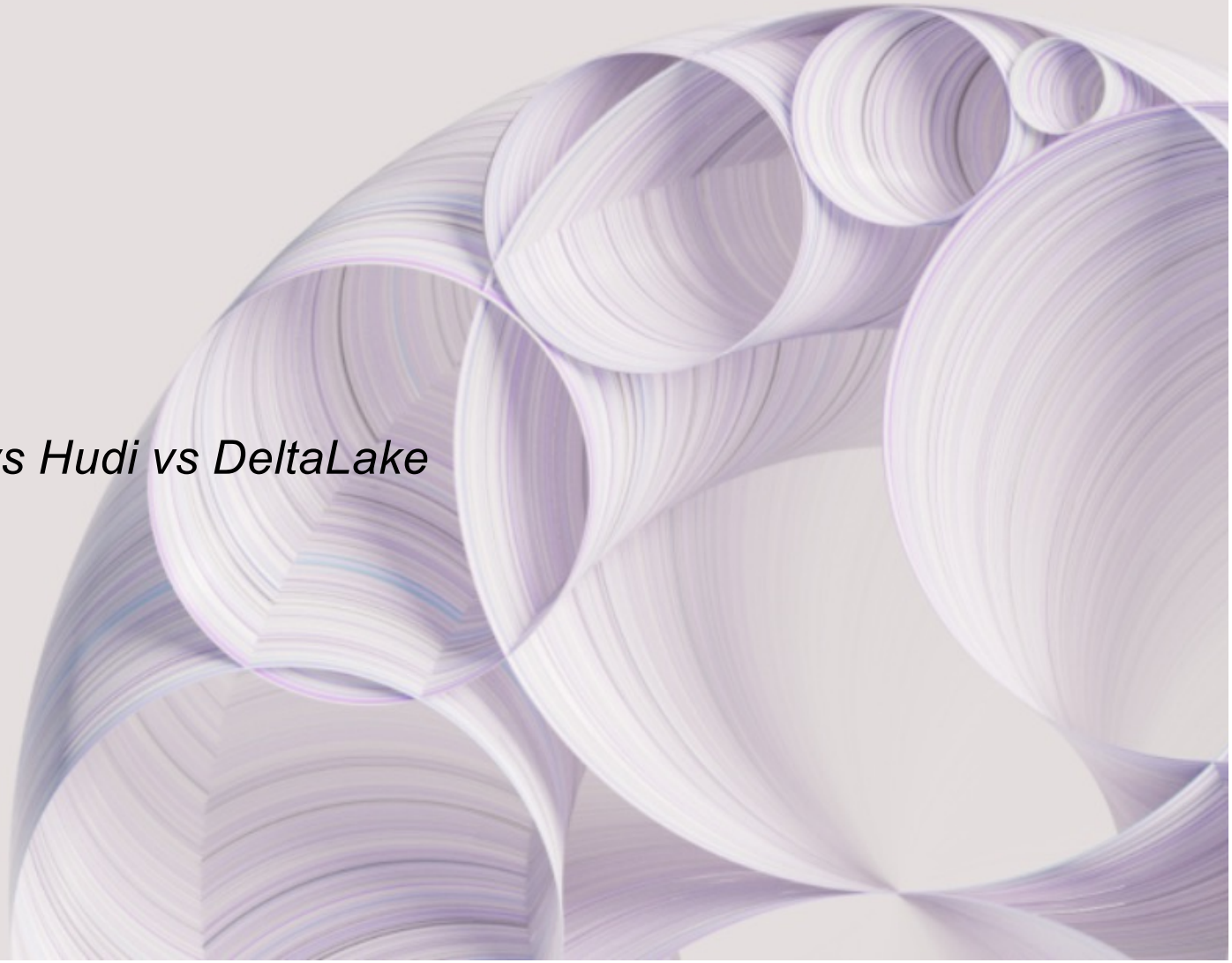
- Enables users to discover and explore relevant and specific data in the warehouse
- Ease of integration with other tools to expose and possibly enhance the information about the data and its availability
- metadata repository that is tightly integrated with all of the Lakehouse engines so that data and metadata are in sync

PrestoDB

Hive Metastore

File Formats

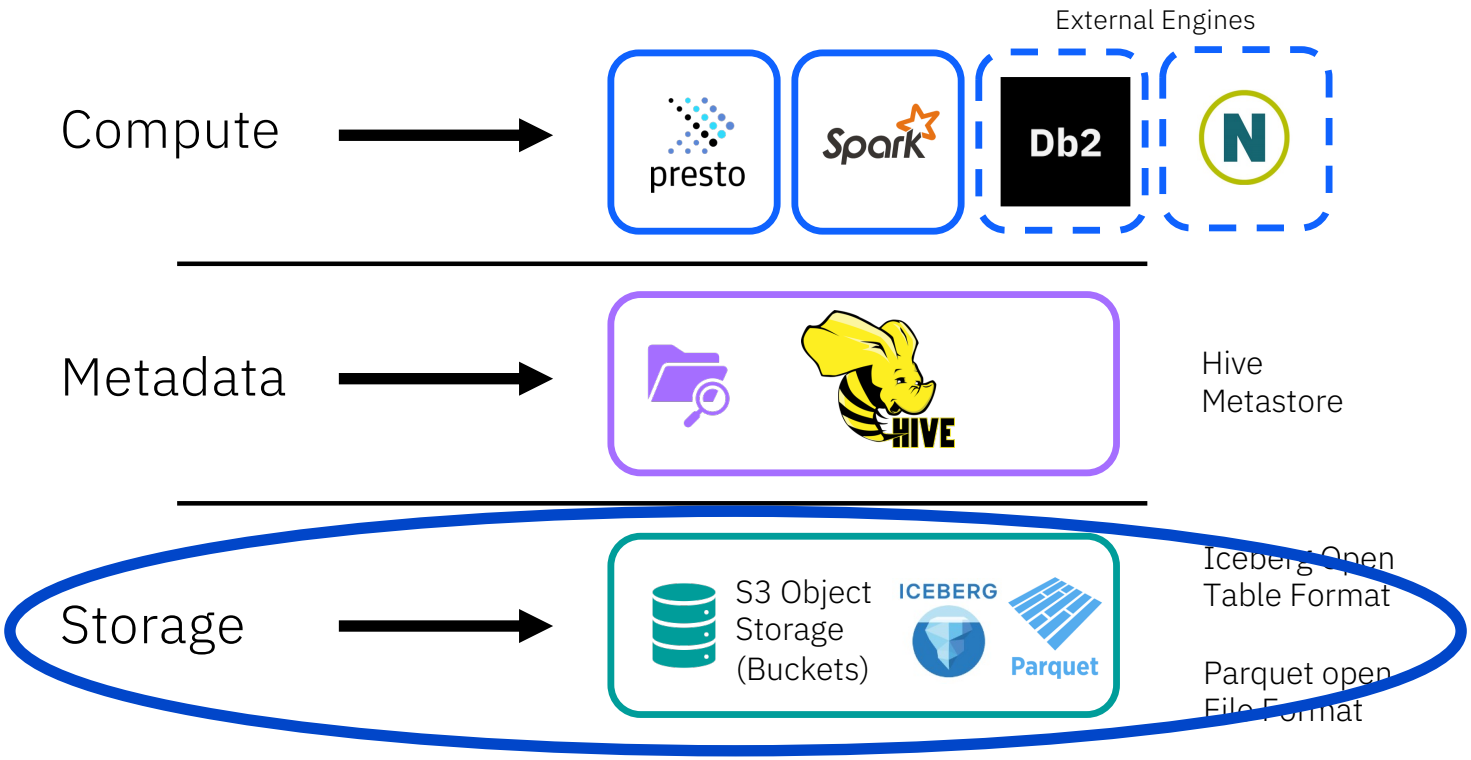
Table Formats *Iceberg vs Hudi vs DeltaLake*



watsonx.data

What is the watsonx.data architecture?

watsonx.data Architecture



File Formats

File Formats

- A row storage format that compresses data efficiently
- Avro format is a binary data storage protocol that provides data serialization
- Saves data in JSON format for easy data reading and interpretation
- Strong support for data schema which changes over time
    - Avro controls data schema changes such as missing fields, added fields, and changed fields
- High flexibility, but compared to columnar file formats, it has less compression.
- Not optimal for fast file search due to its row storage format.

File Formats

- Created by Cloudera and Twitter in 2013

- Can uniquely store data with nested structures in columns
  - nested fields can be read separately without reading all the nested structure fields

- Suitable for working with large volumes of complex data and offers various data compression and encoding options

- useful for reading specific columns from large tables
  - can only read the required columns instead of the entire table
  - faster data processing and reduces I / O referral time

- Column storage capability allows you to quickly filter unrelated data during queries

- There are different codecs for data compression and data files can have different types of compression

File Formats

- Optimized for reading, writing and processing data in Hive and was created by Hortonworks in 2013 to speed up Hive
- Stores a set of rows in a file so that each row of data has a column format
- Stores data compactly and allows you to skip unrelated parts without the need for complex or manual indexing
- Supports decimal data types, dates and complex types (struct, list, map and union)
- Generally in column file format, data read speed is high
- Suitable for analytical work, while in the format of row files, the speed of writing data is high
- Suitable for heavy transaction writing

File Formats

- Row based
- Indexed
- Less compression than Parquet and ORC but faster write speeds
- Up to 3x faster read times
- Good schema evolution support
- Can be used as a landing area prior to additional data transformations

- Columnar
- Originally also designed for Hadoop
- Works very well with Spark
- Good for traditional OLAP queries because of its columnar format
- Carries schema, is self describing
- Data stored in pages
- Good for complex nested data structures

- Also columnar
- Data stored in stripes
- Indexed rows, compatible with HDFS
- KPIs for compression and runtime are very similar to parquet
- Parquet better for Spark, ORC better for Hive/Hadoop

PrestoDB

Hive Metastore

File Formats

Table Formats *Iceberg vs Hudi vs DeltaLake*

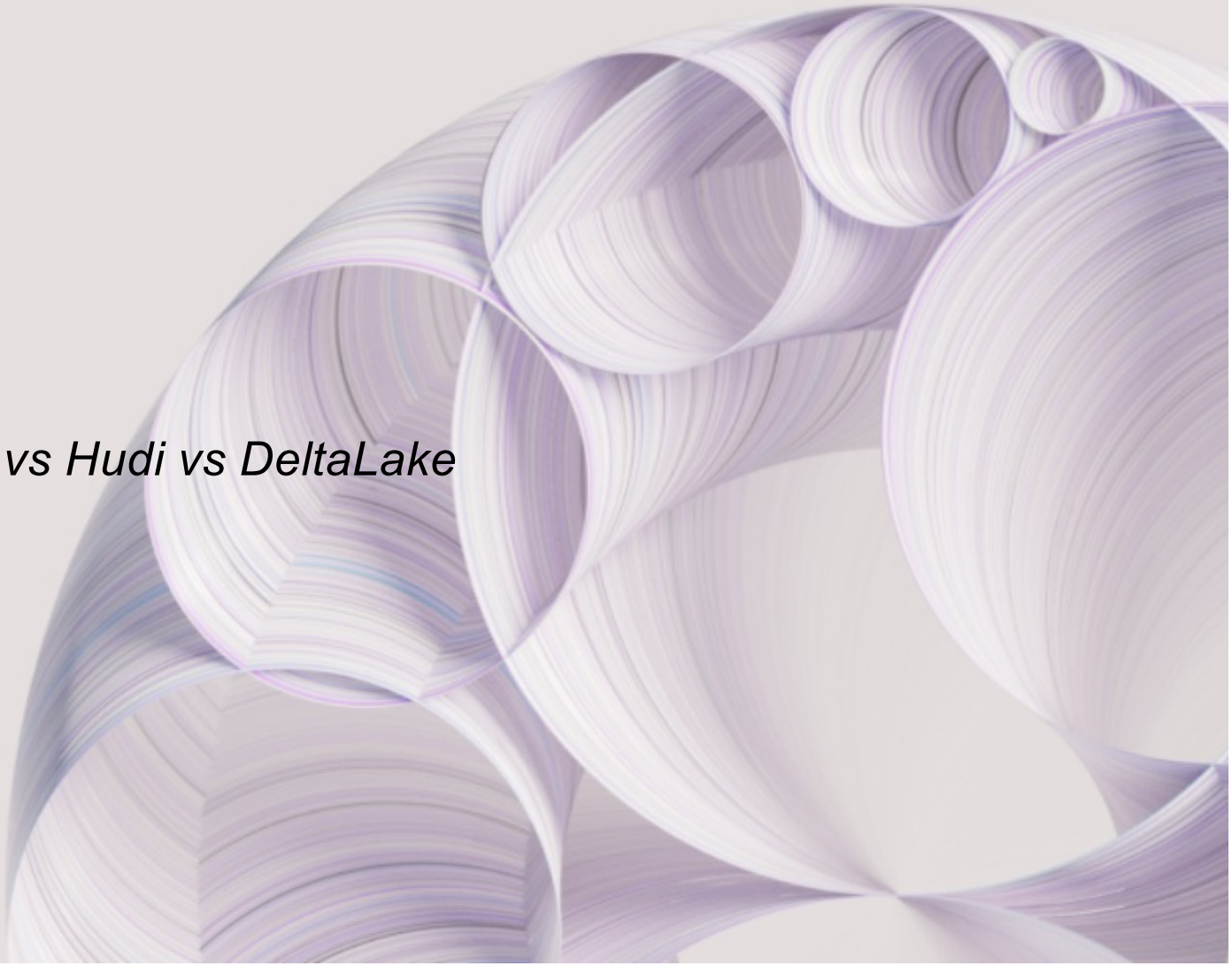
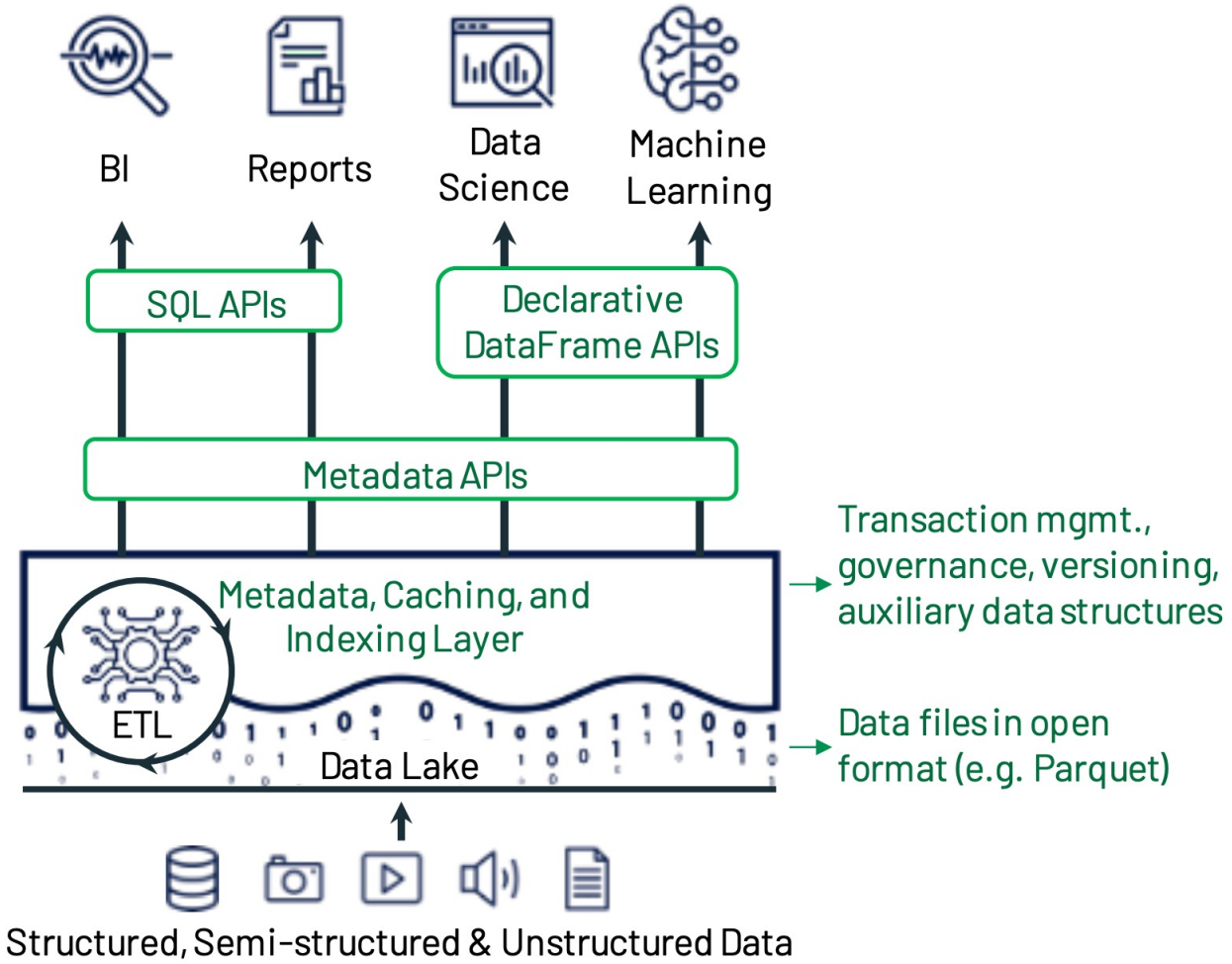


Table Format

Table formats add another metadata layer to allow for data management capabilities.



Open Source Table Formats

- Separation of compute, data, and storage
  1. Leverage low-cost, infinitely scalable object storage
  2. Standardized
     - open file formats (Parquet, ORC, DWRF, JSON, …)
     - table formats (Apache Iceberg, Delta lake, Apache Hudi)
  3. Accessed by scalable compute engines of choice (Presto, Spark, etc.)

open table format for huge analytic datasets optimized for fast query performance

•Schema evolution supports add, drop, update, or rename, and has no side-effects

•Hidden partitioning prevents user mistakes that cause silently incorrect results or extremely slow queries

•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

open table format for huge analytic datasets
optimized for fast query performance

•Schema evolution supports add, drop, update, or rename, and has no side-effects

•Hidden partitioning prevents user mistakes that cause silently incorrect results or extremely slow queries

•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

•Advanced filtering – data files are pruned with partition and column-level stats, using table metadata

- Originally designed to solve correctness problems in eventually-consistent cloud object stores

•Works with any cloud store and reduces NN congestion when in HDFS, by avoiding listing and renames

•Serializable isolation – 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

Incremental data processing framework for low latency minute-level analytics

- Timeline
  - Group all transactions into different types of actions that occur along a timeline
- File Layout
  - Directory-based approach with timestamped files and log files that track changes
- Optional metadata table for additional file pruning
- Table Types
  - COPY_ON_WRITE
  - MERGE_ON_READ
- Query Types
  - Snapshot Queries
  - Incremental Queries
  - Read-Optimized Queries
- Support for create table, insert, update, and delete
- No partition evolution

- ACID transactions on Spark: Serializable isolation levels ensure that readers never see inconsistent data
- Scalable metadata handling: Leverages Spark distributed processing power to handle all the metadata for petabyte-scale tables with billions of files at ease
- Streaming and batch unification: A table in Delta Lake is a batch table as well as a streaming source and sink. Streaming data ingest, batch historic backfill, interactive queries all just work out of the box
- Schema enforcement: Automatically handles schema variations to prevent insertion of bad records during ingestion
- Time travel: Data versioning enables rollbacks, full historical audit trails, and reproducible machine learning experiments
- Upserts and deletes: Supports merge, update and delete operations to enable complex use cases like change-data-capture, slowly-changing-dimension (SCD) operations, streaming upserts, and so on

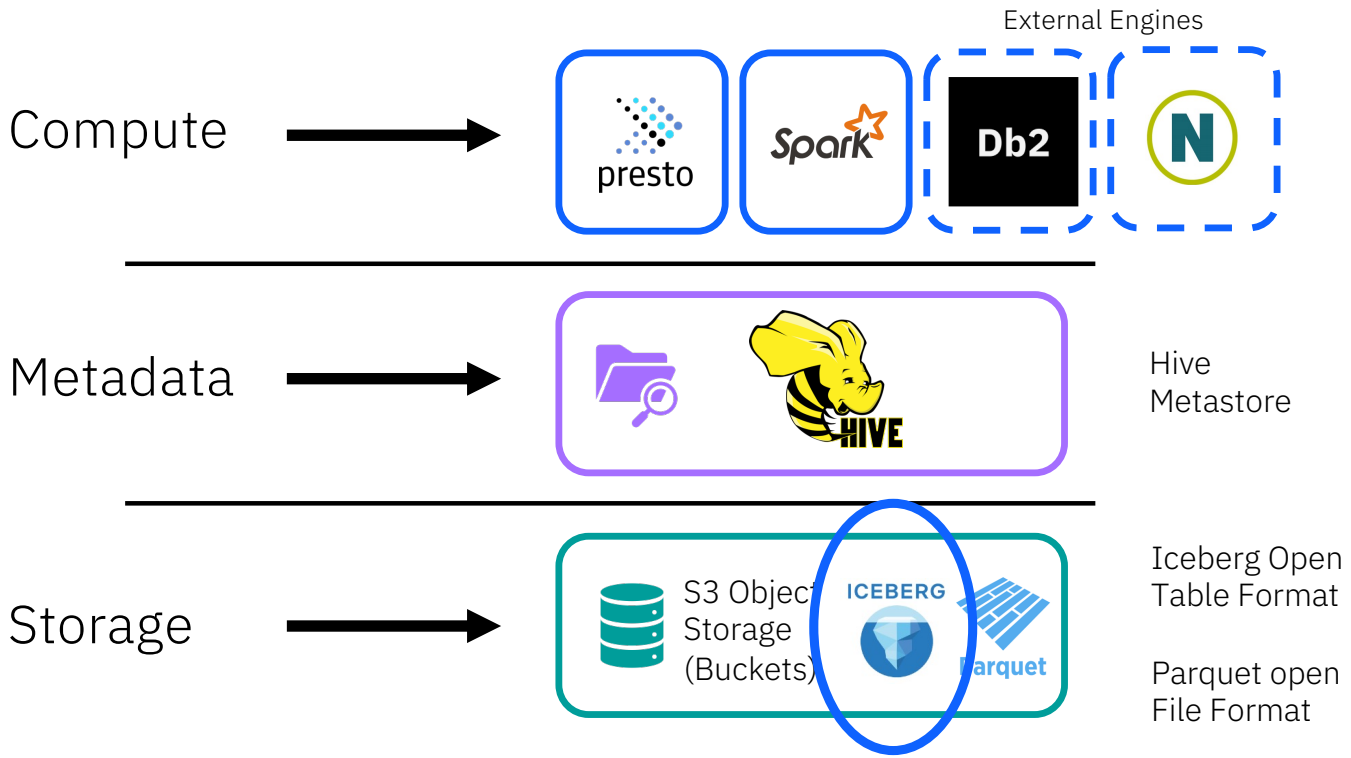
Overall Comparison

- Read/write feature/ingest: Support ACID with OCC
  - Iceberg: partition evolution
  - Hudi: advantage given its origin from streaming ingest, record-level indices (primary keys)
  - Delta: change data feed, delta live tables
- Table Services
  - Iceberg: no indexes; manual invocation
  - Hudi: has more automation
  - Delta: In OSS no indexes & manual services; but at Databricks, managed tables with automation
- Optimistic Concurrency Control
  - Append-only immutable datasets are easier
  - Frequent updates/delete require more fine-grained strategies
- Incremental Pipelines; Ingestion Tools
  - Hudi DeltaStreamer tracks all changes as change streams; supports asynchronous indexing
  - Delta Autoloader (proprietary); Delta has CDF / CDC
  - Iceberg no managed ingestion utility
- Performance
  - by default, Delta + Iceberg are optimized for append-only workloads, while Hudi is by default optimized for mutable workloads
  - Delta and Hudi seem to perform comparable in TPC-DS while Iceberg is slower on Read (2x slower), but similar for write.
- Platform Support: data quality validation, transformers,
  - Iceberg: lacking
  - Hudi: very rich
  - Delta: some manual, but DLT
- Ecosystem:
  - Most vendors support Hudi, except Snowflake and Dremio
  - Good ecosystem support for Delta except Impala, Kafka Connect, and BigQuery

watsonx.data

What is the watsonx.data architecture?

watsonx.data Architecture



What Iceberg is and isn’t

- Table format specification
- A set of APIs and libraries for interaction with that specification
  - These libraries are leveraged in other engines and tools that allow them to interact with iceberg tables

- A storage engine
- An execution engine
- A service

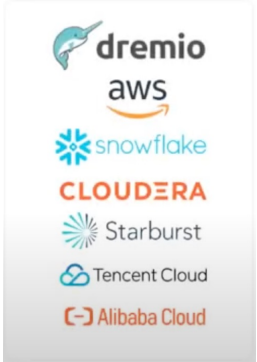
Apache Iceberg – https://iceberg.apache.org

- AI says

> *"Iceberg is an open-source table format that provides transactional, ACID-compliant operations on large datasets. Iceberg is designed to enable efficient queries and data processing on large datasets while providing strong data consistency guarantees. Iceberg supports both batch and streaming workloads and can be integrated with Apache Spark, Presto, and other big data frameworks."*

- Iceberg is
  - An open table format specification that supports huge analytical datasets
  - A set of APIs and libraries for engines to interact with tables following that specification
- Engine-agnostic format understood by different **compute engines**
  - Spark / Dremio / Trino / Presto / Flink / …
- Main features are:
  - Time travel
  - Schema evolution
  - Partition evolution
  - Transactions
  - Performance (pruning using table metadata)

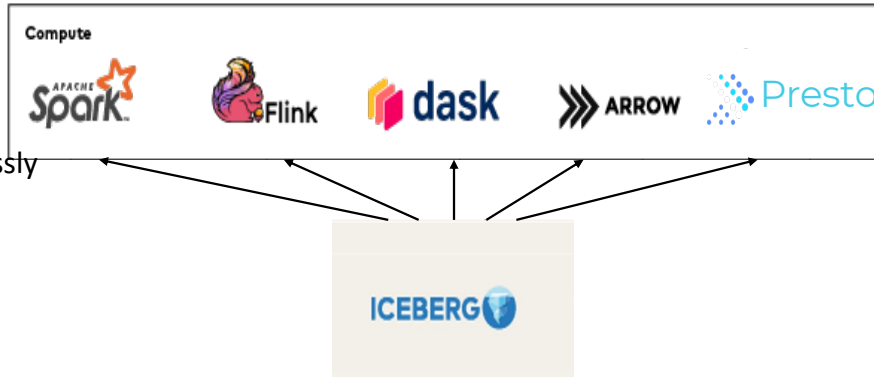
Broad Ecosystem



Community-Led Development

Who is using Iceberg

- Vendors and Applications
  - Lakehouses: watsonx.data, Databricks Lakehouse, Snowflake, Microsoft Azure, AWS, Dremio, Presto, Trino, Google BigLake
  - Data platforms: Spark, Cloudera Data Platform (Hive, Impala)
  - Streaming platforms: Flink, Spark, Apache Beam
  - Applications: Netflix, Adobe, Twitter, Apple, …
    - Apple: several use cases in production, e. g. 2-3 PB of data per table, 2.5 mill files, < 5 sec response time, table management
- Capabilities
  - Flexible compute
    - Don’t move data; multiple engines (many languages, not just JVMs) work seamlessly
  - **Iceberg FileIO (S3FileIO)**
    - **Interface between core Iceberg library and underlying storage**
    - **S3FileIO adopts the latest S3 features for optimized security and performance**
  - Maintaining Iceberg Tables
    - Table migration: In-Place migration or shadow migration to Iceberg
    - Snapshot management
    - Metadata management: expire_snapshots, rewrite_data_files, etc.
    - Compaction (taking several small files and rewrite to fewer larger files to speed up queries), expiring snapshots, remove orphan files

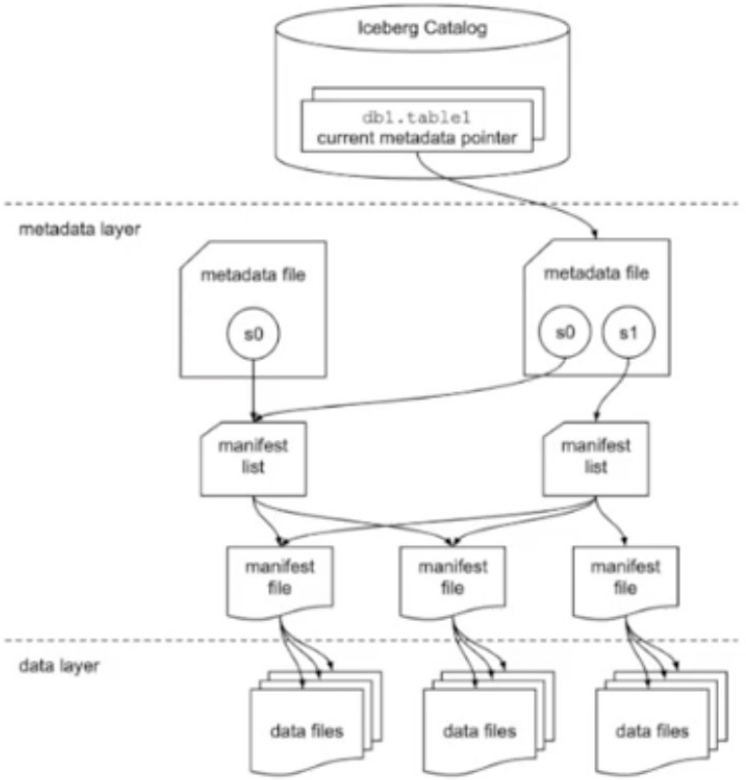


Agenda

- What is Apache Iceberg?
- **Iceberg Architecture**
- Iceberg Metadata
- Migrating to Iceberg
- Time Travel
- Maintenance

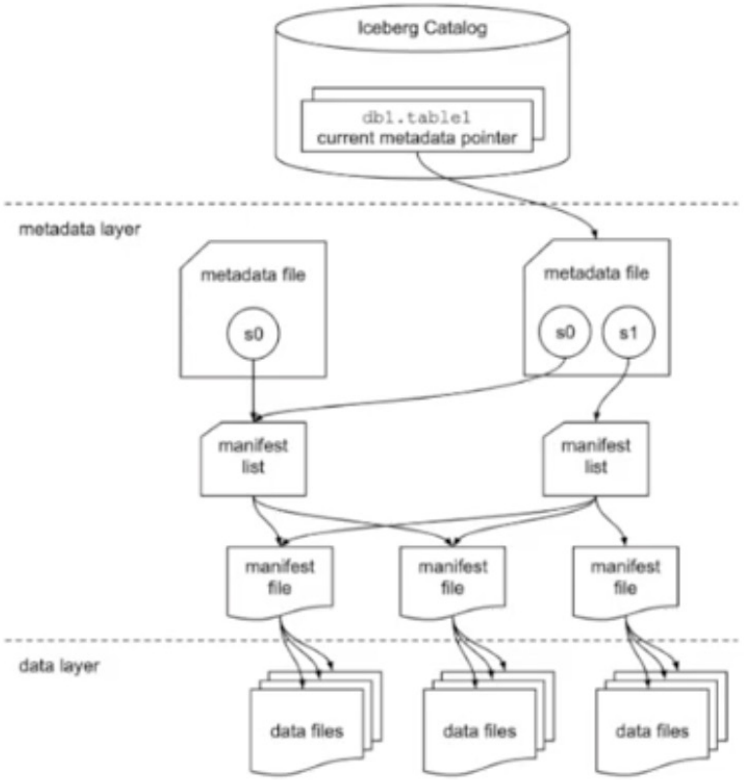
Iceberg Architecture

- Architecture: 3 layers
  - Iceberg Catalog
    - Current metadata pointer for iceberg tables
    - Supports atomic operations for updating the current metadata pointer
  - Three layers of Metadata
    - Metadata defines the table
    - Manifest lists define snapshots with list of manifests
    - Manifests have the metadata about the data (min/max values)
  - Data Layer
    - Data in data formats

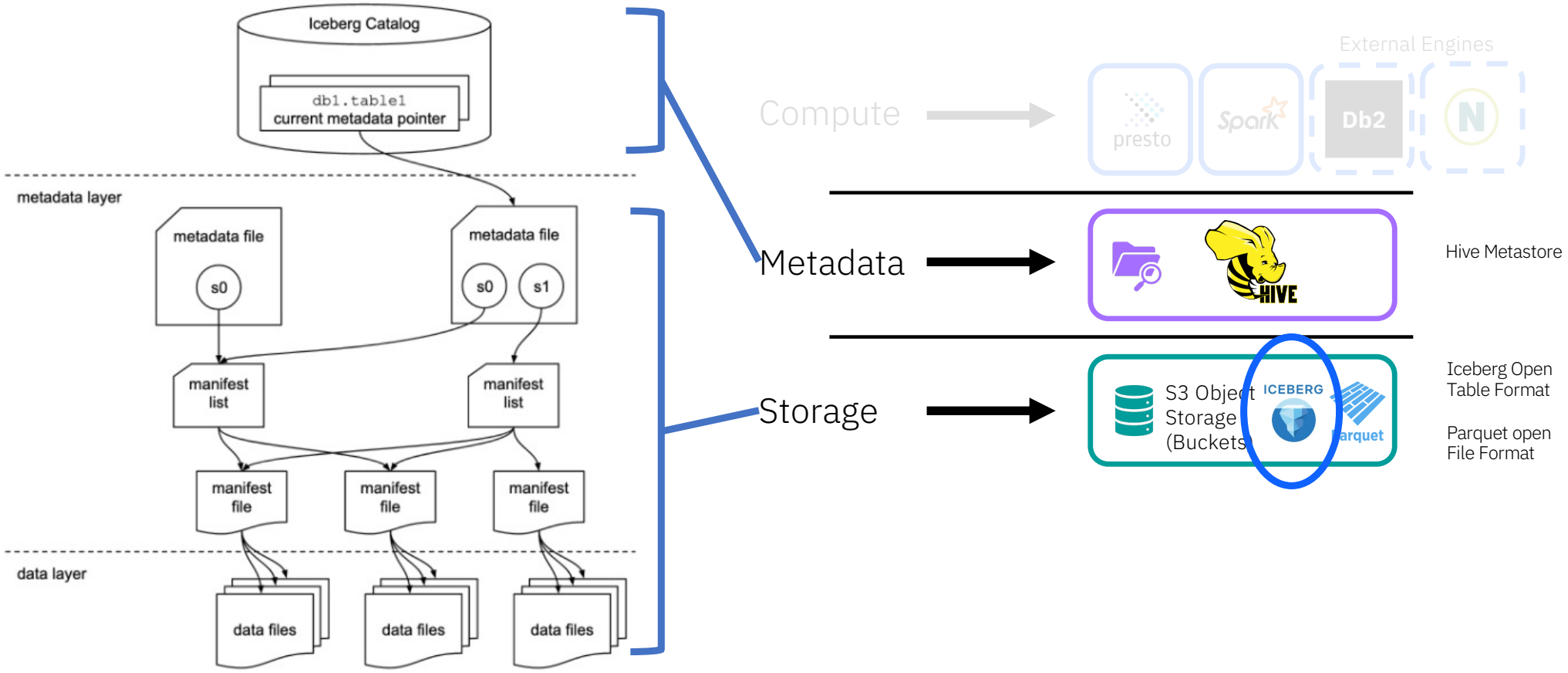


Iceberg Architecture

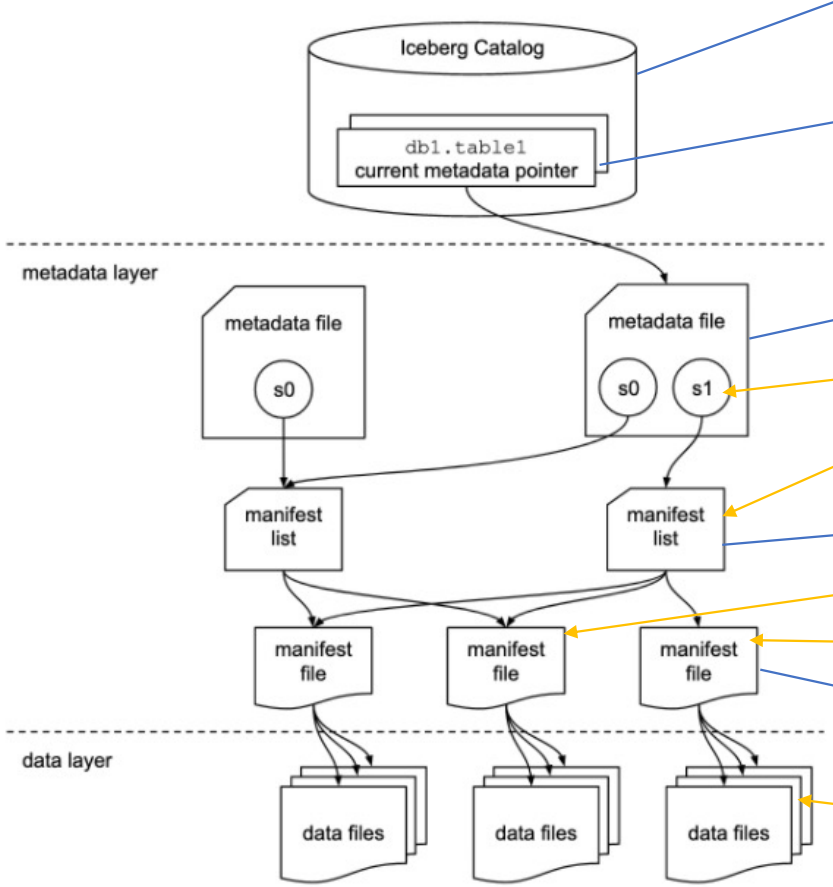
- Architecture: 3 layers
  - Iceberg Catalog
    - Current metadata pointer for iceberg tables
    - Supports atomic operations for updating the current metadata pointer
  - Three layers of Metadata
    - Metadata defines the table
    - Manifest lists define snapshots with list of manifests
    - Manifests have the metadata about the data (min/max values)
  - Data Layer
    - Data in data formats
- Benefits, Capabilities, and Observations
  - Table abstraction enables physical optimizations (hidden partitioning, compaction, evolution)
  - Snapshot isolation through metadata tree
  - Optimistic concurrency control
  - Faster query planning and execution, at scale
  - Any write to a table creates a new snapshot
  - Support for create, insert, merge, row-level updates, and deletes



Iceberg Table Architecture – watsonx.data



Iceberg Table Architecture



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

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>}, ...],
4 "current-snapshot-id": <snapshot-id>,
  "snapshots": [ {
      "snapshot-id": <snapshot-id>
5     "manifest-list": "/path/to/manifest/list.avro"
    }, ...],
  ...
}
```

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}, ...],
  ...
}
```

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": {"<part-field>":{"<data-type>":<value>}},
    "record-count": <num-records>,
    "null-value-counts": {{
      "column-index": "1", "value": 4
    }, ...},
    "lower-bounds": {{
      "column-index": "1", "value": "aaa"
```

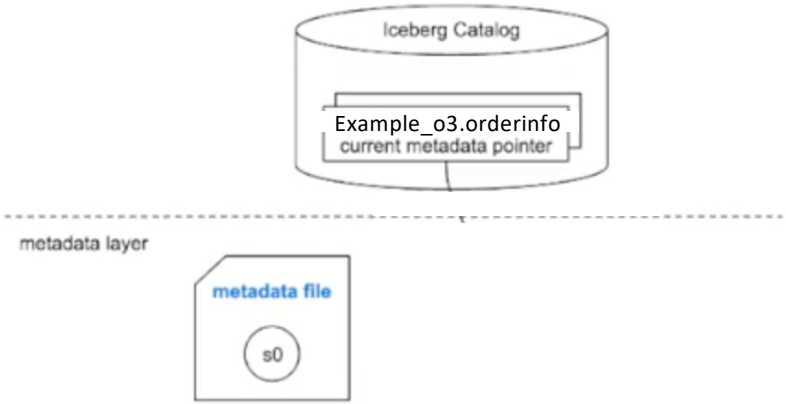
Create a Schema
(in watsonx.data)

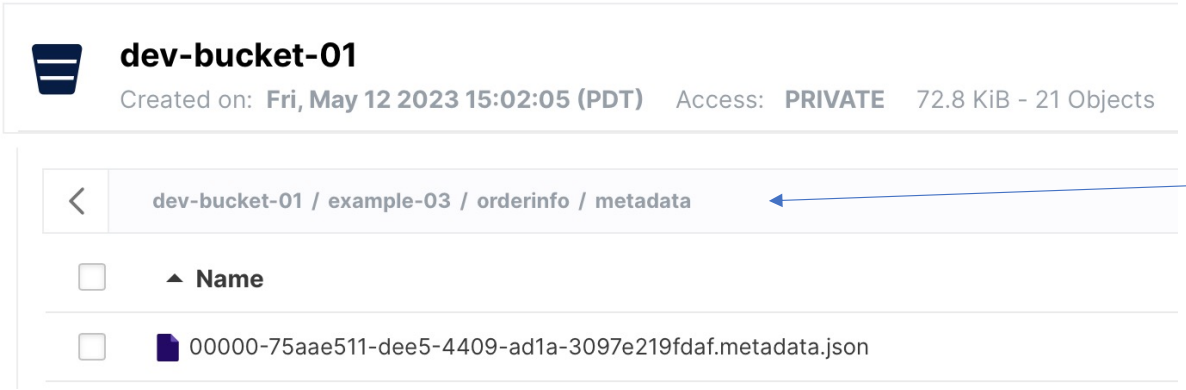
```
create schema "iceberg_minio"."example_03” with
(location='s3a://dev-bucket-01/example-03');
```

Creates the example-03 bucket under “dev-bucket1-01”

Create an Iceberg Table

```
create table "iceberg_minio"."example_03".orderinfo(
  order_id int,
  customer_id int,
  order_amt decimal(10,2)
)
```

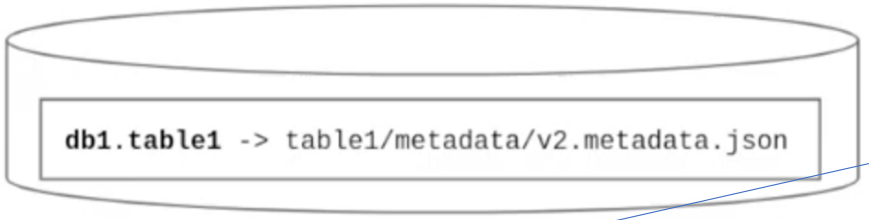




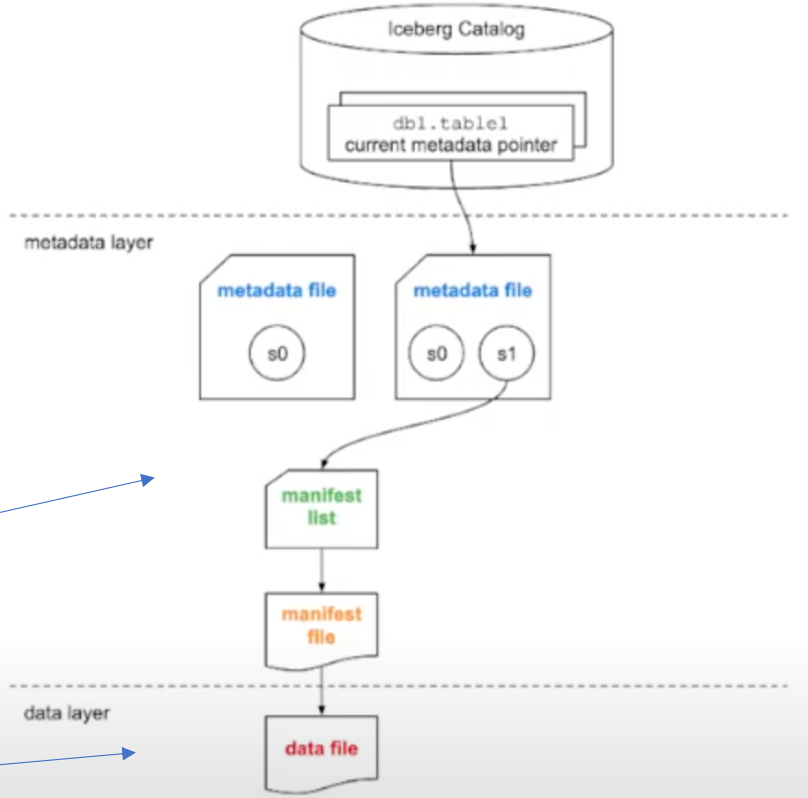
Creates the “orderinfo” bucket with 2 sub-buckets – “data” and “metadata”

Insert into Iceberg Table

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

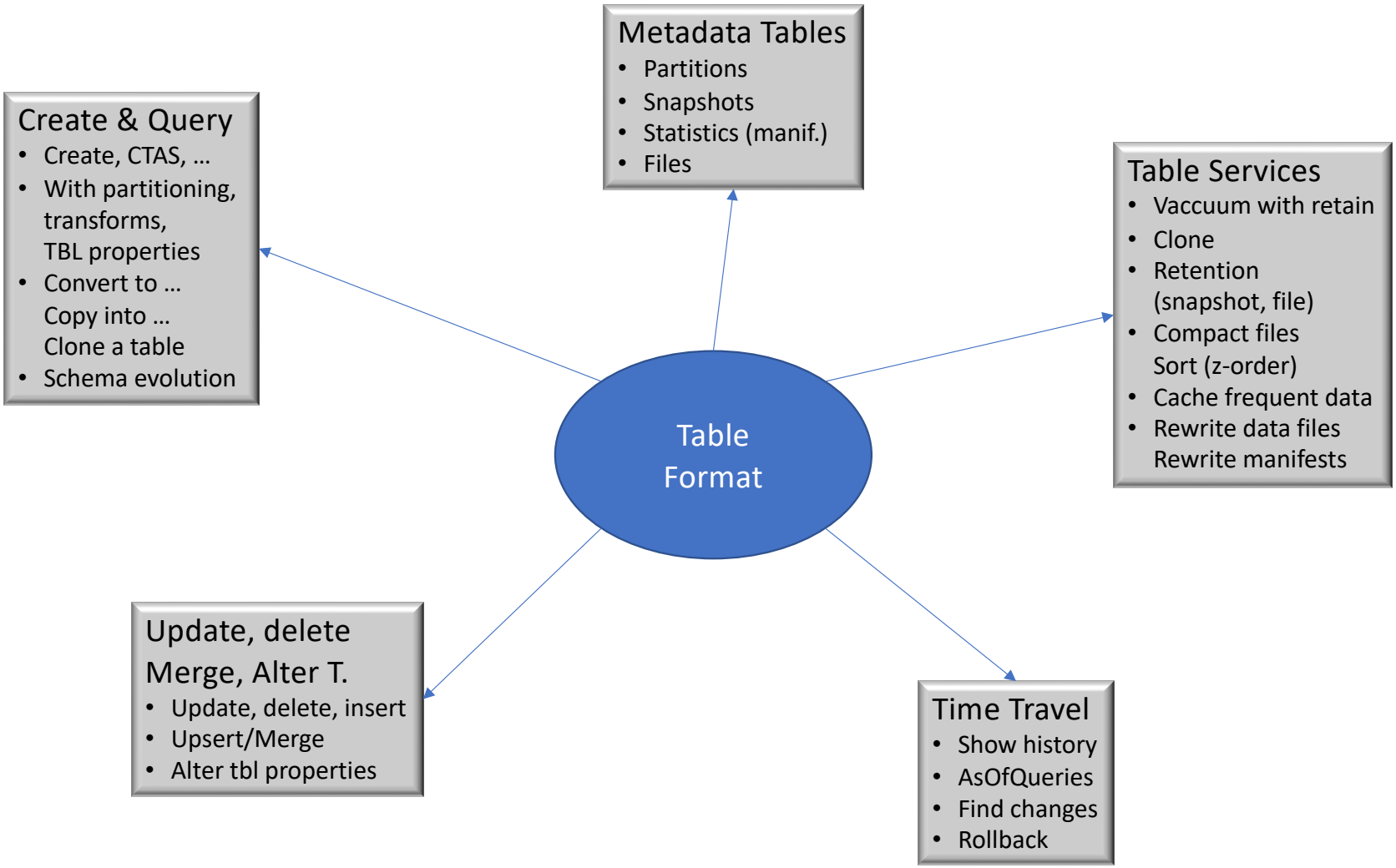



```
table1/
 |- metadata/
 |    |- v1.metadata.json
 |    |- v2.metadata.json
 |    |- snap-2938-1-4103.avro
 |    |- d8f9-ad19-4e.avro
 |- data/
 |    |- order_ts_hour=2021-01-26-08/
 |    |    |- 00000-5-cae2d.parquet
```



Agenda

- What is Apache Iceberg?
- Iceberg Architecture
- **Iceberg Metadata**
- Migrating to Iceberg
- Time Travel
- Maintenance


Metadata Tables
• Partitions
• Snapshots
• Statistics (manif.)
• Files
Create & Query
• Create, CTAS, …
• With partitioning, transforms, TBL properties
• Convert to …
Copy into …
Clone a table
• Schema evolution
Table Services
• Vaccuum with retain
• Clone
• Retention (snapshot, file)
• Compact files
Sort (z-order)
• Cache frequent data
• Rewrite data files
Rewrite manifests
Table Format
Update, delete
Merge, Alter T.
• Update, delete, insert
• Upsert/Merge
• Alter tbl properties
Time Travel
• Show history
• AsOfQueries
• Find changes
• Rollback

Inspecting Presto Iceberg Tables (0.279)

```
presto> describe "store_sales$files";

       Column       |          Type           |
--------------------+-------------------------+
 content            | integer                 |
 file_path          | varchar                 |
 file_format        | varchar                 |
 record_count       | bigint                  |
 file_size_in_bytes | bigint                  |
 column_sizes       | map(integer, bigint)    |
 value_counts       | map(integer, bigint)    |
 null_value_counts  | map(integer, bigint)    |
 nan_value_counts   | map(integer, bigint)    |
 lower_bounds       | map(integer, varchar)   |
 upper_bounds       | map(integer, varchar)   |
 key_metadata       | varbinary               |
 split_offsets      | array(bigint)           |
 equality_ids       | array(integer)          |
(14 rows)
```

```
presto> describe store_sales;

        Column     |     Type|
-------------------+---------+
 ss_sold_date_sk   | integer |
 ss_sold_time_sk   | integer |
 ss_customer_sk    | integer |
  ...
(23 rows)
```

```
presto> describe "store_sales$partitions";

        Column          |         Type                |
------------------------+-----------------------------+
 ss_sold_date_sk        | integer                     |
 row_count              | bigint                      |
 file_count             | bigint                      |
 total_size             | bigint                      |
 ss_sold_time_sk        | row("min" integer,          |
                        |     "max" integer,          |
                        |     "null_count" bigint)    |
 ss_item_sk             | row("min" integer,          |
                        |     "max" integer,          |
                        |     "null_count" bigint)    |
...
(26 rows)
```

```
presto> describe "store_sales$history";

       Column         |           Type           |
----------------------+--------------------------+
 made_current_at      | timestamp with time zone |
 snapshot_id          | bigint                   |
 parent_id            | bigint                   |
 is_current_ancestor  | boolean                  |
(4 rows)
```

```
presto> describe "store_sales$manifests";

          Column            |           Type                      |
----------------------------+-------------------------------------+
 path                       | varchar                             |
 length                     | bigint                              |
 partition_spec_id          | integer                             |
 added_snapshot_id          | bigint                              |
 added_data_files_count     | integer                             |
 existing_data_files_count  | integer                             |
 deleted_data_files_count   | integer                             |
 partitions                 | array(row("contains_null" boolean,  |
                            |           "lower_bound" varchar,    |
                            |           "upper_bound" varchar))   |
(8 rows)
```

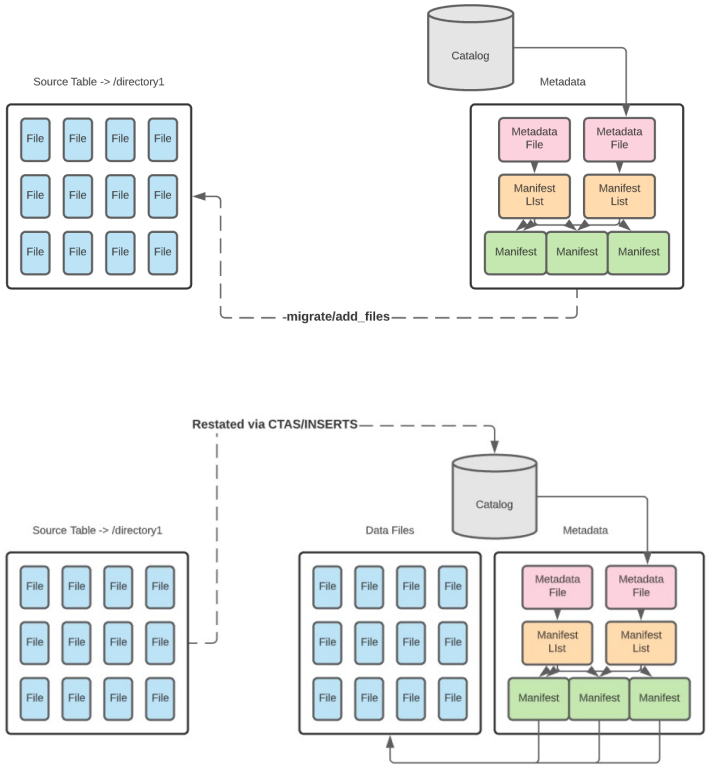
```
presto> describe "store_sales$snapshots";

    Column      |           Type           |
----------------+--------------------------+
 committed_at   | timestamp with time zone |
 snapshot_id    | bigint                   |
 parent_id      | bigint                   |
 operation      | varchar                  |
 manifest_list  | varchar                  |
 summary        | map(varchar, varchar)    |
(6 rows)
```

Agenda

- What is Apache Iceberg?
- Iceberg Architecture
- Iceberg Metadata
- Migrating to Iceberg
- Time Travel
- Maintenance

Migrate from Hive format to Iceberg

- In-Place Migration
  - Use Iceberg procedures built into SPARK (as of watsonx.data GA)
  - Pros
    - Less time-consuming; Only write Iceberg metadata;
  - Cons
    - Add new data during metadata write; Won't work if data needs to be restated
    - Possible sub-optimal performance without re-partitioning
- Shadow Migration
  - Creates Iceberg metadata and restate all data files as main table
  - Pros
    - Audit and validate the data; Apply schema and partition changes; Data corruption unlikely
  - Cons
    - Double storage (temporarily); Takes long time



https://iceberg.apache.org/docs/latest/spark-procedures/#table-migration

Time Travel

- Use cases are repetitive ML training on stable dataset, data quality assessment as-of, etc.
- In Presto via table snapshots: *$snapshot* metadata table

```
presto> insert into customer (custkey, name) values (1111, 'Mr. Presto');
presto> select custkey, name from customer where custkey < 883908;
custkey |        name
---------+--------------------
    1111 | Mr. Presto
  883906 | Customer#000883906
  883907 | Customer#000883907

presto> select * from iceberg.bjr."customer$snapshots" order by committed_at asc;
        committed_at         |     snapshot_id     |      parent_id      | operation |                                                                  manifest_list
-----------------------------+---------------------+---------------------+-----------+-----------------------------------------------------------------------------------------------------------------------------------------
 2023-03-12 21:35:12.214 UTC | 7602905952653382334 | NULL                | append    | file:/data/presto/bjr.db/customer/metadata/snap-7602905952653382334-1-30a305ba-1b5e-4241-88ca-73d88dd00df5.avro
 2023-03-12 21:39:25.069 UTC | 1261483210916897774 | 7602905952653382334 | append    | file:/data/presto/bjr.db/customer/metadata/snap-1261483210916897774-1-8d611d50-7db4-43b7-8c33-ab2dcd034fb3.avro

presto> call iceberg.system.rollback_to_snapshot('bjr','customer',7602905952653382334);
presto> select custkey, name from customer where custkey < 883908;
custkey |        name
---------+--------------------
  883906 | Customer#000883906
  883907 | Customer#000883907

presto> insert into customer (custkey, name) values (1112, 'Mrs. Presto');
presto> select custkey, name from customer where custkey < 883908;
custkey |        name
---------+--------------------
    1112 | Mrs. Presto
  883906 | Customer#000883906
  883907 | Customer#000883907

presto> select * from "customer$snapshots" order by committed_at asc;
        committed_at         |     snapshot_id     |      parent_id      | operation |                                                                  manifest_list
-----------------------------+---------------------+---------------------+-----------+-----------------------------------------------------------------------------------------------------------------------------------------
 2023-03-12 21:35:12.214 UTC | 7602905952653382334 | NULL                | append    | file:/data/presto/bjr.db/customer/metadata/snap-7602905952653382334-1-30a305ba-1b5e-4241-88ca-73d88dd00df5.avro
 2023-03-12 21:39:25.069 UTC | 1261483210916897774 | 7602905952653382334 | append    | file:/data/presto/bjr.db/customer/metadata/snap-1261483210916897774-1-8d611d50-7db4-43b7-8c33-ab2dcd034fb3.avro
 2023-03-12 22:15:13.523 UTC | 1651844482013862977 | 7602905952653382334 | append    | file:/data/presto/bjr.db/customer/metadata/snap-1651844482013862977-1-720334fa-ee8a-48aa-b266-32a02458d9d5.avro
```

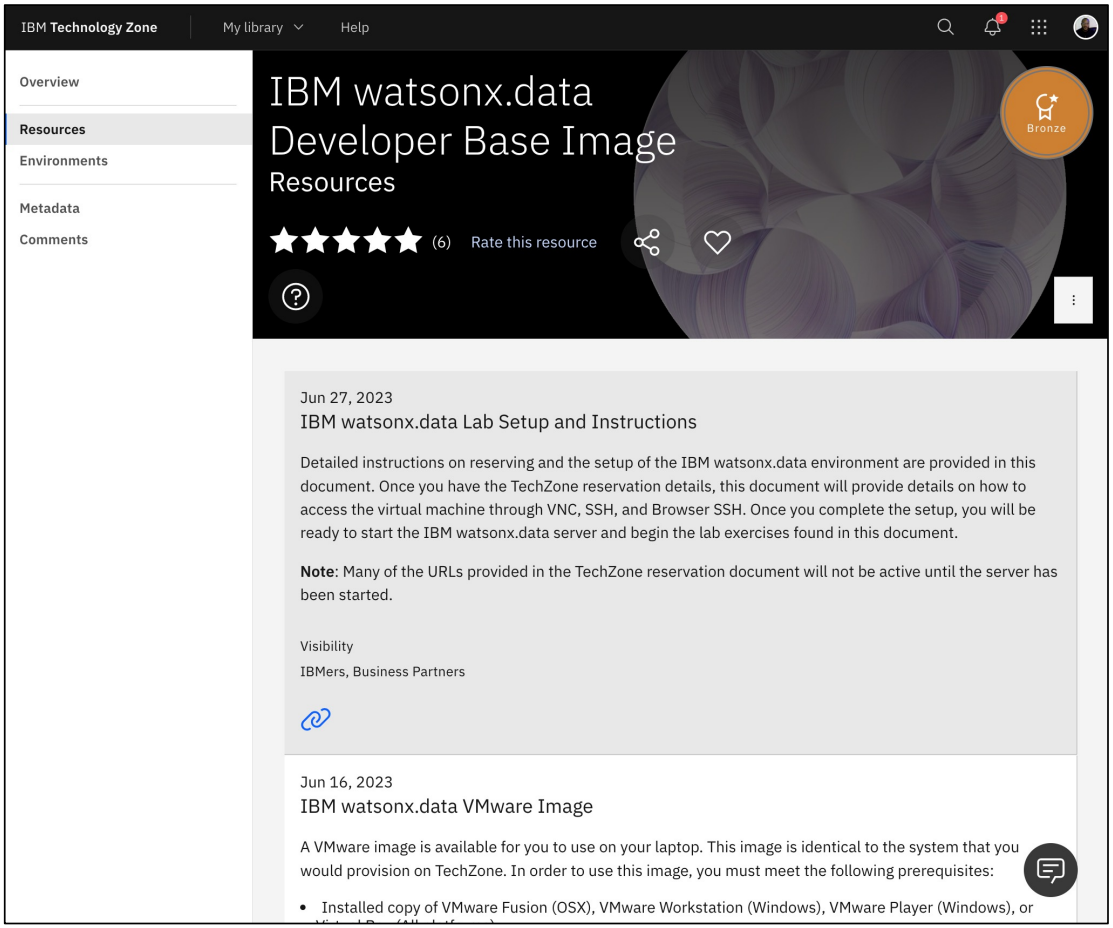
- Iceberg Spark Procedures for Snapshot management
  - rollback_to_snapshot(), rollback_to_timestamp(), set_current_snapshot (), cherrypick_snapshot
- At GA – watsonx.data presto client (and UI) supports rollback to snapshot only. More on roadmap.

Environment Setup
Installation and configuration of WatsonX.data

1. TechZone instance overview
2. installation on virtual machine
3. by downloading ova file from TechZone instance
4. By using SaaS

Try watsonx.data yourself!

- Deploy locally using VMware Image
  - VMWare Image
- Deploy via TechZone
  - Visit watsonx.data Dev TechZone page and provision environment
  - NOTE: If environment provision request is denied, try again after a few hours. TechZone may be overloaded.
- Environment setup instructions
  - ibm.biz/wxd-setup
- Lab Instructions
  - ibm.biz/wxd-lab
- watsonx.data Documentation
  - https://www.ibm.com/docs/en/watsonxdata/1.0.x
- IBM Data Lakehouse Technology Pattern
  - https://ibm.biz/DataLakehousePattern
- Get assistance & connect!
  - #ce-watsonx-Americas
  - #technology-patterns-lakehouse
  - #watsonx-data-lh-content-productdocumentation
  - #watsonx-data-techbites (private channel – request access)
  - Watsonx.data Seismic




QUIZ TIME

DAY 4

Sample Discovery Questions

- What type of database are they using today
- Is cost an issue or starting to become an issue
- What are the use cases and how do they serve the business
- What are the required SLAs? Who are the users accessing? How fast do they need answers?
- Are there ETL workloads within the EDW, what are the time windows
- Is there historical data in the EDW that is not frequently accessed but still required to be there?
- Overall data size and instance size

IBM Cross Sell

What do I sell if my customer has:

- IBM analytics appliance?
- IBM Db2 Warehouse or Netezza?
- Db2 for z/OS?

IBM Modernize + Cross Sell

Customer Goal: Customer has an IBM on-premises solution and want to move to next generation of self-managed and SaaS offerings

Customer motivation:

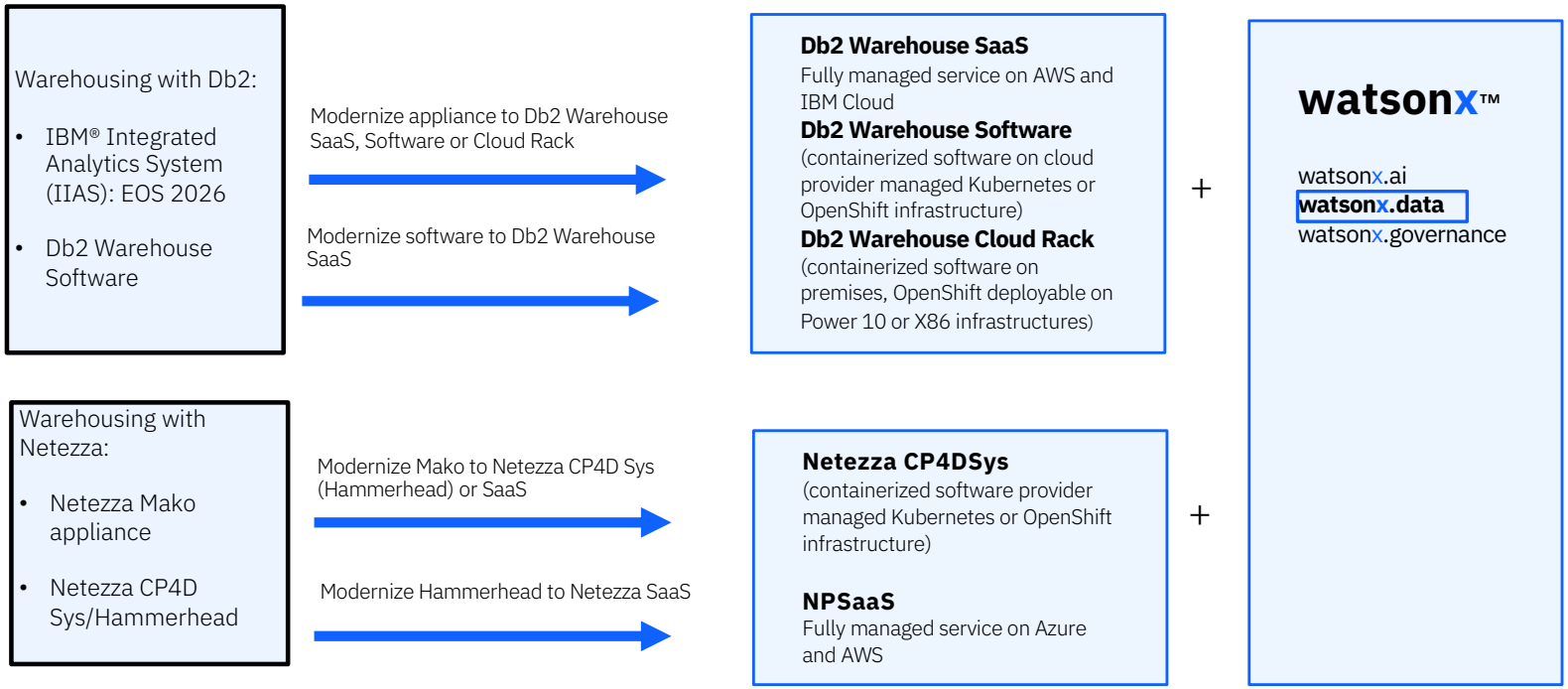
- Looking for next generation features
- Limited compute or data storage resulting in offloading or expanding data infrastructure
- Difficulty scaling up and down for workloads
- New use cases: AI/ML, Real-time analytics, data engineering, data sharing

Challenges to Address:

- System migration compatibility
- Sizing management
- Cost controls on the cloud
- Access to all data across hybrid cloud without duplication/movement

Key Messages:

1. Modernize your database appliance with like for like compatibility and support new use cases
2. Optimize workloads for AI with fit for purpose engines and reduce warehouse costs by 50%, access all governed data across hybrid cloud



IBM Db2 for z/OS Cross-sell

Customer Goal: Customer has IBM Db2 for z/OS and wants to access transactional data from mainframe for AI use cases

Customer motivation:

- Brand new use cases with mainframe transactional data: AI/ML, Real-time analytics, data engineering, data sharing

Challenges to Address:

- System migration compatibility

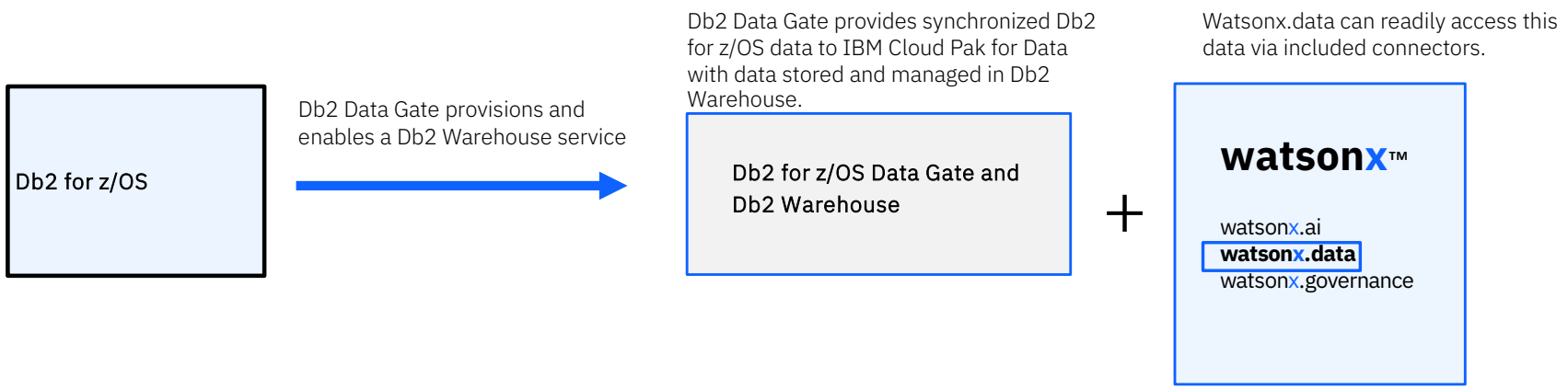
Key Messages z/OS and Db2 Warehouse:

- An integrated, optimized synchronization feature maintains currency between source Db2 for z/OS data on IBM Z and Db2 Warehouse targets
- Db2 Warehouse is a relational databases that delivers advanced data management and analytics capabilities

Key Messages z/OS and .data:

- Access and analyze data from mainframe systems in near-real time
- Consume less processor capacity with built-in synchronization
- Use the most up-to-date data from mainframe for machine learning models

---



Concepts and market entry points

Three key concepts for IBM watsonx.data

1. Presto is a next-generation open-source SQL engine designed to run efficiently over data lakes.

2. Warehouses and first-generation lakehouses are monolithic, and not optimized to work on all workloads. Only IBM watsonx.data's multi-engine architecture allows for true workload optimization.

3. Iceberg is an open-table format that allows multiple engines to access the same data – this means, Snowflake, Netezza, and IBM watsonx.data can all access data in Iceberg at the same time.

Market entry points + use cases

1. Warehouse Optimization narrative

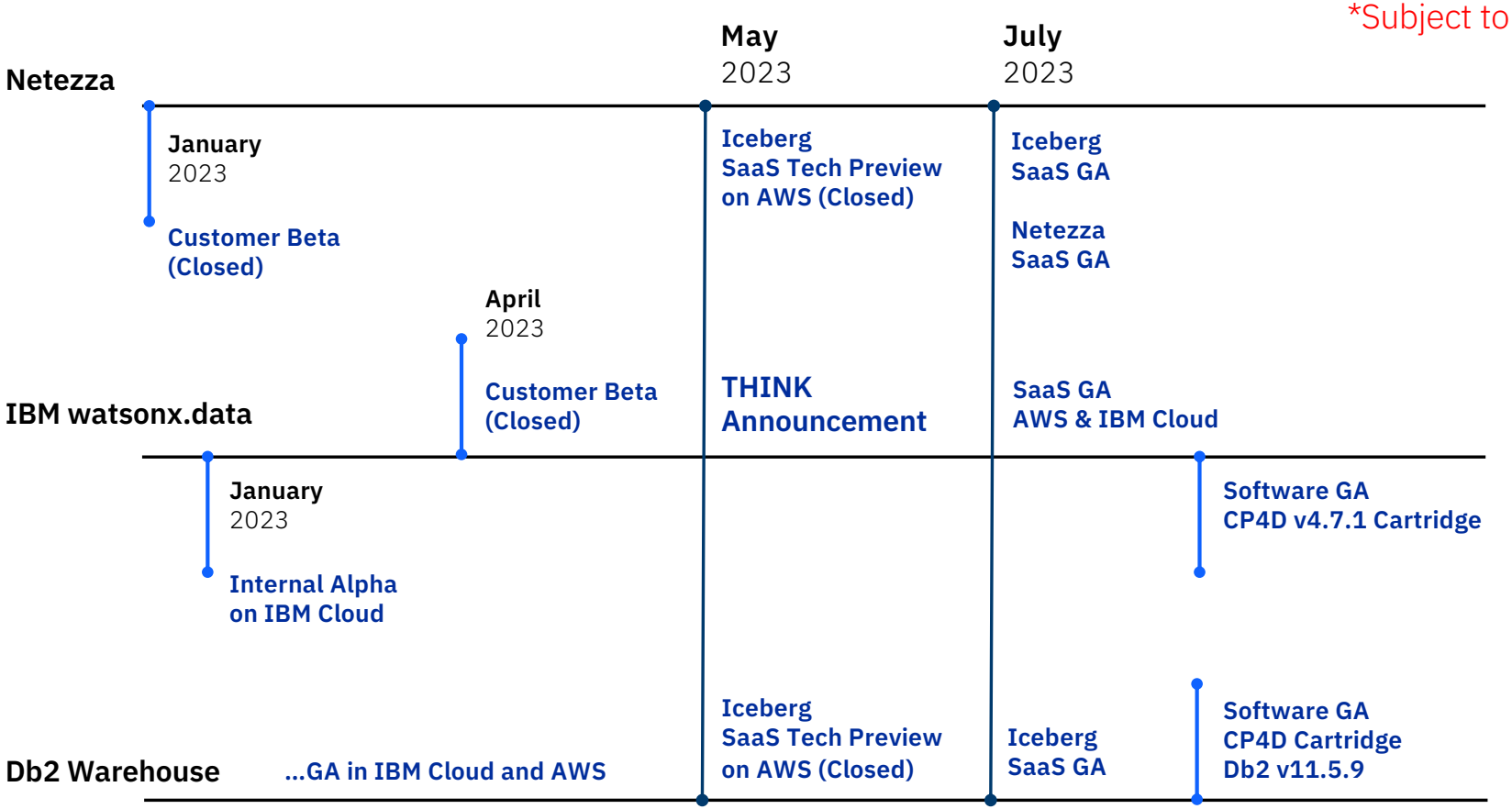
- Competitive and IBM (Netezza + Db2)
- Talk track – Client is concerned with the spend on traditional warehouse today – looking to optimize for both performance and cost
- **Value prop:** Cost optimization and openness through the shared meta layer and fit-for-purpose engines
- Use case – Snowflake write-intensive workloads moving to Spark and/or Presto, thus reducing cost of Snowflake virtual data warehouses

2. Modernizing data lake narrative

- Modernizing storage architecture to facilitate shared metadata and fit-for-purpose engines
- Talk track – Converting legacy file storage structures into open-file structures and assigning those into the shared meta layer, thus facilitating fit-for-purpose engines
- Value prop: Modernize with warehouse-like performance for querying data in open formats, built-in governance
- Use case – Move from HDFS to open-source iceberg within a consolidated shared meta layer

IBM watsonx.data – High-level milestones & timeline

*Subject to change

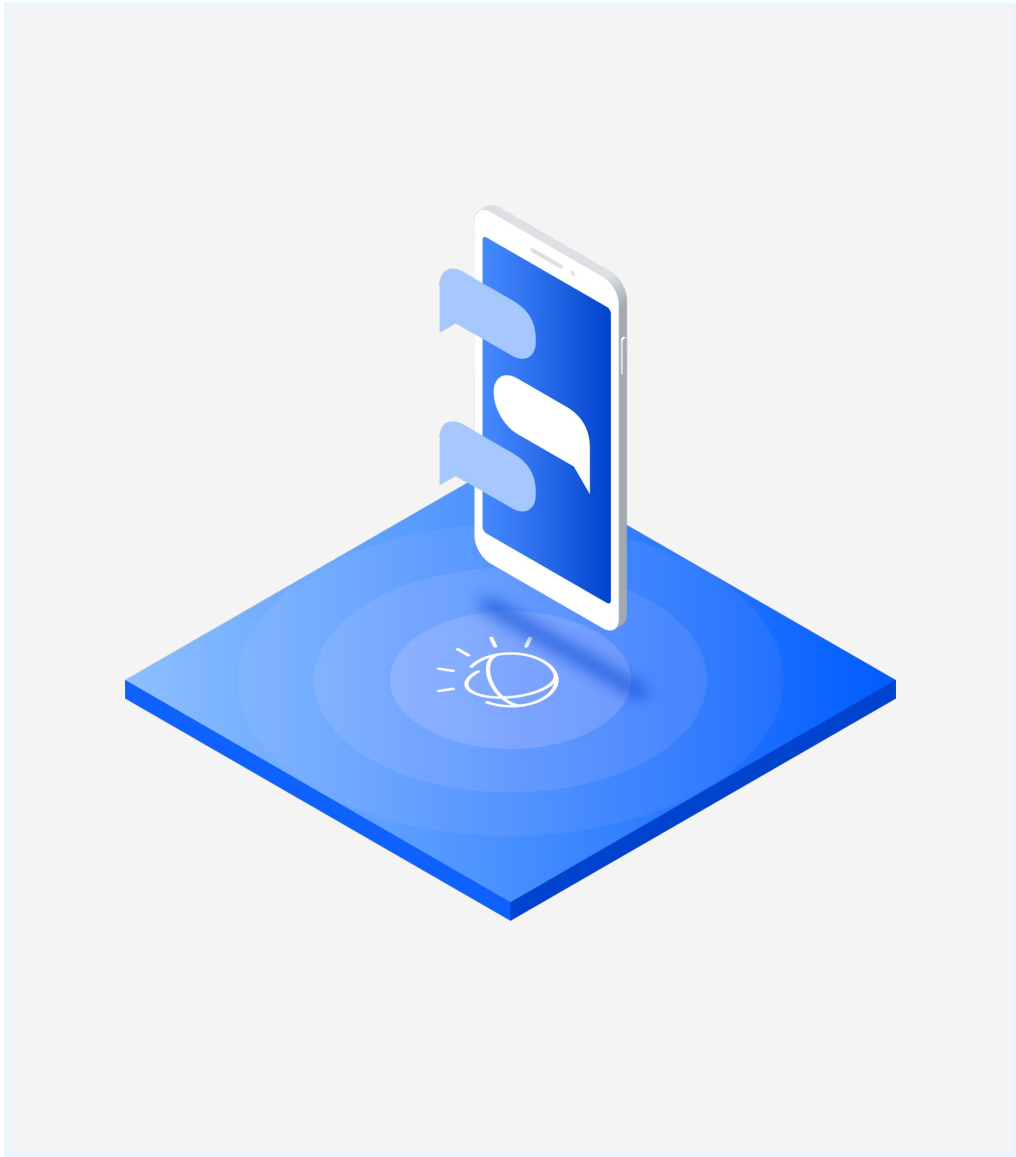


Competitors and objection handling

IBM **watsonx.data** competitive insights

- Learn more about the competition
- Provide feedback
- Request help with an opportunity

Get the presentation on Seismic →



Types of competitors

A data lakehouse combines the best features of data warehouses and data lakes to provide cost optimization for clients. Compute and storage are separated so that data can be accessed from different engines.

An augmented data warehouse may be able to access different types of data, but all compute processing is performed through the data warehouse engine. There is no compute cost optimization allowing different engines, only the compute amount used by the data warehouse engine can be adjusted.

Data lakehouse competitors

- Designed for both structured and unstructured data
- Based on open table and data formats for data storage
- Fit for purpose query engine for different use cases
- Cost optimization for compute engine and storage
- Separate compute and storage

Augmented data warehouse competitors

- Primarily designed for structured and semi-structured data
- Uses proprietary or open file formats, but supports open table format for data storage
- Query processing uses data warehouse engine
- Compute and storage are not completely separated

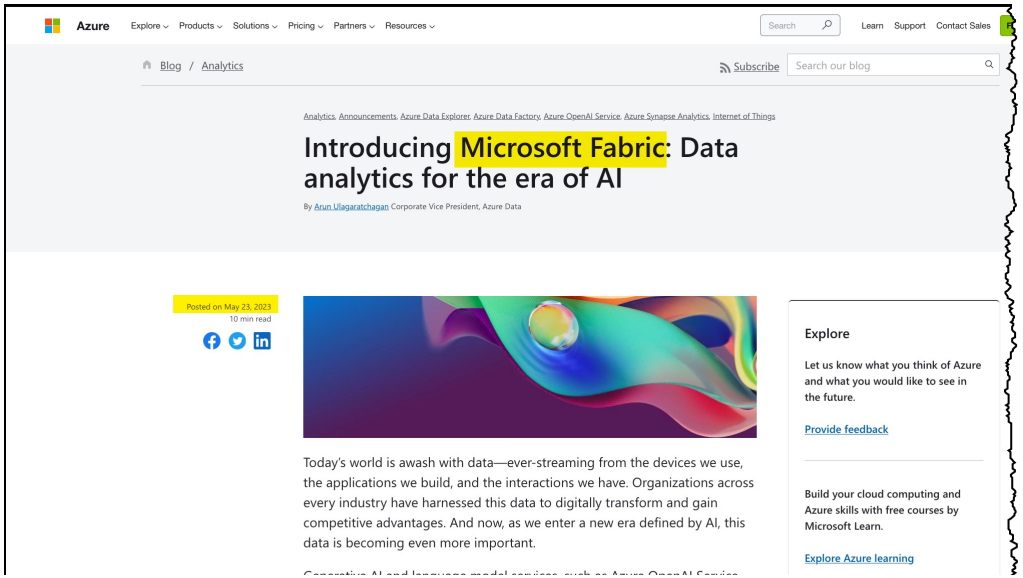
watsonx.data

Competitor overview (weaknesses not mentioned)

All competitors are relatively new companies (within the past decade) and are rapidly growing in the public cloud market

	Competitor	Overview
Data Lakehouse Competitors	Databricks	Databricks pioneered the term “Lakehouse”. It’s currently positioned as a leader in the emerging Lakehouse market. Founded in 2013 by the creators of Apache Spark, Databricks offers a unified analytics platform for a variety of use-cases such as data engineering, machine learning, data science, and AI.
	Dremio	Dremio has gained significant recognition for its proprietary Dremio query engine (Sonar) technology. It is positioned as a leader in the data lakehouse market, with a growing market share. Founded in 2015, Dremio offers an open, cloud-native data lakehouse engine that simplifies and accelerates data processing and analytics.
	Starburst	Starburst has made a name for itself in the data access and analytics space. It is positioned as a leader in the enterprise data access market, with a growing market share. Founded in 2017, Starburst offers a cloud-native platform that enables fast and easy access to data across a range of sources.
	Amazon Athena	Amazon Athena was first released in 2016 and is the AWS data lakehouse offering that utilizes Apache Spark for analytics on data in open file formats and the Trino engine for SQL queries. Amazon Athena combines with other AWS services, like AWS Lake Formation, for data governance to build a complete lakehouse solution.
Others	Snowflake	Founded in 2012, Snowflake has made significant strides in the cloud data warehousing market and is currently positioned as a leader in the cloud data platform space. Snowflake supports open table formats but locks clients into the Snowflake environment that is locked and controlled (not open-source based). It has a single SQL query engine and a limited ability to access data outside of a Snowflake data warehouse, and no hybrid cloud capability.
	Amazon Redshift Spectrum	Amazon Redshift Spectrum is a Redshift service that allows direct queries on data stored in Amazon S3 files without having to load the data into an Amazon Redshift data warehouse. Amazon Redshift Spectrum requires an active Amazon Redshift data warehouse cluster to execute queries, so it is tightly integrated with Amazon Redshift and extends the data warehouse to access external tables in Amazon S3.

watsonx.data
Breaking news



May 23, 2023 – Microsoft announces Microsoft Fabric

Points to remember

- Microsoft Fabric is only in PREVIEW
- Microsoft Fabric is only available on Microsoft Azure
- Microsoft Fabric is more than a data lakehouse, it is a data analytics platform

IBM discussion points

- **watsonx.data** competes directly with Azure Databricks
- Microsoft Fabric competes with the overall **watson**x platform
- PREVIEW items are NOT available for production use

Source: Introducing Microsoft Fabric: Data analytics for the era of AI blog

Response to Databricks Announcement June 2023

Databricks recently made several announcements at their Data + AI Summit (June 26-29, 2023), this document provides a summary of these new features and capabilities along with competitive analysis from an IBM watsonx.data perspective.

The key news was in the areas of Delta Lake and Unity Catalog. Each announcement will contain a technical overview of the feature or capability, its status (including a tentative release data if available), and a competitive analysis compared to watsonx.data.

Review the response on Seismic →

Competitive landscape

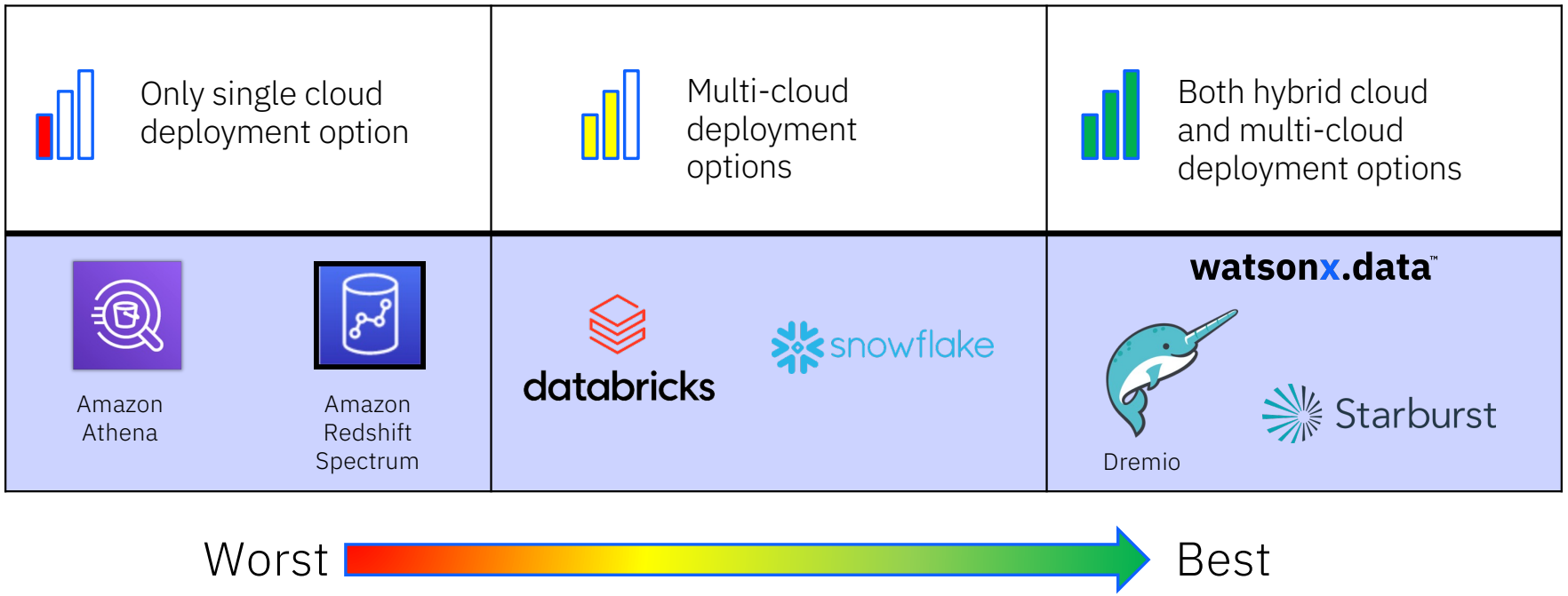
Data lakehouse competitors

Details	databricks		Starburst	
Deployment options	Public cloud only	Public cloud & on-premises	Public cloud & on-premises	AWS only
Query engines	• Apache Spark • Photon	• Dremio Sonar (proprietary)	• Starburst (Trino-based)	• Apache Spark • Amazon (Trino-based)
Open table format support	• Delta Lake	• Apache Iceberg	• Apache Iceberg • Delta Lake	• Apache Iceberg (Parquet only)

Other competitors

Details	snowflake	
Deployment options	Public cloud only	AWS only
Query engines	• Snowflake	• Amazon Redshift
Open table format support	• Supports Apache Iceberg but primarily uses Cloud Object Storage (COS)	• None, uses Amazon S3 files defined as external tables

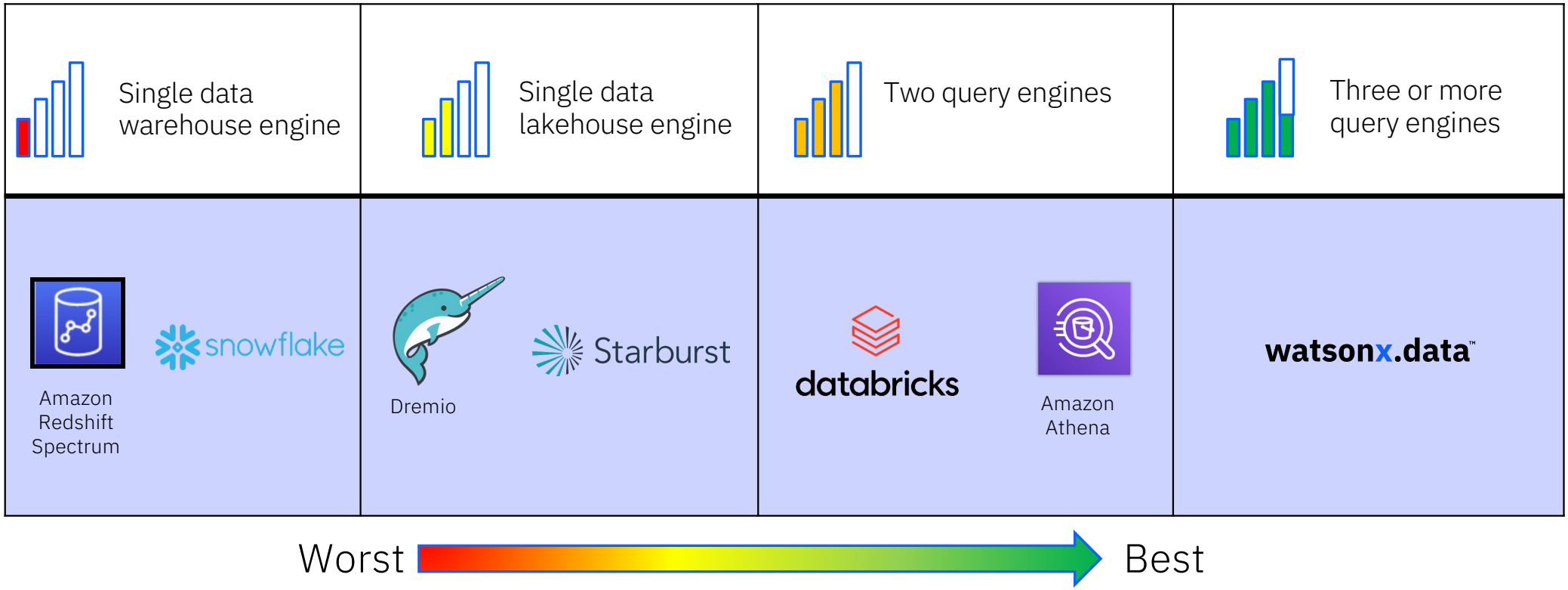
watsonx.data
Primary competitors at a glance
Hybrid cloud



watsonx.data

Primary competitors at a glance

Multiple query engines

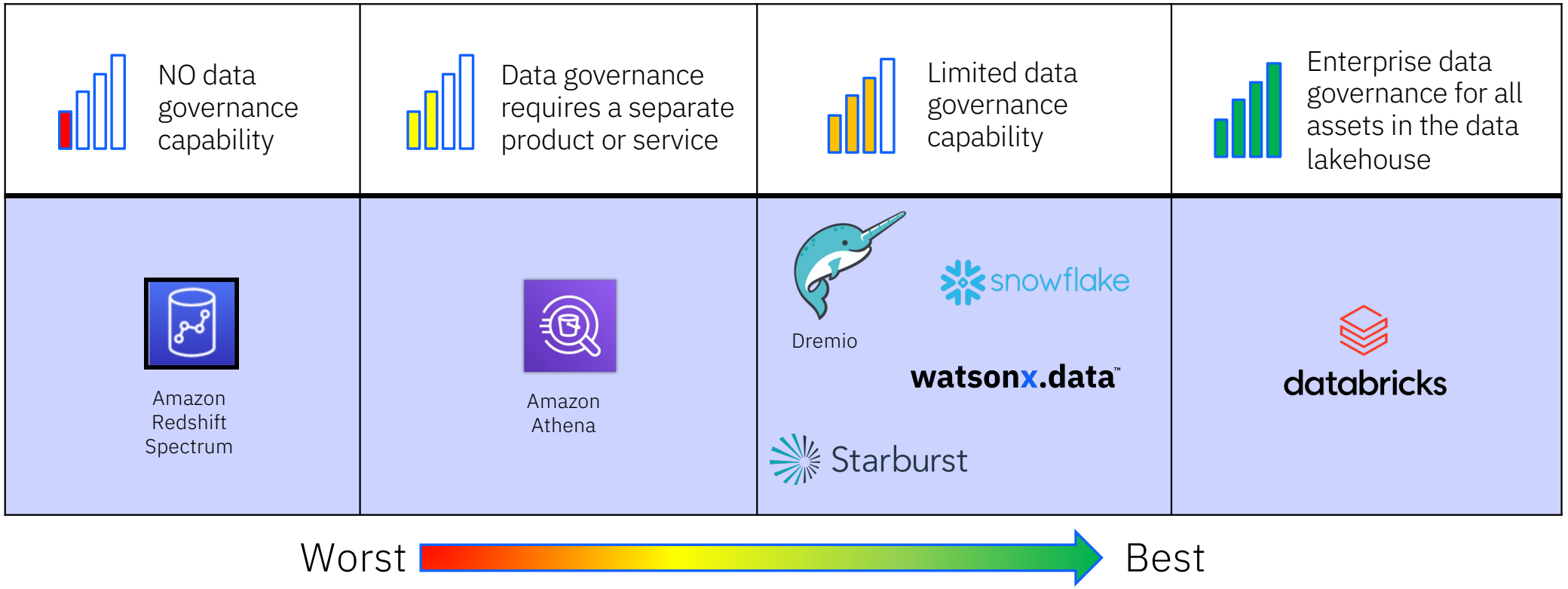


watsonx.data
Primary competitors at a glance
Open-source based

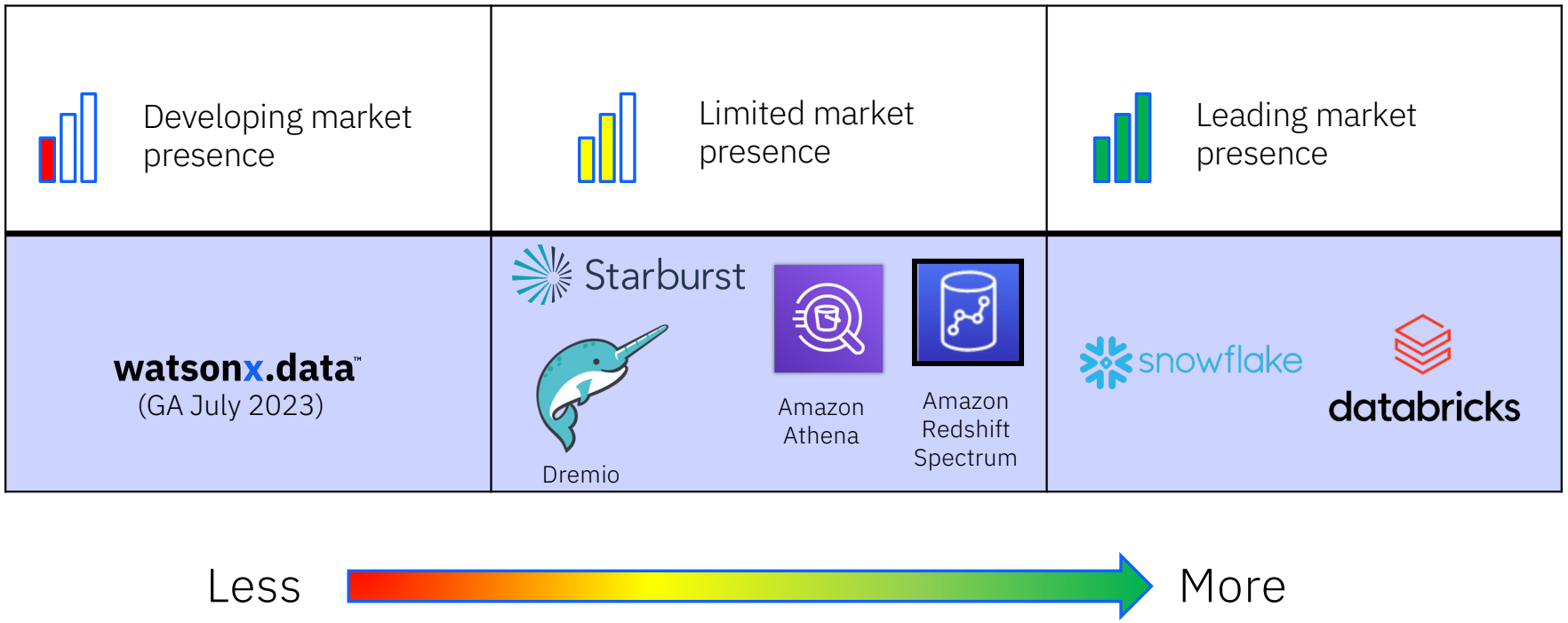
NO open-source components	Single open-source component	Multiple open-source components and single vendor focused community	Multiple open-source components and strong community
Amazon Redshift Spectrum	snowflake	Dremio, Amazon Athena, Starburst, databricks	watsonx.data™

Worst → Best

watsonx.data
Primary competitors at a glance
Data governance



watsonx.data
Primary competitors at a glance
Market presence



watsonx.data
Differentiators

- No other data lakehouse offering has integrated data warehouse engines in addition to the Apache Spark and open-source query engines
- The cloud hyperscalers (AWS, Microsoft Azure, and GCP) along with Databricks provide no hybrid cloud deployment capability
- Deployment flexibility in other clouds – no other data lakehouse offering can be deployed as easily across different cloud platforms
- Other data lakehouse competitors do NOT have the level of experience with mission critical applications, and level of research in query optimization and query processing, as IBM
- Watsonx.data plus other IBM data sources (Netezza Performance Server and Db2) deliver a query performance spectrum not offered by other data lakehouse competitors
- Watsonx.data and its selection of Apache Iceberg and Presto delivers an open solution versus a single contributor open-source lock-in

- Sales calculator overview and how to size, licensing details

watsonx Pricing (for 7/7 GA)

watsonx.data (SaaS)

Consumption-based charges, driven off 3 Price metrics*

- Quantity of nodes deployed
- Duration of nodes deployed
- Support services deployed (e.g., Access Control, Metastore)

*watsonx.data only available for Standard and Premium Tier Customers

Pricing

	Price (USD) / Hour
Cache coordinator node per hour	$2.80
Cache worker node cost per hour	$2.80
Compute coordinator node per hour	$6.50
Compute worker node cost per hour	$6.50
Supporting services per hour	$3.00

watsonx.data (On-Prem)

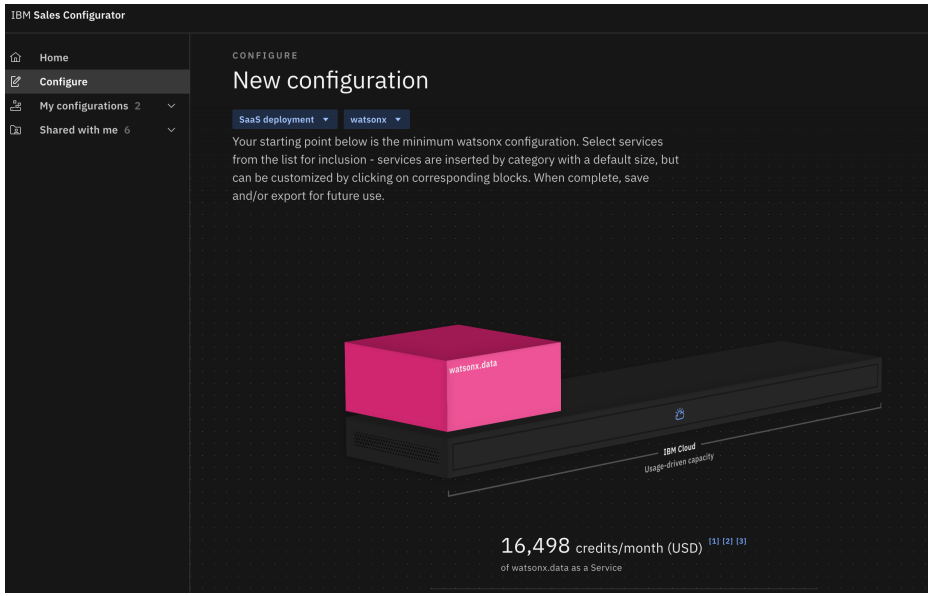
- Price Metric: VPCs (16 VPCs/node) @ $7500/VPC
- Minimum of 10 nodes (160 VPCs) recommended
- Production versions available in Perpetual, Subscription and Monthly Licenses. Options for Non-production, reserved, and various support tiers also available.

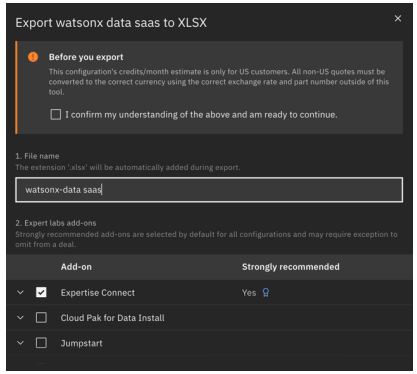
Pricing

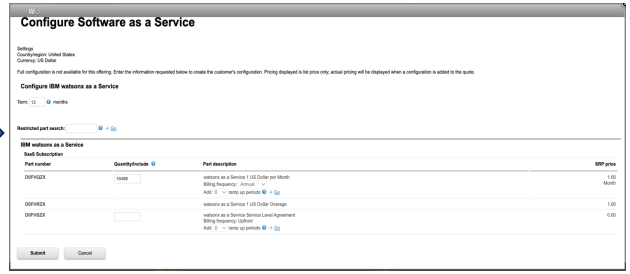
License Type	Price (USD) / VPC
Perpetual	$7,500
Subscription	$250 / month
Monthly	$312.50 /month

On-prem Parts are listed here

Leverage the sales configurator for sizing- SW and SaaS







Note: Recommend engaging Expert Labs Services and Client Engineering to accelerate productive use

watsonx.data SaaS Pricing – What you need to know

Consumption-based charges, driven off 3 Price metrics*
- Quantity of nodes deployed
- Duration of nodes deployed
- Support services deployed (e.g., Access Control, Metastore)

*watsonx.data only available for Standard and Premium Tier Customers

Consumption measured in Resource Units (RU). 1 RU = $1 USD

Pricing	RU / Hour
Cache node cost per hour	$2.80
Compute node cost per hour	$6.50
Supporting services per hour	$3.00

Key Takeaways
- Usage can occur on IBM Cloud or AWS infrastructure
- Pricing depends on type of nodes, number of nodes used, and usage of nodes
  - AWS infrastructure provides two node types (Cache and Compute Optimized), IBM Cloud infrastructure has one node type (Cached Optimized). Prices are the same for AWS and IBM Cloud
  - Number of nodes relates to T-shirt sizes.
  - Usage relates to uptime of services.
- Pricing is measured in Resource Units. 1 RU = $1 USD

Estimated T-Shirt Sizes for IBM Cloud

Sizing	Nodes	Total vCPU	Total RAM (GiB)	Resource Unit per Month*
Starter	2 nodes	32 vCPU	256 GB	•Using service 100% of the time: 6278 •Using service 70% of the time: 5052 •Using service 35% of the time: 3621
Small	4 nodes	64 vCPU	512 GB	•Using service 100% of the time: 10366 •Using service 70% of the time: 7913 •Using service 35% of the time: 5052
Medium	7 nodes	112 vCPU	896 GB	•Using service 100% of the time: 16498 •Using service 70% of the time: 12205 •Using service 35% of the time: 7198
Large	12 nodes	192 vCPU	1536 GB	•Using service 100% of the time: 28762 •Using service 70% of the time: 20790 •Using service 35% of the time: 11490

*Resource Unit Values in the table can be converted to US Dollar value at 1:1 Ratio

Standard (on-Prem) T-shirt Sizing

T-shirt size for a single cluster with performance characteristic: If customer wants multiple identical environments for different departments or Dev+Prod, multiple across
This assumes a worst-case scenario - 100% of your data is in active memory. **Note**: These sizes below are each for 1 cluster

	Description	License (VPC) for 1 cluster of this profile	Base Entitlement (UI,HMS, Etc)	Total Entitlement	Total HW requirements
Small	1 Coordinator node, 3 worker nodes Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Example of estimated workload supported by the configuration: • 1 to 5 queries processing 100GB → 400 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 8GB each • 100-200GB Active data compressed on disk required to be processed.	64	12	76	76 vCPU Total Memory:608Gb
Medium	1 Coordinator node, 9 worker Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Example of estimated workload supported by the configuration: • 1 to 5 queries processing 250GB → 1125 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 22GB each • 250-500GB Active data compressed on disk required to be processed.	160	24	184	184vCPU Memory: 1472Gb
Large	1 Coordinator node 19 worker Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Example of estimated workload supported by the configuration: • 1 to 5 queries processing 500GB → 2250 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 45GB each • 500-1000GB Active data compressed on disk required to be processed.	320	48	368	368vCPU Memory: 2944Gb
X-Large	70 Nodes to configure as desired (i.e 10 clusters of 1 head node and 6 worker nodes or 1 large cluster with 70 nodes). Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Example of estimated workload supported by the configuration: • 1 to 5 queries processing 1750GB → 7875 GB expanded/uncompressed active data in memory or up to 175 queries processing up to 45GB each • 1750-3500GB Active data compressed on disk required to be processed.	1120	48	1168	1168vCPU Memory: 9344Gb

Roadmap

- its Demo Time Retail Demo

- Watsonx.data hands on lab - Watsonx.data for Retail


QUIZ TIME

watsonx.data™

watsonx.data™

Agenda

- Watsonx.data hands on lab - Watsonx.data for Retail
- Use Cases
- Competitive Insights
- Roadmap
- Sales calculator overview and how to size, licensing details
- Training recap followed by quiz

Sample Discovery Questions

- What type of database are they using today
- Is cost an issue or starting to become an issue
- What are the use cases and how do they serve the business
- What are the required SLAs? Who are the users accessing? How fast do they need answers?
- Are there ETL workloads within the EDW, what are the time windows
- Is there historical data in the EDW that is not frequently accessed but still required to be there?
- Overall data size and instance size

IBM Cross Sell

What do I sell if my customer has:

- IBM analytics appliance?
- IBM Db2 Warehouse or Netezza?
- Db2 for z/OS?

IBM Modernize + Cross Sell

Customer Goal: Customer has an IBM on-premises solution and want to move to next generation of self-managed and SaaS offerings

Customer motivation:

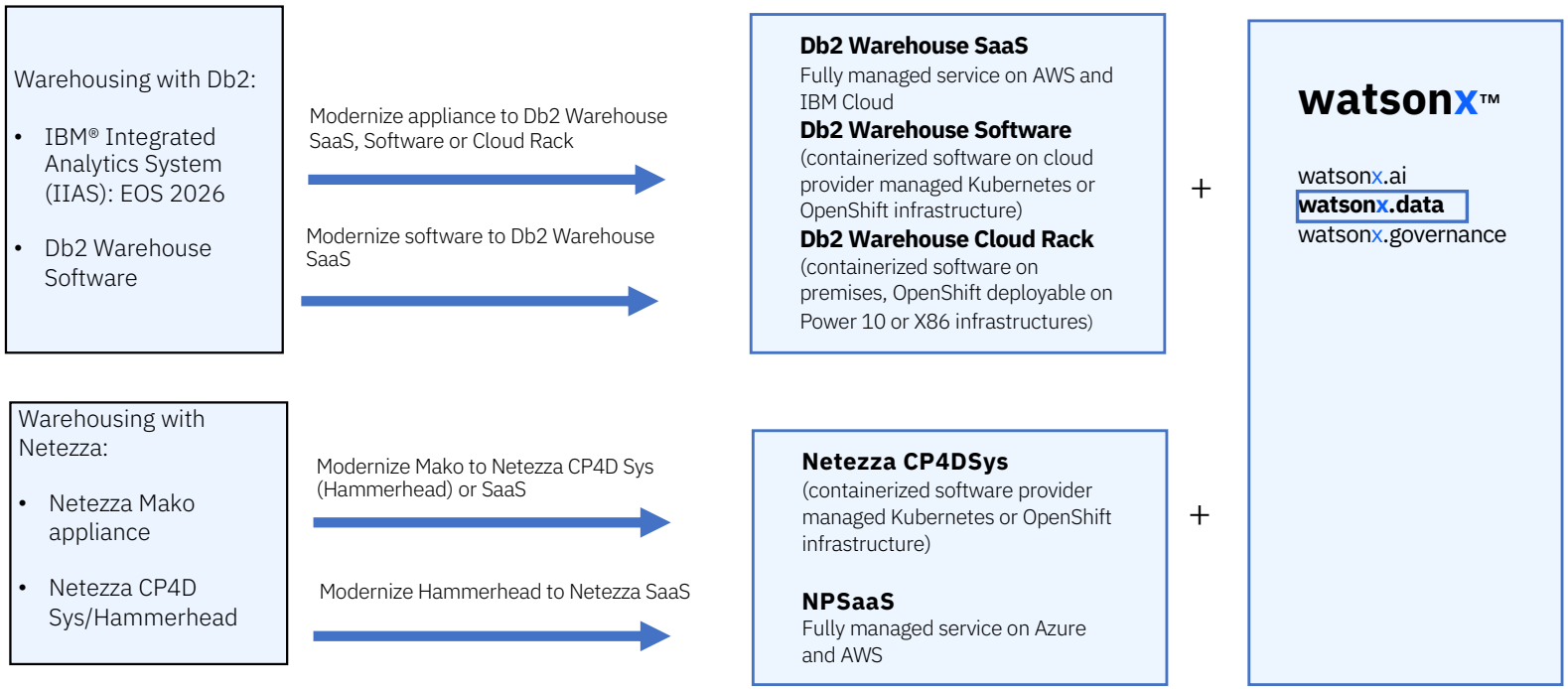
- Looking for next generation features
- Limited compute or data storage resulting in offloading or expanding data infrastructure
- Difficulty scaling up and down for workloads
- New use cases: AI/ML, Real-time analytics, data engineering, data sharing

Challenges to Address:

- System migration compatibility
- Sizing management
- Cost controls on the cloud
- Access to all data across hybrid cloud without duplication/movement

Key Messages:

1. Modernize your database appliance with like for like compatibility and support new use cases
2. Optimize workloads for AI with fit for purpose engines and reduce warehouse costs by 50%, access all governed data across hybrid cloud



IBM Db2 for z/OS Cross-sell

Customer Goal: Customer has IBM Db2 for z/OS and wants to access transactional data from mainframe for AI use cases

Customer motivation:

- Brand new use cases with mainframe transactional data: AI/ML, Real-time analytics, data engineering, data sharing

Challenges to Address:

- System migration compatibility

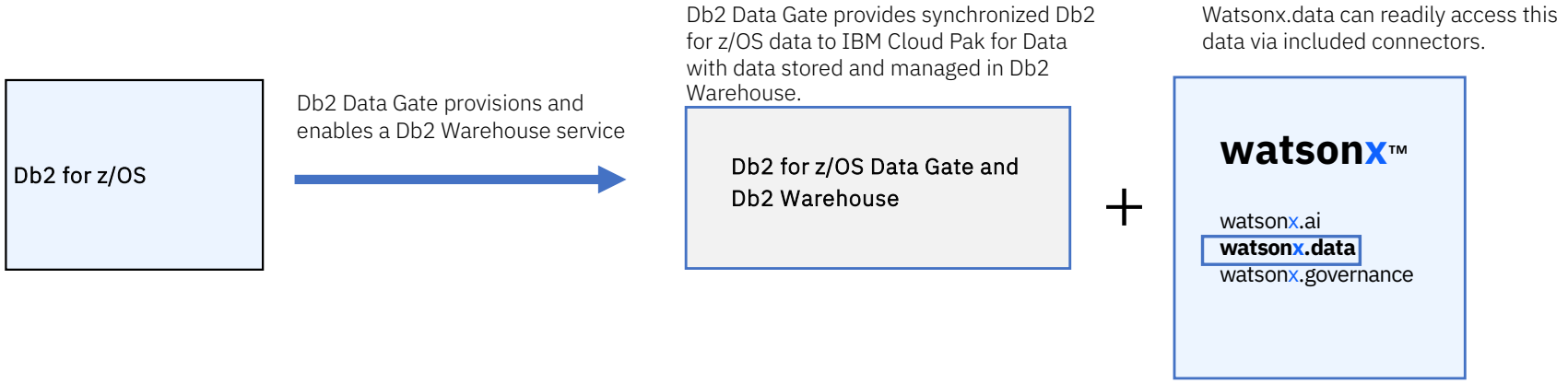
Key Messages z/OS and Db2 Warehouse:

- An integrated, optimized synchronization feature maintains currency between source Db2 for z/OS data on IBM Z and Db2 Warehouse targets
- Db2 Warehouse is a relational databases that delivers advanced data management and analytics capabilities

Key Messages z/OS and .data:

- Access and analyze data from mainframe systems in near-real time
- Consume less processor capacity with built-in synchronization
- Use the most up-to-date data from mainframe for machine learning models

---



Concepts and market entry points

Three key concepts for IBM watsonx.data

1. Presto is a next-generation open-source SQL engine designed to run efficiently over data lakes.

2. Warehouses and first-generation lakehouses are monolithic, and not optimized to work on all workloads. Only IBM watsonx.data's multi-engine architecture allows for true workload optimization.

3. Iceberg is an open-table format that allows multiple engines to access the same data – this means, Snowflake, Netezza, and IBM watsonx.data can all access data in Iceberg at the same time.

Market entry points + use cases

1. Warehouse Optimization narrative

- Competitive and IBM (Netezza + Db2)
- Talk track – Client is concerned with the spend on traditional warehouse today – looking to optimize for both performance and cost
- **Value prop:** Cost optimization and openness through the shared meta layer and fit-for-purpose engines
- Use case – Snowflake write-intensive workloads moving to Spark and/or Presto, thus reducing cost of Snowflake virtual data warehouses

2. Modernizing data lake narrative

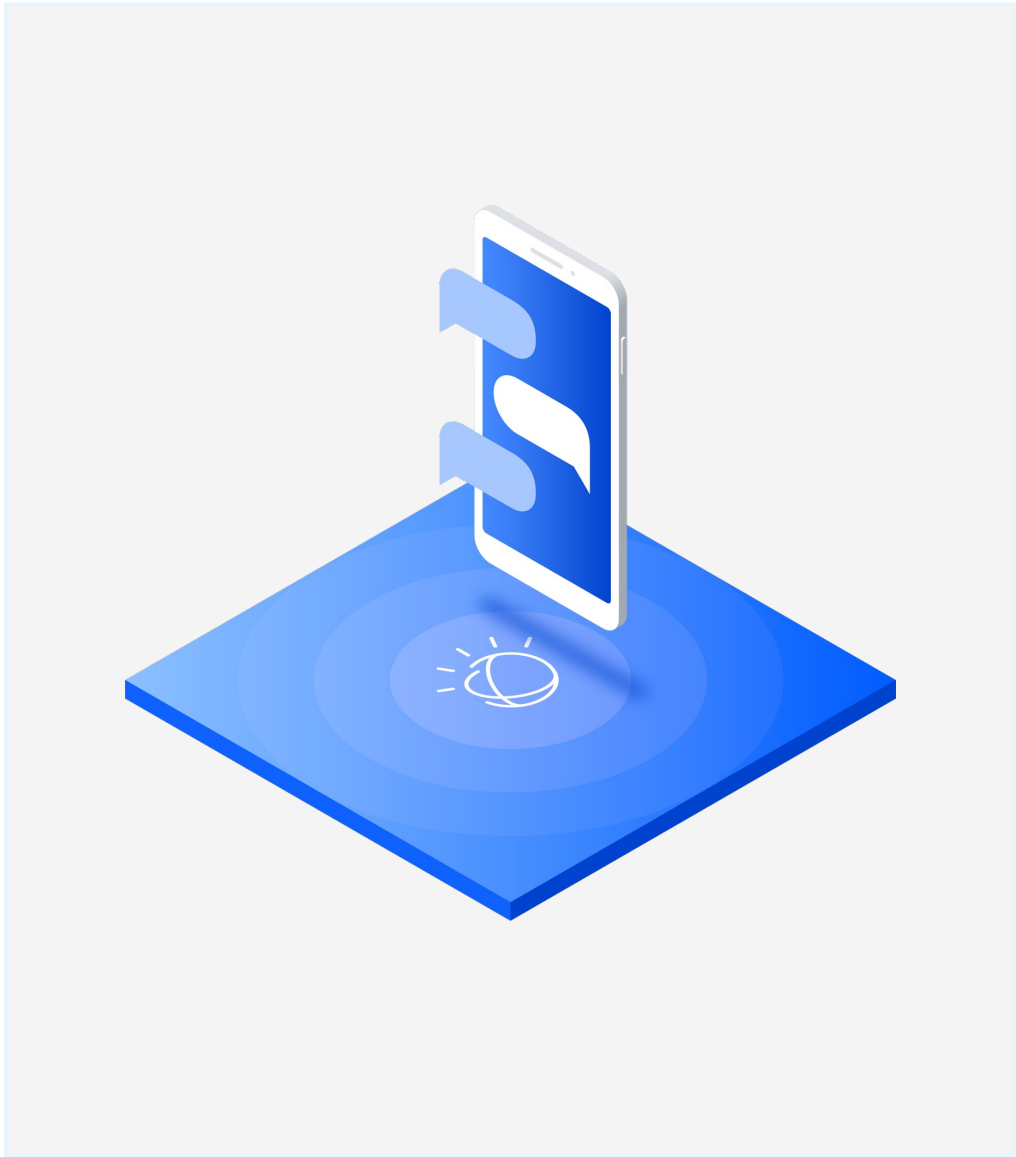
- Modernizing storage architecture to facilitate shared metadata and fit-for-purpose engines
- Talk track – Converting legacy file storage structures into open-file structures and assigning those into the shared meta layer, thus facilitating fit-for-purpose engines
- Value prop: Modernize with warehouse-like performance for querying data in open formats, built-in governance
- Use case – Move from HDFS to open-source iceberg within a consolidated shared meta layer

Competitors and objection handling

IBM **watsonx.data** competitive insights

- Learn more about the competition
- Provide feedback
- Request help with an opportunity

Get the presentation on Seismic →



Types of competitors

A data lakehouse combines the best features of data warehouses and data lakes to provide cost optimization for clients. Compute and storage are separated so that data can be accessed from different engines.

An augmented data warehouse may be able to access different types of data, but all compute processing is performed through the data warehouse engine. There is no compute cost optimization allowing different engines, only the compute amount used by the data warehouse engine can be adjusted.

- Data lakehouse competitors

- Designed for both structured and unstructured data
- Based on open table and data formats for data storage
- Fit for purpose query engine for different use cases
- Cost optimization for compute engine and storage
- Separate compute and storage

- Augmented data warehouse competitors

- Primarily designed for structured and semi- structured data
- Uses proprietary or open file formats, but supports open table format for data storage
- Query processing uses data warehouse engine
- Compute and storage are not completely separated

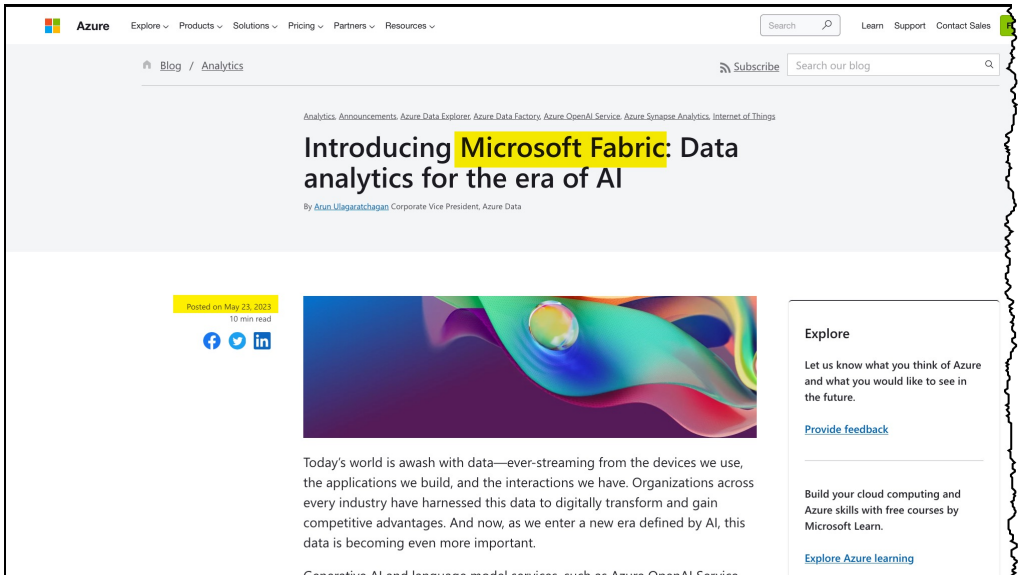
watsonx.data

Competitor overview (weaknesses not mentioned)

All competitors are relatively new companies (within the past decade) and are rapidly growing in the public cloud market

	Competitor	
Data Lakehouse Competitors	Databricks	Databricks pioneered the term “Lakehouse”. It’s currently positioned as a leader in the emerging Lakehouse market. Founded in 2013 by the creators of Apache Spark, Databricks offers a unified analytics platform for a variety of use-cases such as data engineering, machine learning, data science, and AI.
	Dremio	Dremio has gained significant recognition for its proprietary Dremio query engine (Sonar) technology. It is positioned as a leader in the data lakehouse market, with a growing market share. Founded in 2015, Dremio offers an open, cloud-native data lakehouse engine that simplifies and accelerates data processing and analytics.
	Starburst	Starburst has made a name for itself in the data access and analytics space. It is positioned as a leader in the enterprise data access market, with a growing market share. Founded in 2017, Starburst offers a cloud-native platform that enables fast and easy access to data across a range of sources.
	Amazon Athena	Amazon Athena was first released in 2016 and is the AWS data lakehouse offering that utilizes Apache Spark for analytics on data in open file formats and the Trino engine for SQL queries. Amazon Athena combines with other AWS services, like AWS Lake Formation, for data governance to build a complete lakehouse solution.
Others	Snowflake	Founded in 2012, Snowflake has made significant strides in the cloud data warehousing market and is currently positioned as a leader in the cloud data platform space. Snowflake supports open table formats but locks clients into the Snowflake environment that is locked and controlled (not open-source based). It has a single SQL query engine and a limited ability to access data outside of a Snowflake data warehouse, and no hybrid cloud capability.
	Amazon Redshift Spectrum	Amazon Redshift Spectrum is a Redshift service that allows direct queries on data stored in Amazon S3 files without having to load the data into an Amazon Redshift data warehouse. Amazon Redshift Spectrum requires an active Amazon Redshift data warehouse cluster to execute queries, so it is tightly integrated with Amazon Redshift and extends the data warehouse to access external tables in Amazon S3.

watsonx.data
Breaking news



May 23, 2023 – Microsoft announces Microsoft Fabric

- Points to remember
- Microsoft Fabric is only in PREVIEW
- Microsoft Fabric is only available on Microsoft Azure
- Microsoft Fabric is more than a data lakehouse, it is a data analytics platform

- IBM discussion points
- **watsonx.data** competes directly with Azure Databricks
- Microsoft Fabric competes with the overall **watsonx** platform
- PREVIEW items are NOT available for production use

Source: Introducing Microsoft Fabric: Data analytics for the era of AI blog

Response to Databricks Announcement June 2023

Databricks recently made several announcements at their Data + AI Summit (June 26-29, 2023), this document provides a summary of these new features and capabilities along with competitive analysis from an IBM watsonx.data perspective.

The key news was in the areas of Delta Lake and Unity Catalog. Each announcement will contain a technical overview of the feature or capability, its status (including a tentative release data if available), and a competitive analysis compared to watsonx.data.

Review the response on Seismic →

Competitive landscape

Data lakehouse competitors

Details	databricks		Starburst	
Deployment options	Public cloud only	Public cloud & on-premises	Public cloud & on-premises	AWS only
Query engines	• Apache Spark • Photon	• Dremio Sonar (proprietary)	• Starburst (Trino-based)	• Apache Spark • Amazon (Trino-based)
Open table format support	• Delta Lake	• Apache Iceberg	• Apache Iceberg • Delta Lake	• Apache Iceberg (Parquet only)

Other competitors

Details	snowflake	
Deployment options	Public cloud only	AWS only
Query engines	• Snowflake	• Amazon Redshift
Open table format support	• Supports Apache Iceberg but primarily uses Cloud Object Storage (COS)	• None, uses Amazon S3 files defined as external tables

watsonx.data

Primary competitors at a glance

Hybrid cloud

watsonx.data

Primary competitors at a glance

Multiple query engines

Single data warehouse engine	Single data lakehouse engine	Two query engines	Three or more query engines
Amazon Redshift Spectrum snowflake	Dremio Starburst	databricks Amazon Athena	watsonx.data™

watsonx.data

Primary competitors at a glance

Open-source based

NO open-source components	Single open-source component	Multiple open-source components and single vendor focused community	Multiple open-source components and strong community
Amazon Redshift Spectrum	snowflake	Dremio, Amazon Athena, Starburst, databricks	watsonx.data™

Worst → Best

watsonx.data

Primary competitors at a glance

Data governance

NO data governance capability	Data governance requires a separate product or service	Limited data governance capability	Enterprise data governance for all assets in the data lakehouse
Amazon Redshift Spectrum	Amazon Athena	Dremio, snowflake, watsonx.data™, Starburst	databricks

Worst → Best

watsonx.data

Primary competitors at a glance

Market presence

watsonx.data
Differentiators

- No other data lakehouse offering has integrated data warehouse engines in addition to the Apache Spark and open-source query engines
- The cloud hyperscalers (AWS, Microsoft Azure, and GCP) along with Databricks provide no hybrid cloud deployment capability
- Deployment flexibility in other clouds – no other data lakehouse offering can be deployed as easily across different cloud platforms
- Other data lakehouse competitors do NOT have the level of experience with mission critical applications, and level of research in query optimization and query processing, as IBM
- Watsonx.data plus other IBM data sources (Netezza Performance Server and Db2) deliver a query performance spectrum not offered by other data lakehouse competitors
- Watsonx.data and its selection of Apache Iceberg and Presto delivers an open solution versus a single contributor open-source lock-in

Watsonx.data T-shirt sizing

watsonx Pricing (for 7/7 GA)

watsonx.data (SaaS)

Consumption-based charges, driven off 3 Price metrics*

- Quantity of nodes deployed
- Duration of nodes deployed
- Support services deployed (e.g., Access Control, Metastore)

*watsonx.data only available for Standard and Premium Tier Customers

Pricing

	Price (USD) / Hour
Cache coordinator node per hour	$2.80
Cache worker node cost per hour	$2.80
Compute coordinator node per hour	$6.50
Compute worker node cost per hour	$6.50
Supporting services per hour	$3.00

watsonx.data (On-Prem)

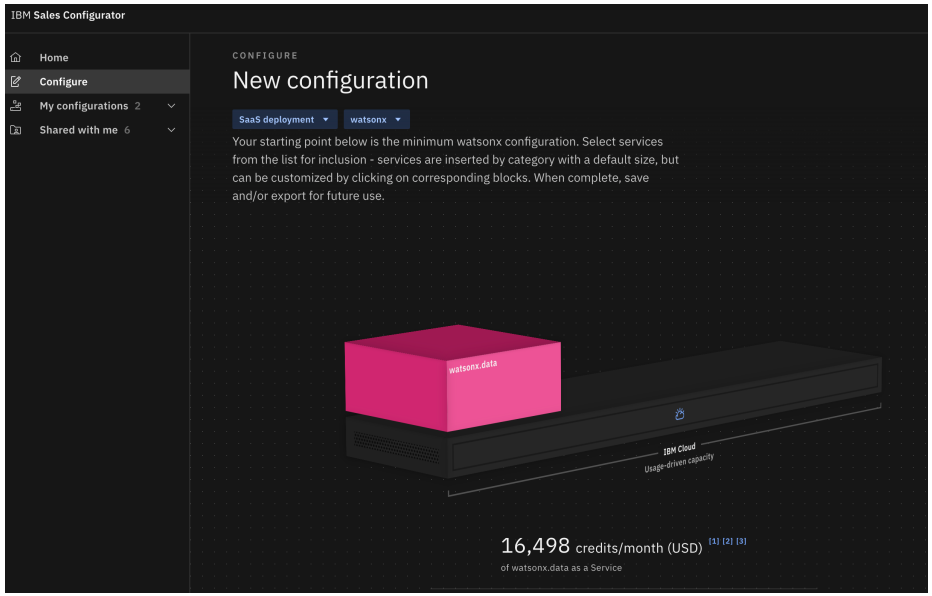
- Price Metric: VPCs (16 VPCs/node) @ $7500/VPC
- Minimum of 10 nodes (160 VPCs) recommended
- Production versions available in Perpetual, Subscription and Monthly Licenses. Options for Non-production, reserved, and various support tiers also available.

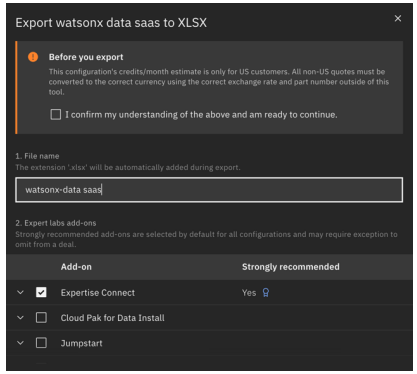
Pricing

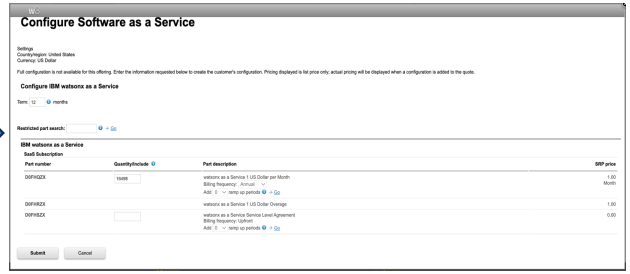
License Type	Price (USD) / VPC
Perpetual	$7,500
Subscription	$250 / month
Monthly	$312.50 /month

On-prem Parts are listed here

Leverage the sales configurator for sizing- SW and SaaS







Note: Recommend engaging Expert Labs Services and Client Engineering to accelerate productive use

Watsonx.data – PIDS

IBM Confidential

Start with minimum configuration of 10 nodes…
*… and add P*Q on top as needed*

Recommend multiples of 10 nodes (10, 20, etc)

Service	Part Number	List Price	Price Metric	Quantity*	Total Price	After 70% Discount
watsonx.data (License + S&S)	D0F4SZX	$7,500 /core	VPCs (16 VPCs/node)	Ex) 160 VPCs	$1,200,000	$360,000
watsonx.data (S&S)	E0F4RZX	$1,500 /core	VPCs (16 VPCs/node)	Ex) 160 VPCs	$240,000	$72,000
watsonx.data (Subscription)	D0F3HZX	$3,000 /core	VPCs (16 VPCs/node)	Ex) 160 VPCs	$480,000	$144,000
Expert Labs Services Bundle	D06W5ZX (Plan) D06W3ZX (Install) D06W6ZX (Build)	$21,000 $18,000 $37,000	Weeks	Qty 2	$21,000 $18,000 $37,000	n/a

watsonx.data SaaS Pricing – What you need to know

Consumption-based charges, driven off 3 Price metrics*
- Quantity of nodes deployed
- Duration of nodes deployed
- Support services deployed (e.g., Access Control, Metastore)

*watsonx.data only available for Standard and Premium Tier Customers

Consumption measured in Resource Units (RU). 1 RU = $1 USD

Pricing	RU / Hour
Cache node cost per hour	$2.80
Compute node cost per hour	$6.50
Supporting services per hour	$3.00

Key Takeaways
- Usage can occur on IBM Cloud or AWS infrastructure
- Pricing depends on type of nodes, number of nodes used, and usage of nodes
  - AWS infrastructure provides two node types (Cache and Compute Optimized), IBM Cloud infrastructure has one node type (Cached Optimized). Prices are the same for AWS and IBM Cloud
  - Number of nodes relates to T-shirt sizes.
  - Usage relates to uptime of services.

Estimated T-Shirt Sizes for IBM Cloud

Sizing	Nodes	Total vCPU	Total RAM (GiB)	Resource Unit per Month*
Starter	2 nodes	32 vCPU	256 GB	•Using service 35% of the time: 3621 •Using service 70% of the time: 5052 •Using service 100% of the time: 6278
Small	4 nodes	64 vCPU	512 GB	•Using service 35% of the time: 5052 •Using service 70% of the time: 7913 •Using service 100% of the time: 10366
Medium	7 nodes	112 vCPU	896 GB	•Using service 35% of the time: 7198 •Using service 70% of the time: 12205 •Using service 100% of the time: 16498
Large	12 nodes	192 vCPU	1536 GB	•Using service 35% of the time: 11490 •Using service 70% of the time: 20790 •Using service 100% of the time: 28762

*1 RU = 1 watsonx or CPDaaS credit = $1 USD at list price

Up to 25% discount on credits

Standard (on-Prem) T-shirt Sizing (High-Level)

S, M, and L T-shirt sizes are for a single cluster with performance characteristics. Xl and XXL are bundles of entitlement for multiple clusters. The sizing chart assumes a worst-case scenario - 100% of your data is in active memory.

T-Shirt Sizing	Description	Total VPC Entitlement	Effective Price After Standard Discounts (75%)	Total HW requirements
Small	**Lowest cost option for customers to handle a small use case** Supports 1 to 5 queries processing 100GB → 400 GB expanded/uncompressed active data in memory or up to 50 concurrent queries processing up to 8GB each	76	Perpetual: $142,500 1Yr Subscription: $57,000	76 vCPU Total Memory:608Gb
Medium	**Recommended sizing to start customers on** Supports 1 to 5 queries processing 250GB → 1125 GB expanded/uncompressed active data in memory or up to 50 concurrent queries processing up to 22GB each	184	Perpetual: $345,000 1Yr Subscription: $138,000	184vCPU Memory: 1472Gb
Large	**Largest t-shirt sizing for a single presto cluster** Supports 1 to 5 queries processing 500GB → 2250 GB expanded/uncompressed active data in memory or up to 50 concurrent queries processing up to 45GB each	368	Perpetual: $690,000 1Yr Subscription: $276,000	368vCPU Memory: 2944Gb
X-Large	**A bundle of entitlements allowing customers to deploy multiple clusters. Good for large customers looking to stand up multiple environments** Enough entitlement to stand up multiple clusters that can support 1 to 5 queries processing 1750GB → 7875 GB expanded/uncompressed active data in memory or up to 175 queries processing up to 45GB each	1168	Perpetual: $2,190,000 1Yr Subscription: $876,000	1168vCPU Memory: 9344Gb
XX-Large	**For our largest customers who need enough entitlements to deploy multiple presto clusters for multiple teams** Enough entitlement to stand up multiple clusters that can support 1 to 5 queries processing 5000GB → 22,500 GB expanded/uncompressed active data in memory or up to 500 queries processing up to 45GB each	3248	Perpetual: $6,090,000 1Yr Subscription: $2,436,000	3248vCPU Memory: 25600Gb

Standard (on-Prem) T-shirt Sizing (Detailed)

S, M, and L T-shirt sizes are for a single cluster with performance characteristics. Xl and XXL are bundles of entitlement for multiple clusters. The sizing chart assumes a worst-case scenario - 100% of your data is in active memory.

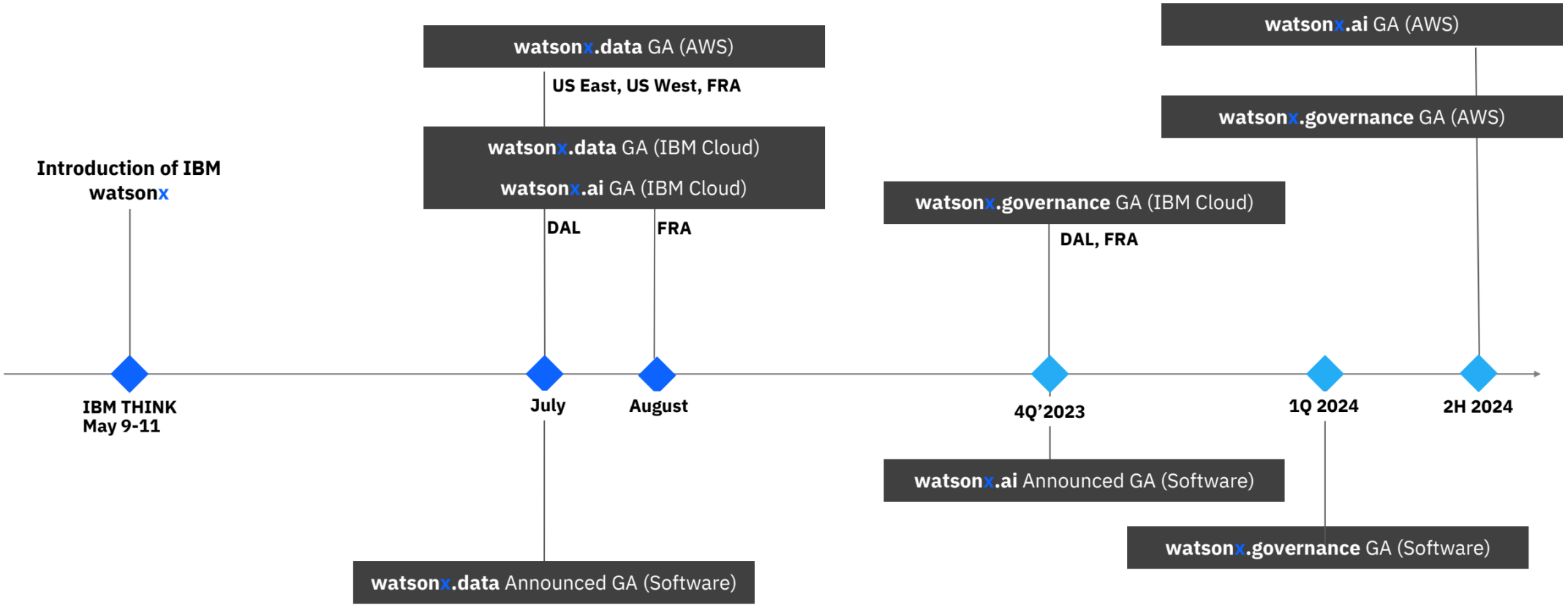
T-Shirt Sizing	Description	Total VPC Entitlement	Price	Total HW requirements
Small	1 Coordinator node, 3 worker nodes Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Base entitlements for (UI,HMS, etc) = 12 vCPUs Example of estimated workload supported by the configuration: • 1 to 5 queries processing 100GB → 400 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 8GB each • 100-200GB Active data compressed on disk required to be processed.	76	Perpetual List: $570,000 75% Discount: $142,500 1Yr Subscription List: $228,000 75% Discount: $57,000	76 vCPU Total Memory:608Gb
Medium	1 Coordinator node, 9 worker Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Base entitlements for (UI,HMS, etc) = 24 vCPUs Example of estimated workload supported by the configuration: • 1 to 5 queries processing 250GB → 1125 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 22GB each • 250-500GB Active data compressed on disk required to be processed.	184	Perpetual List: $1,380,000 75% Discount: $345,000 1Yr Subscription List: $552,000 75% Discount: $138,000	184vCPU Memory: 1472Gb
Large	1 Coordinator node 19 worker Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Base entitlements for (UI,HMS, etc) = 48 vCPUs Example of estimated workload supported by the configuration: • 1 to 5 queries processing 500GB → 2250 GB expanded/uncompressed active data in memory or up to 50 queries processing up to 45GB each • 500-1000GB Active data compressed on disk required to be processed.	368	Perpetual List: $2,760,000 75% Discount: $690,000 1Yr Subscription List: $1,104,000 75% Discount: $276,000	368vCPU Memory: 2944Gb
X-Large	70 Nodes to configure as desired (i.e 10 clusters of 1 head node and 6 worker nodes or 1 large cluster with 70 nodes). Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Base entitlements for (UI,HMS, etc) = 48 vCPUs Example of estimated workload supported by the configuration: • 1 to 5 queries processing 1750GB → 7875 GB expanded/uncompressed active data in memory or up to 175 queries processing up to 45GB each • 1750-3500GB Active data compressed on disk required to be processed.	1168	Perpetual List: $8,760,000 75% Discount: $2,190,000 1Yr Subscription List: $3,504,000 75% Discount: $876,000	1168vCPU Memory: 9344Gb
XX-Large	200 Nodes to configure as desired (i.e 10 clusters of 1 head node and 19 worker nodes or 1 large cluster with 200 nodes). Each node assumes 16vCPU, 128Gb of memory, and 2 x 1900GB NVMe, up to 10 GbE. Base entitlements for (UI,HMS, etc) = 48 vCPUs Example of estimated workload supported by the configuration: • 1 to 5 queries processing 5000GB → 22,500 GB expanded/uncompressed active data in memory or up to 500 queries processing up to 45GB each • 5000-10,000GB Active data compressed on disk required to be processed.	3248	Perpetual List: $24,360,000 75% Discount: $6,090,000 1Yr Subscription List: $9,744,000 75% Discount: $2,436,000	3248vCPU Memory: 25600Gb

watsonx.data

Roadmap

watsonx

Target timeline



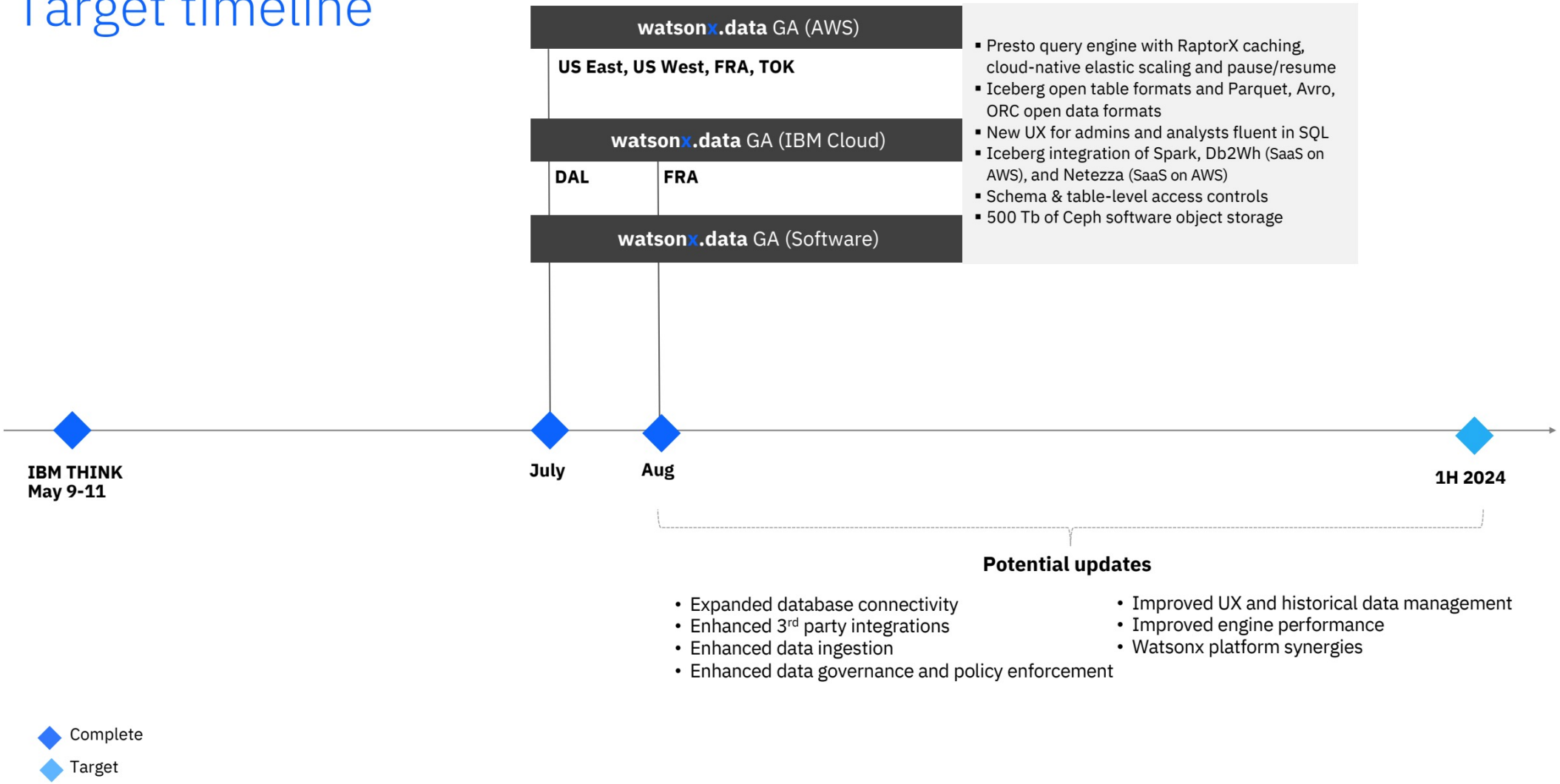
Complete

Target

DAL – Dallas, FRA – Frankfurt

watsonx.data

Target timeline




SUMMARY


QUIZ TIME