databricks

Houston User Workshop

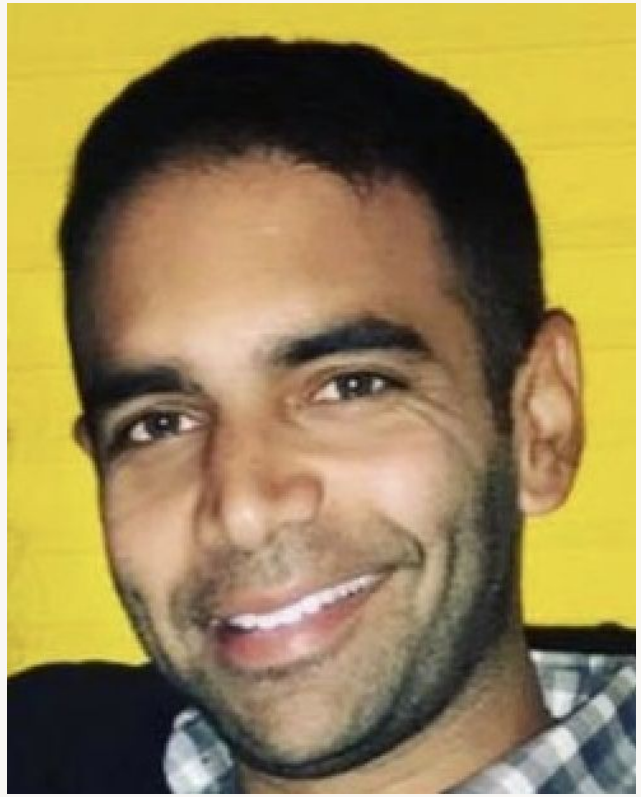
The unification of data, analytics, and AI

Jai Karve, Solutions Architect

Scott Spielman, Solutions Architect

About Me

Jai Karve, Solutions Architect, Databricks



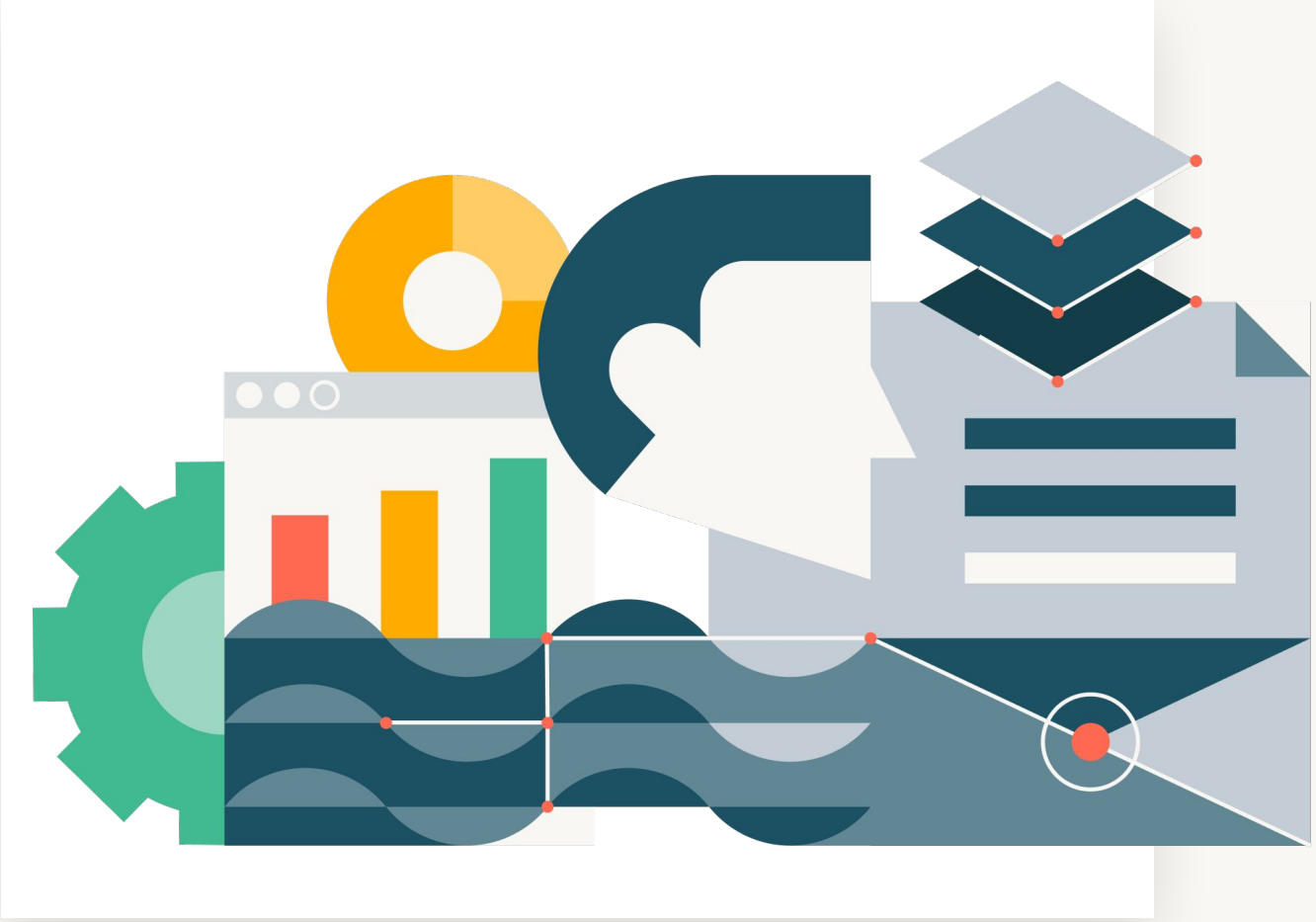
- Favorite food: Oysters
- Shellfish enthusiast
- Been to 40 states & 5 continents
- New Dad (1 year old son)
- Used to have hair

About Me

Scott Spielman, Solutions Architect, Databricks

- Favorite food: Lobster and Crab
- Places I've lived: Boston, Ithaca, NY, Columbus, OH, San Francisco, Austin, Boulder, CO
- Skiing, Car, Guitar Enthusiast

Presentation Overview



What we'll cover

- Environment Setup
- Lakehouse Introduction
- Why Delta
- Delta Live Tables
- Unity Catalog
- Databricks SQL
- Databricks ML (mlflow)

Hands on Overview

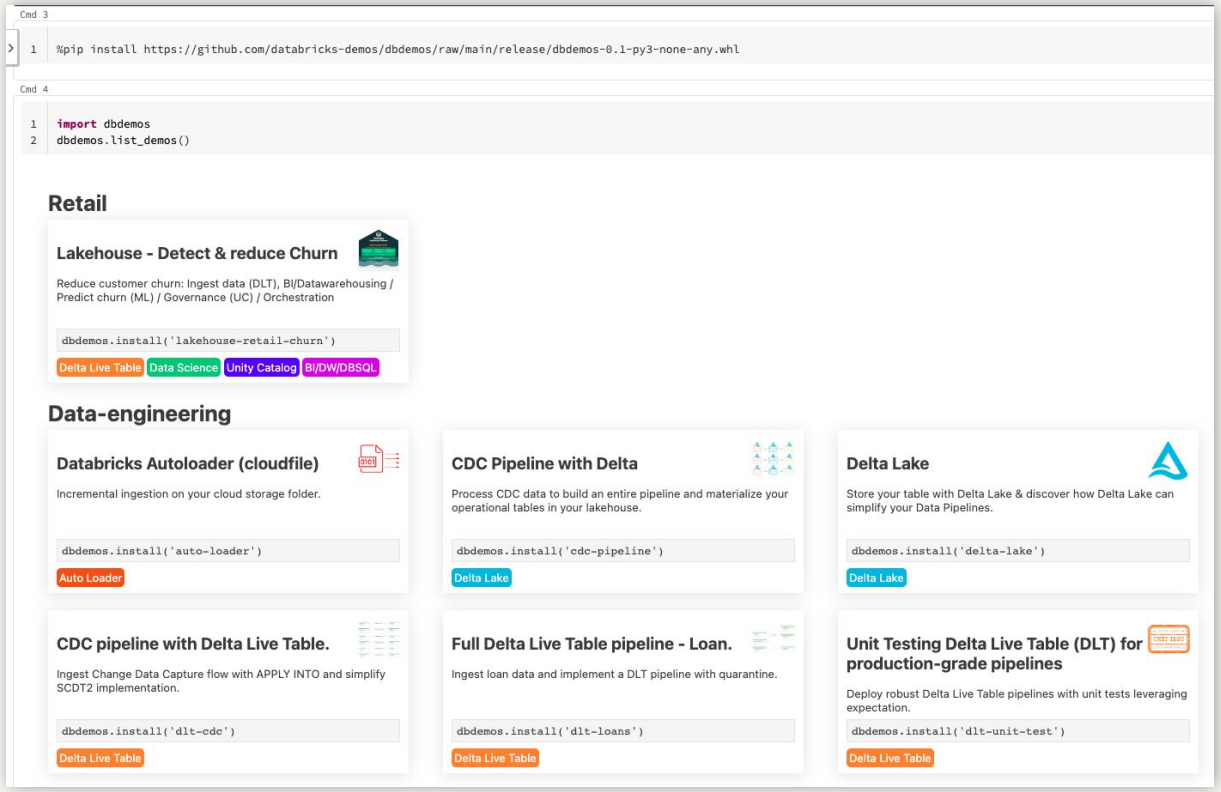
What we'll do

- ETL using DLT
- Data Access using Unity Catalog
- Dashboards with Databricks SQL
- Eat Lunch
- Have Fun
- Ask Questions

Let’s get you set up

Dbdemos: Ready-to use content

Available for customers, industry & product focus



With 1 command line, install your demo Ready-to use, with end 2 end story.

Nice visual, dashboards, pipelines

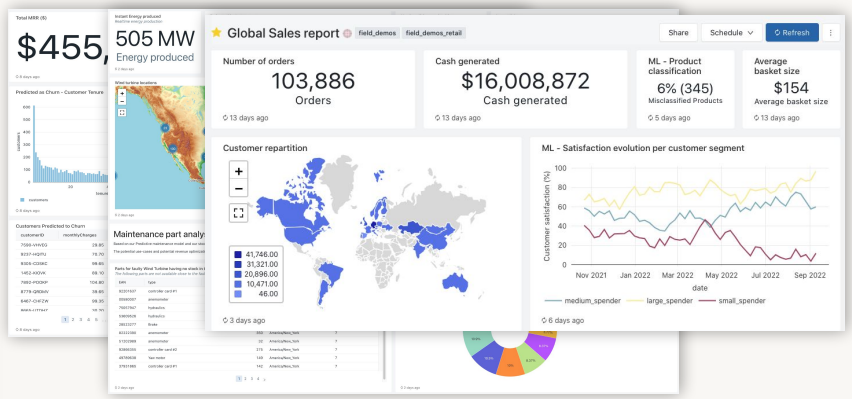
Install in any workspace, for any industry

- Data engineering
- Governance / Security (UC)
- Data Science
- DW / BI

IoT & Predictive Maintenance Demo

Detect faulty wind turbine and orchestrate maintenance accordingly

- Ingest data from external system in streaming (sensors/ERP/...) and then transform it using Delta Live Tables (DLT)
- Secure our ingested data to ensure governance and security
- Leverage Databricks DBSQL and the warehouse endpoints to build dashboard to analyze the ingested data
- **The installed assets will serve as solutions. We will be building some of these assets out (i.e. DLT) as part of the workshop portion of the class.**



Loads demos in your workspace (commands must be in separate cells & executed sequentially)

```
%pip install dbdemos
```

```
import dbdemos
dbdemos.install('lakehouse-iot-platform')
```

Installs data, dashboards, ML models, DLT, Workflows

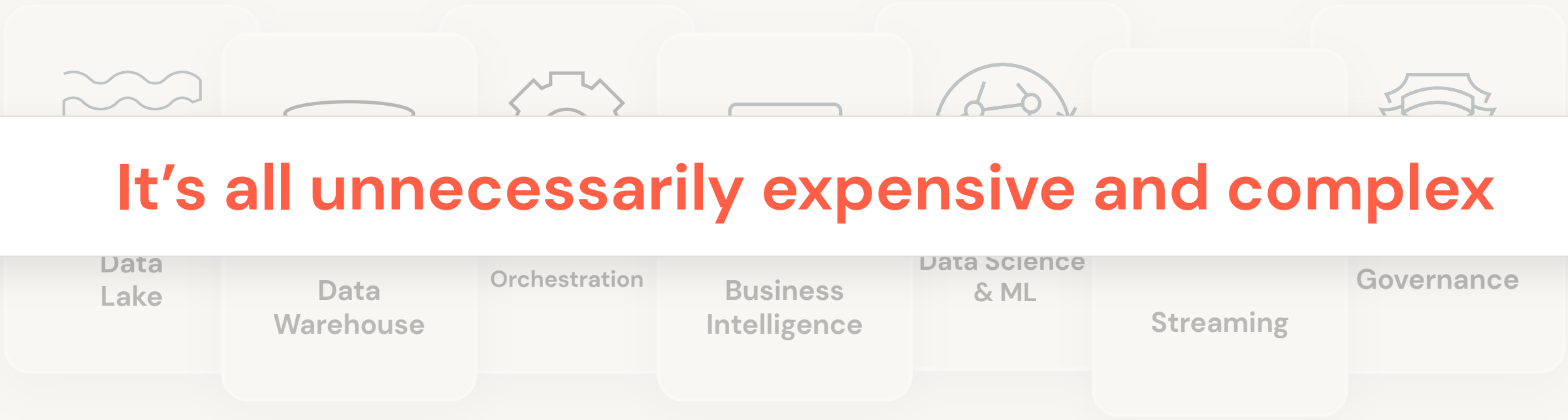
GitHub Assets

Class resources and this presentation

- Github repo located at https://github.com/jaikarve/dbx-htx-workshop contains:
  - Import the Workbook DBC archive into workspace; provides exercise instructions, hints, and stubbed out cells to fill out answers
  - Access this presentation as a PDF
- **Action Item:** Import the Workbook DBC archive into your workspace.

Lakehouse Introduction

Today, you stitch together too many platforms



It's all unnecessarily expensive and complex

Data silos drive high operational costs

Inconsistent policies reduce trust in the data

Disparate tools inhibit cross-team productivity

A data lakehouse takes a different approach

One platform to support multiple personas

BI & Data Warehousing

Data Engineering

Data Streaming

Data Science & ML

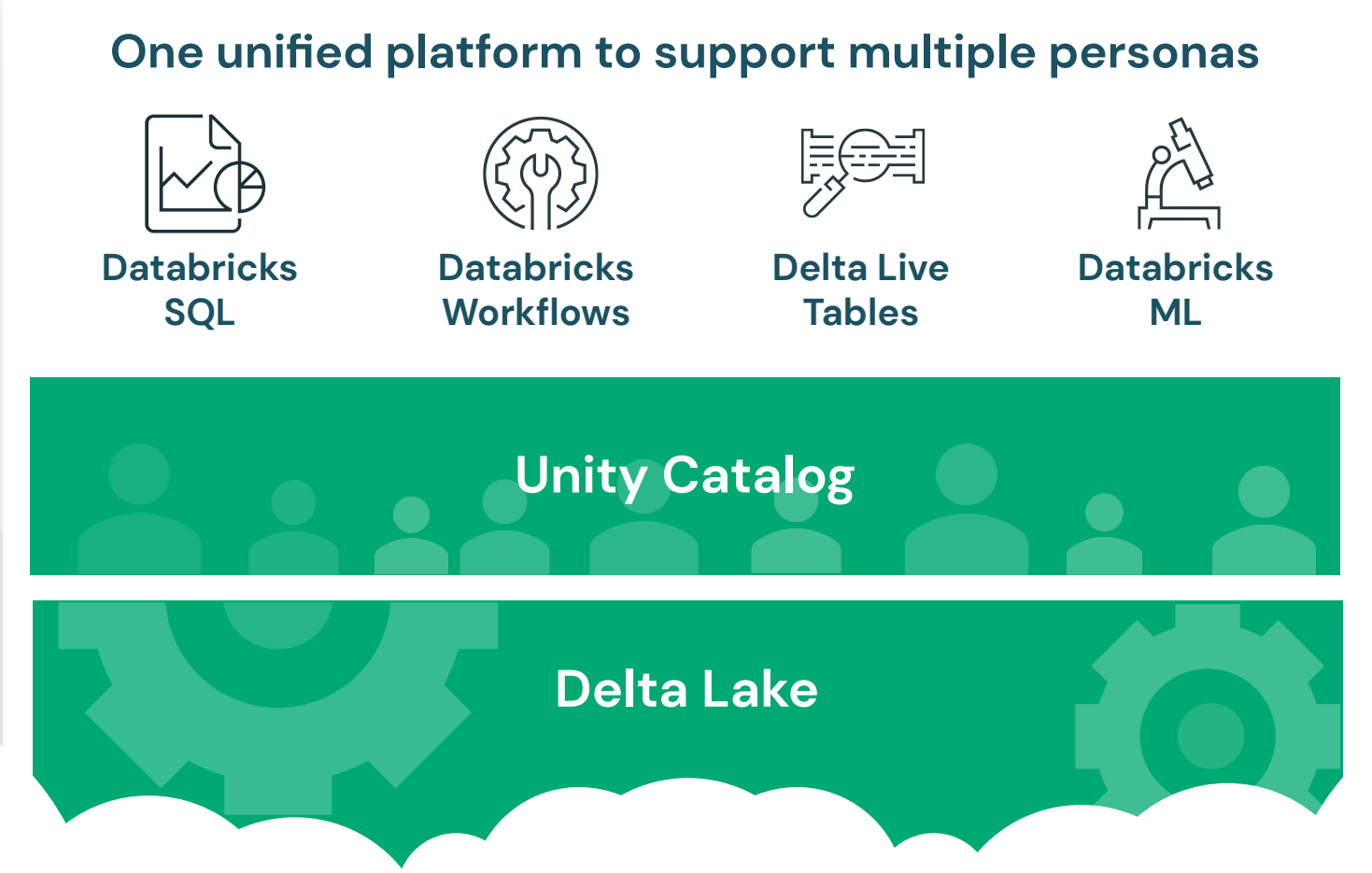
One security and governance model for all data access across the organization

One platform to store and manage all structured, semi-structured, and unstructured data

Cloud Data Lake

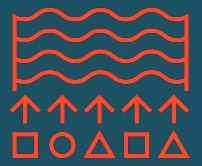
All Raw Data
(Logs, Texts, Audio, Video, Images)

Databricks delivers the only unified Lakehouse



Why Delta

Data Lake



DELTA LAKE

An open approach to bringing
data management and governance
to data lakes

Better reliability with transactions

48x faster data processing with indexing

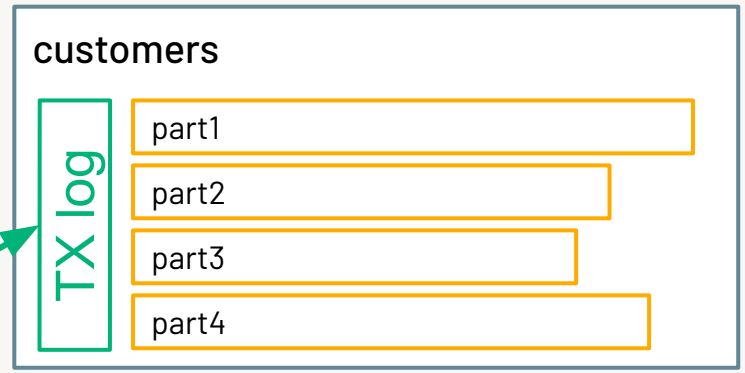
Data governance at scale with
fine-grained access control lists

Data Warehouse

“Why do I need Delta? I am perfectly happy with Parquet.”

- DELTA is NOT a file format. It is a PROTOCOL to govern how data is written and read in the data lakes!
- Your data still remains in Parquet! Delta is just a small log in JSON format next to those files
- What’s the **benefit**?
  - **UNLOCKS** the Databricks Lakehouse
  - **RELIABILITY**: ACID transactions, Time travel
  - **PERFORMANCE**: Integrated with Spark and Photon, Delta Caching out of box
  - **QUALITY**: Schema enforcement, RESTORE

Delta Tables



Transaction Log Provides Metadata Layer for Consistency

Data in Parquet

```
%fs ls /tmp/bernhard/loan_by_state_delta
```

path
dbfs:/tmp/bernhard/loan_by_state_delta/_delta_log/
dbfs:/tmp/bernhard/loan_by_state_delta/part-00000-cd80fd12-457b-4058-a904-3628c6e90a57-c000.snappy.parquet
dbfs:/tmp/bernhard/loan_by_state_delta/part-00004-69b2735a-18d1-474e-8318-ecd13ca410a7-c000.snappy.parquet
dbfs:/tmp/bernhard/loan_by_state_delta/part-00009-05545d85-f051-4a37-852a-18298fb4f3cd-c000.snappy.parquet
dbfs:/tmp/bernhard/loan_by_state_delta/part-00010-3fa88ba6-61cc-45cd-8c0d-e0949d1bbcce-c000.snappy.parquet

https://databricks.com/blog/2019/08/21/diving-into-delta-lake-unpacking-the-transaction-log.html

Delta Live Tables

Delta Live Tables

The best way to do ETL on the lakehouse

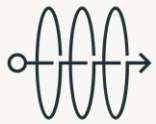
```
CREATE STREAMING TABLE raw_data
AS SELECT *
FROM cloud_files ("/raw_data", "json")
```

```
CREATE LIVE TABLE clean_data
AS SELECT ...
FROM LIVE.raw_data
```

Accelerate ETL development
Declare **SQL or Python** and DLT automatically orchestrates the DAG, handles retries, changing data

Automatically manage your infrastructure
Automates complex tedious activities like **recovery, auto-scaling, and performance optimization**

Ensure high data quality
Deliver reliable data with built-in **quality controls, testing, monitoring, and enforcement**



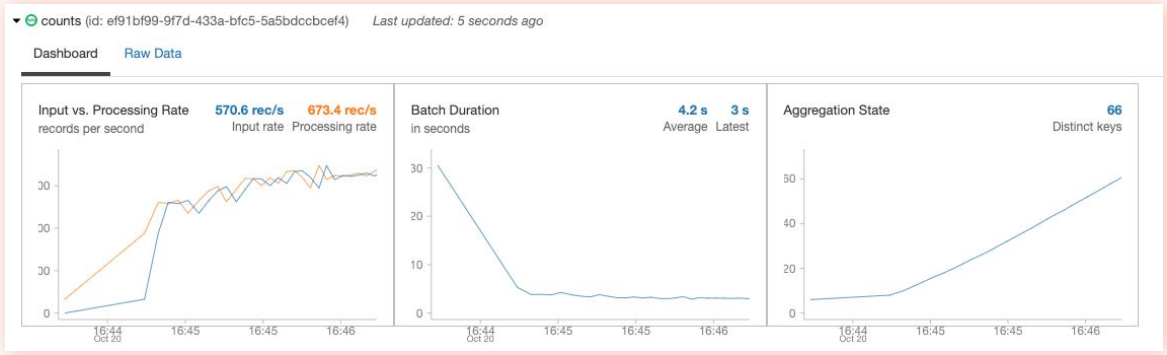
Unify batch and streaming
Get the simplicity of SQL with freshness of streaming with one **unified API**

Continuous or scheduled data ingestion

Simple SQL syntax for streaming ingestion

Cmd 2

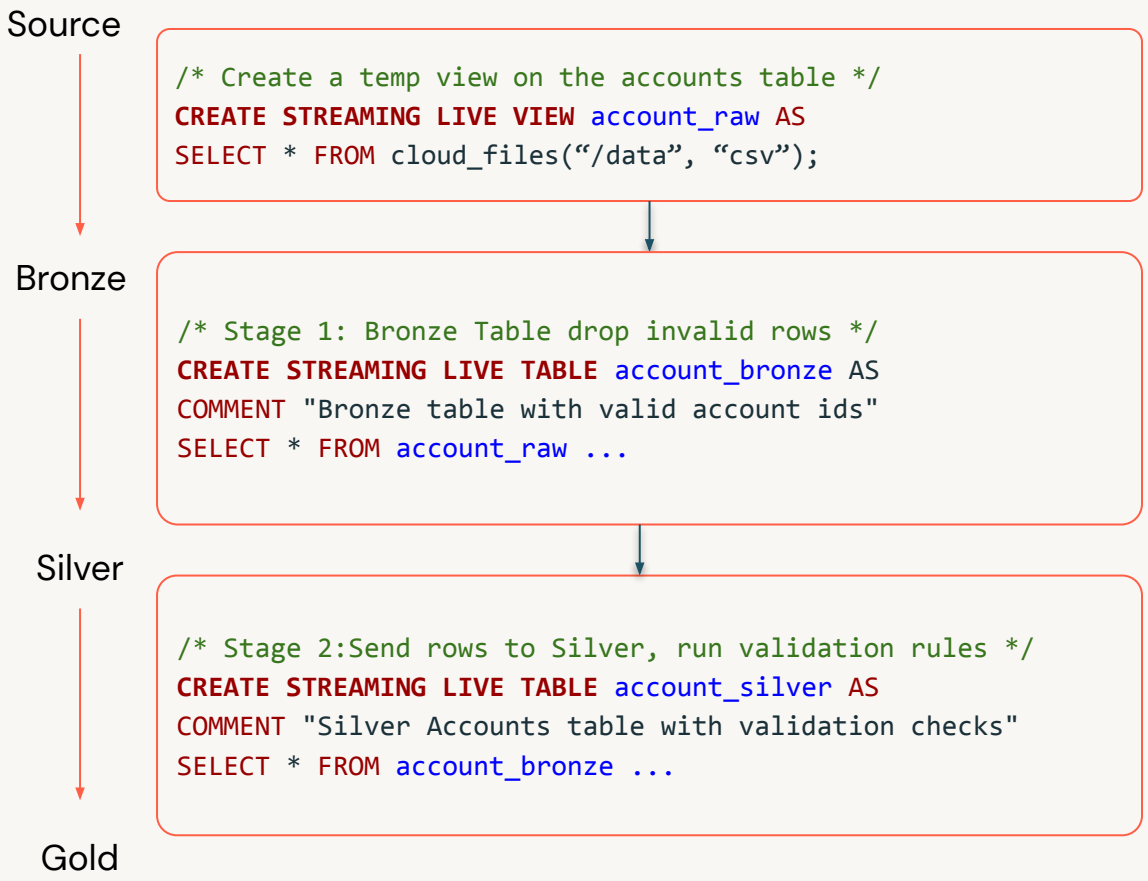
```
1  CREATE STREAMING LIVE TABLE sales_orders_raw
2  COMMENT "The raw sales orders, ingested from /databricks-datasets."
3  TBLPROPERTIES ("myCompanyPipeline.quality" = "bronze")
4  AS
5  SELECT * FROM cloud_files
6  ("/databricks-datasets/retail-org/sales_orders/",
7  "json", map("cloudFiles.inferColumnTypes", "true"))
8
```



- Incrementally and efficiently process new data files as they arrive in cloud storage using Auto Loader
- Automatically infer schema of incoming files or superimpose what you know with Schema Hints
- Automatic schema evolution
- Rescue data column – never lose data again

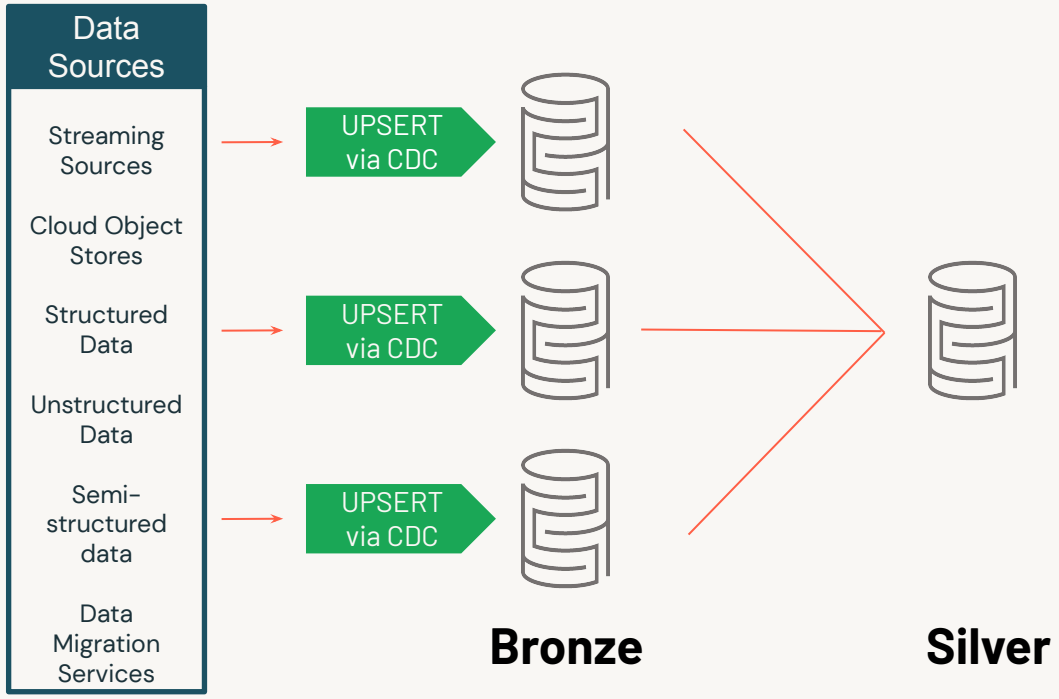
Schema Evolution	JSON	CSV	AVRO	PARQUET
	✅	✅	✅	✅

Declarative SQL & Python APIs



- Use intent-driven declarative development to abstract away the **"how"** and define **"what"** to solve
- Automatically generate **lineage** based on table dependencies across the data pipeline
- Automatically checks for errors, missing dependencies and syntax errors

Change Data Capture (CDC)

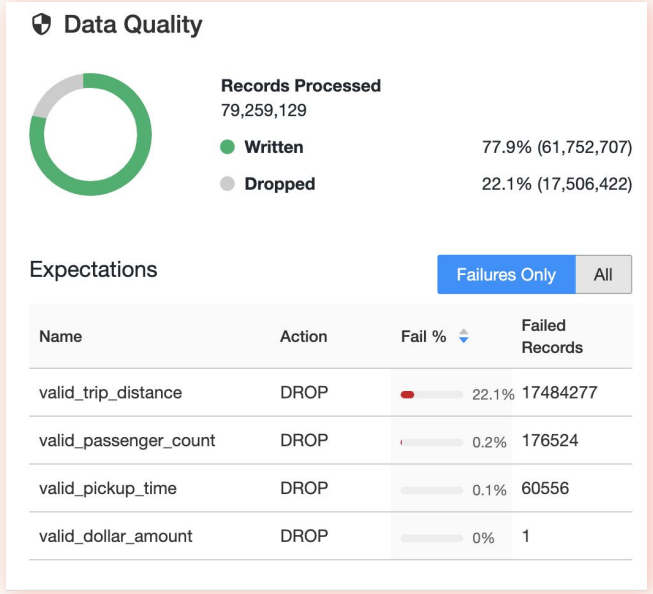


- Stream change records (inserts, updates, deletes) from any data source supported by DBR, cloud storage, or DBFS
- Simple, declarative "APPLY CHANGES INTO" API for SQL or Python
- Handles out-of-order events
- Schema evolution
- SCD2 support

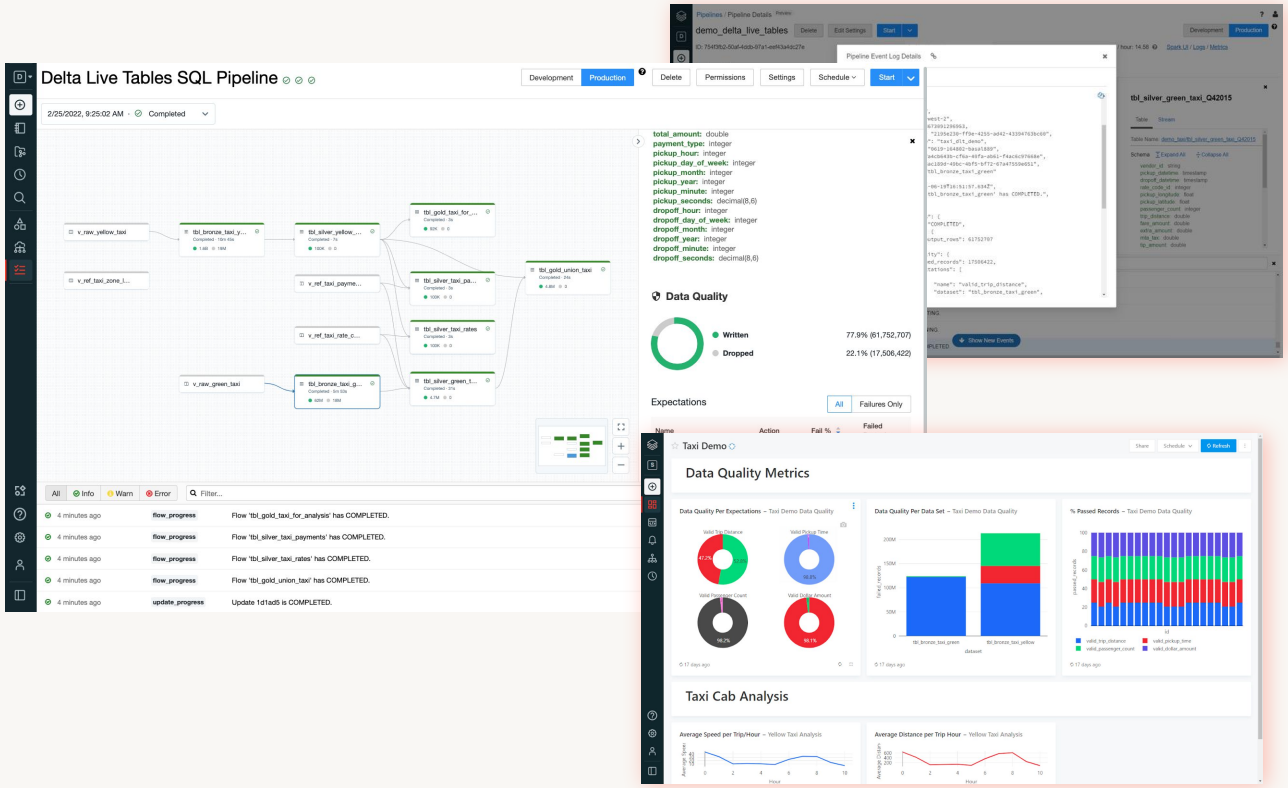
Data quality validation and monitoring

- Define data quality and integrity controls within the pipeline with data expectations
- Address data quality errors with flexible policies: fail, drop, alert, quarantine(future)
- All data pipeline runs and quality metrics are captured, tracked and reported

```
/* Stage 1: Bronze Table drop invalid rows */
CREATE STREAMING LIVE TABLE fire_account_bronze AS
( CONSTRAINT valid_account_open_dt EXPECT (acconut_dt is not
null and (account_close_dt > account_open_dt)) ON VIOLATION DROP
ROW
COMMENT "Bronze table with valid account ids"
SELECT * FROM fire_account_raw ...
```



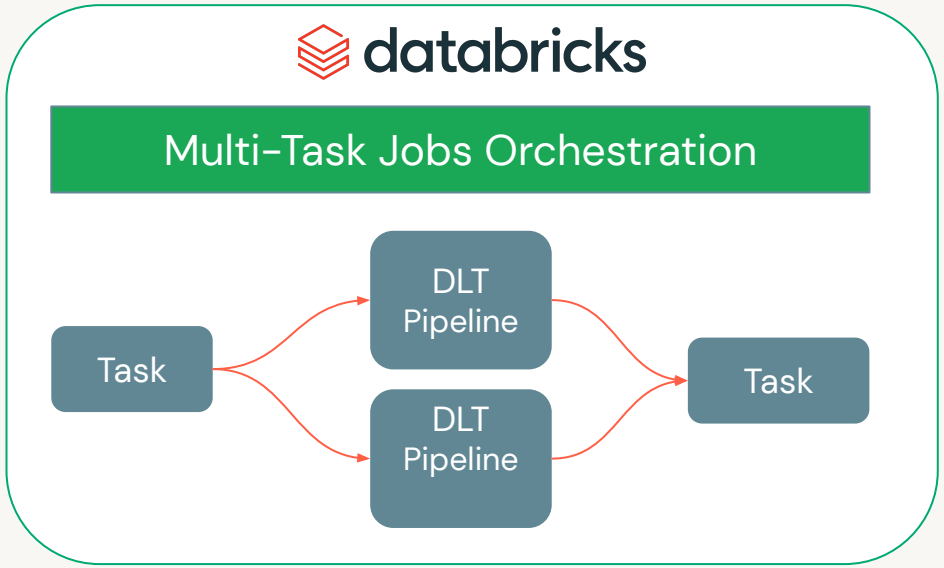
Data pipeline observability



- High-quality, high-fidelity lineage diagram that provides visibility into how data flows for impact analysis
- Granular logging for operational, governance, quality and status of the data pipeline at a row level
- Continuously monitor data pipeline jobs to ensure continued operation
- Notifications using Databricks SQL

Workflow Orchestration

Simplify orchestration and management of data pipelines



- Easily **orchestrate** DLT Pipelines and tasks in the same DAG
- **Fully integrated** in Databricks platform, making inspecting results, debugging faster
- Orchestrate and manage workloads in **multi-cloud environments**
- You can run a Delta Live Tables pipeline as part of a data processing workflow with **Databricks jobs, Apache Airflow, or Azure Data Factory.**

Delta Live Tables Exercise

Unity Catalog

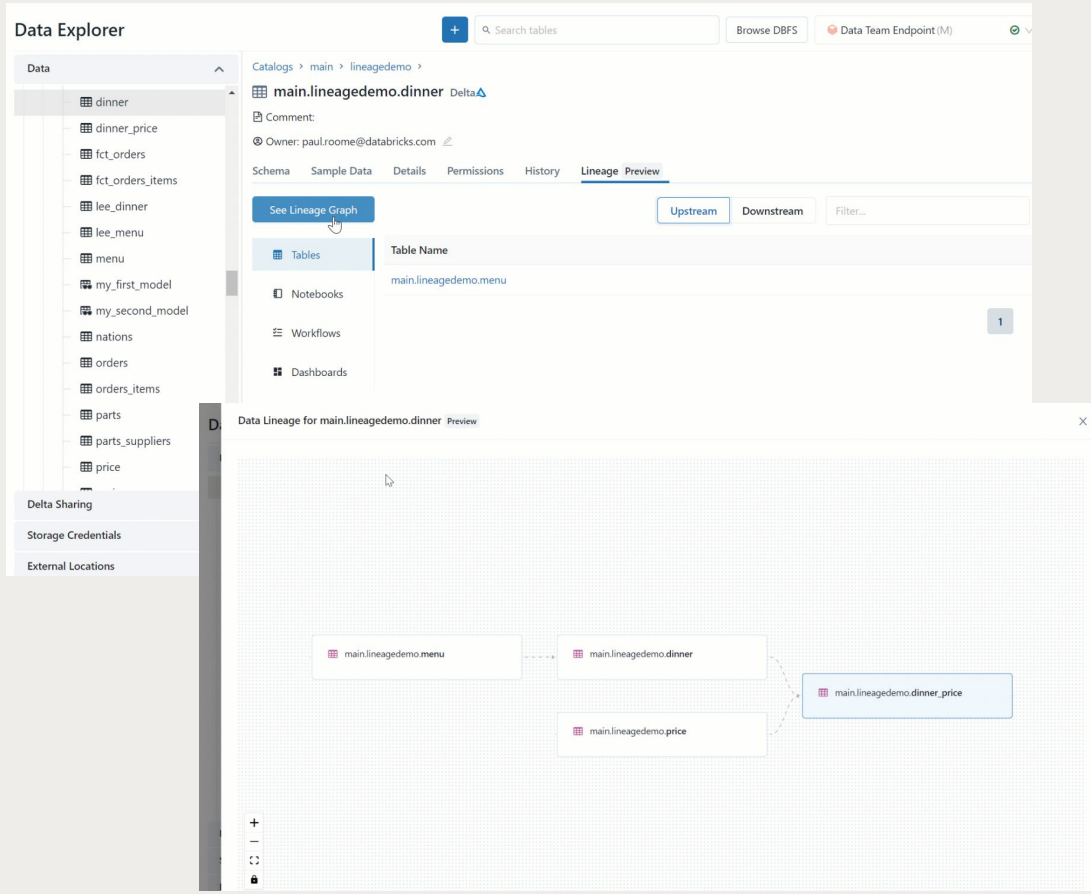
Unity Catalog for Lakehouse Governance

Govern and manage all data assets

- Warehouse, Tables, Columns
- Data Lake, Files
- Machine Learning Models
- Dashboards and Notebooks

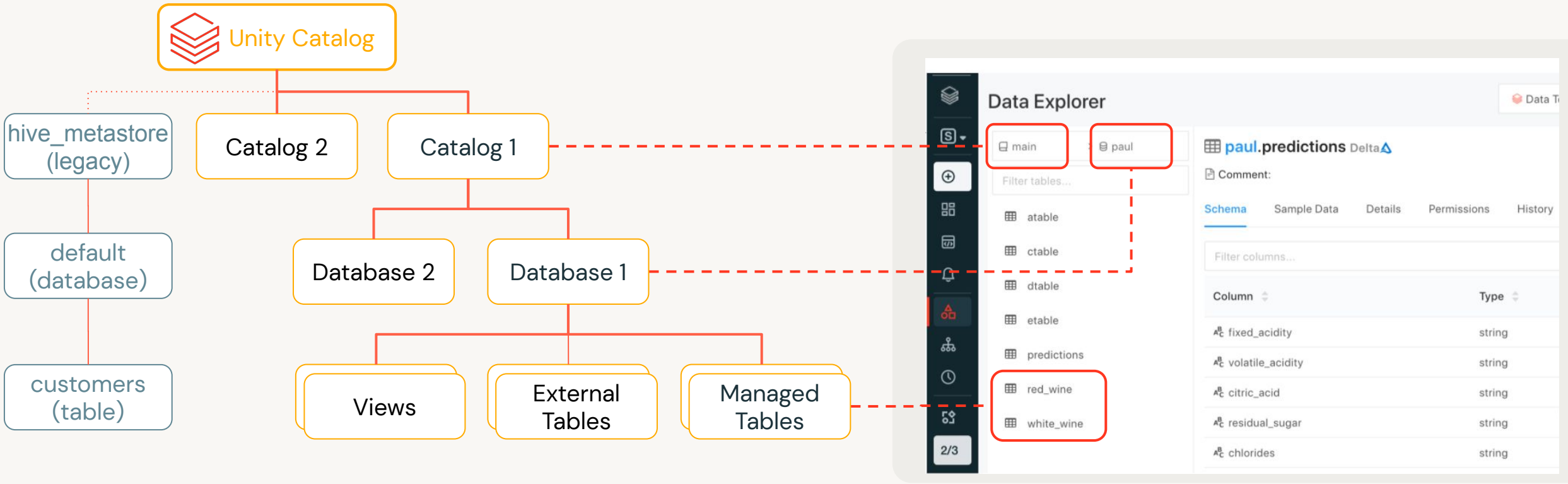
Capabilities

- Data lineage
- Attribute-based access control
- Security policies
- Table or column level tags
- Auditing
- Data sharing



Three level namespace

Seamless access to your existing metastores




```
SELECT * FROM main.paul.red_wine;  -- <catalog>.<database>.<table>

SELECT * FROM hive_metastore.default.customers;
```

Centralized Access Controls

Centrally grant and manage access permissions across workloads

Using ANSI SQL DCL

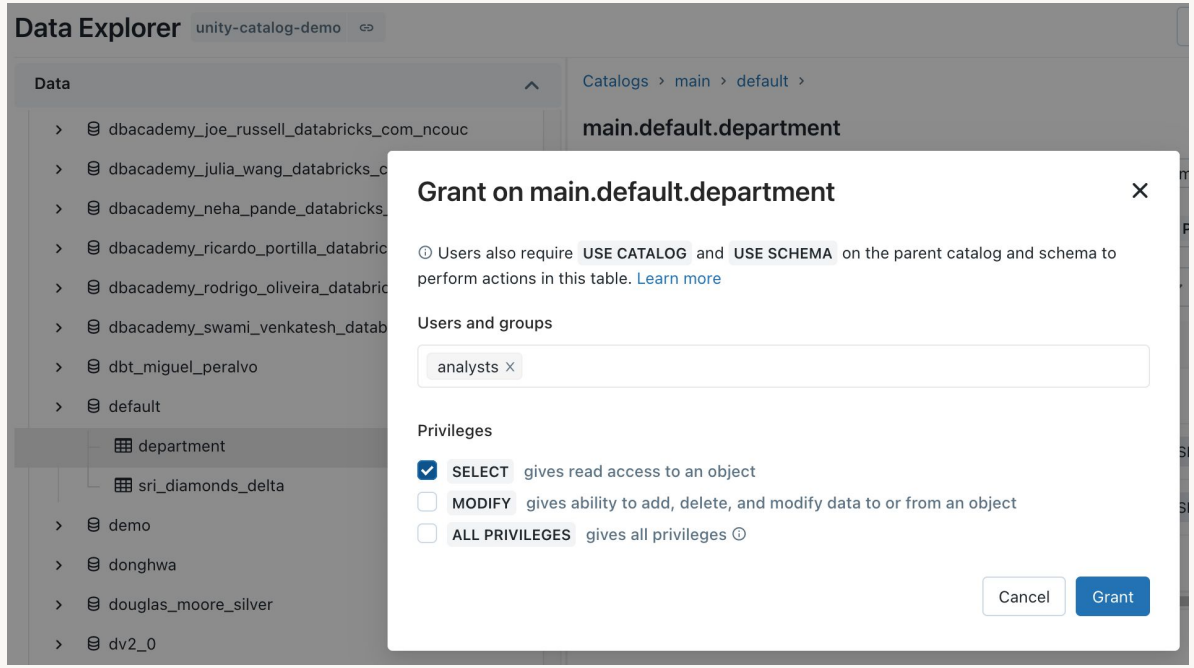
```
GRANT <privilege> ON <securable_type>
<securable_name> TO `<principal>`

GRANT SELECT ON iot.events TO engineers
```

Choose permission level

'Table'= collection of files in S3/ADLS

Sync groups from your identity provider

Using UI



Row Level Security and Column Level Masking

Provide differential fine grained access to datasets

Only show specific rows

```
CREATE FUNCTION <name> ( <parameter_name >
<parameter_type> .. )
RETURN {filter clause whose output must be a boolean}


CREATE FUNCTION us_filter(region STRING)
RETURN IF(IS_MEMBER('admin'), true, region="US");

ALTER TABLE sales SET ROW FILTER us_filter ON region;
```

Test for group membership

Assign reusable filter to table

Specify filter predicates

Mask or redact sensitive columns

```
CREATE FUNCTION <name> (<parameter_name>,
<parameter_type>, [, <column>...])
RETURN {expression with the same type as the first
parameter}

CREATE FUNCTION ssn_mask(ssn STRING)
RETURN IF(IS_MEMBER('admin'), ssn, "****");

ALTER TABLE users ALTER COLUMN table_ssn SET MASK
ssn_mask;
```

Test for group membership

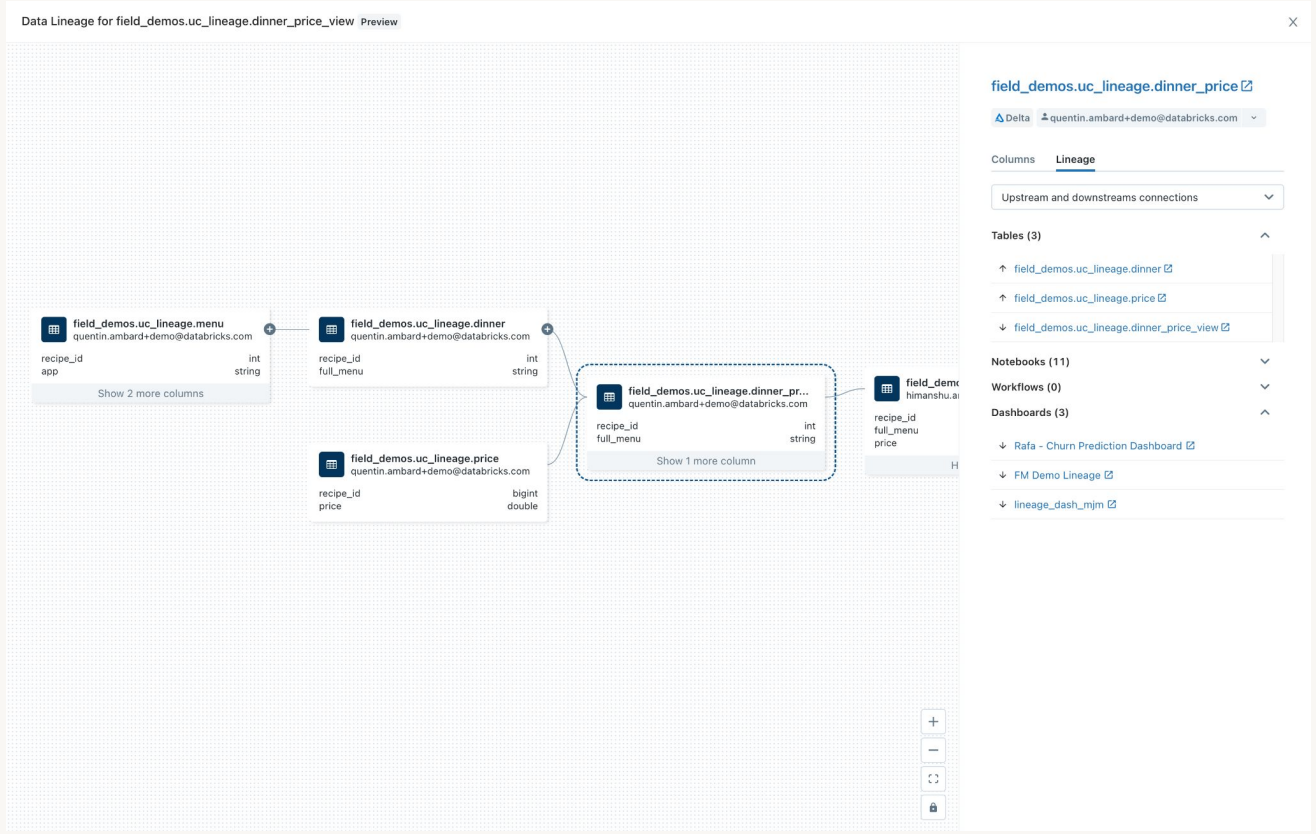
Assign reusable mask to column

Specify mask or function to mask

Automated lineage for all workloads

End-to-end visibility into how data flows and consumed in your organization

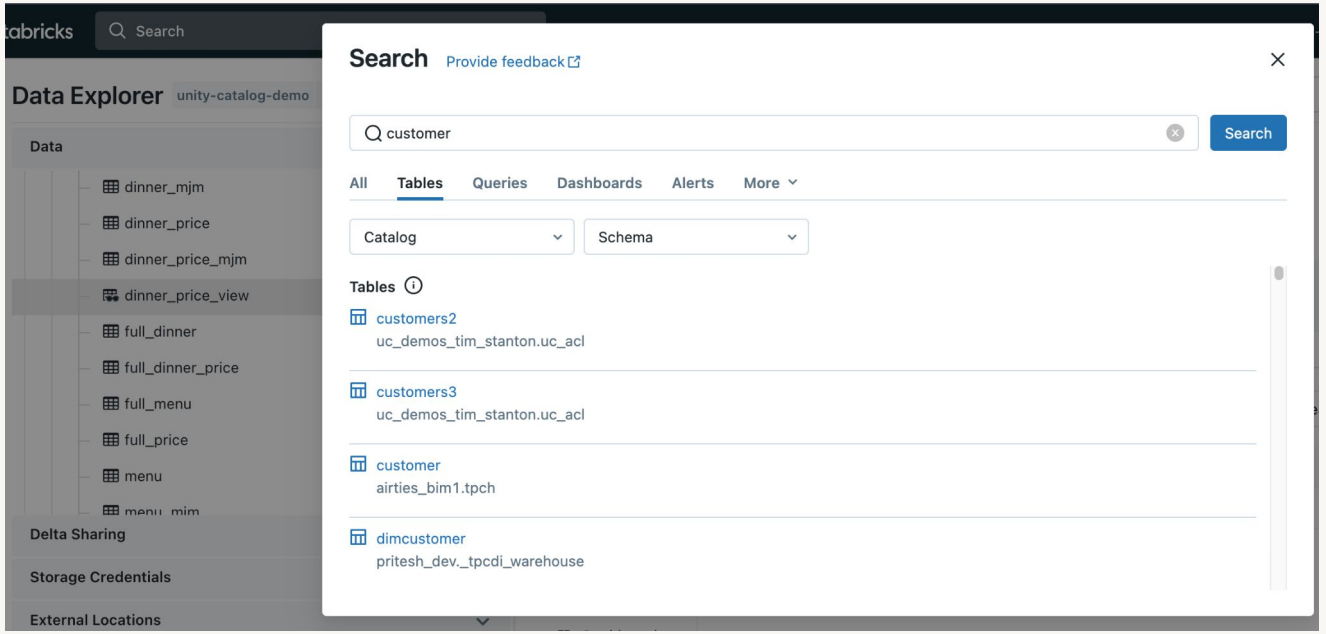
- Auto-capture runtime data lineage on a Databricks cluster or SQL warehouse
- Track lineage down to the table and column level
- Leverage common permission model from Unity Catalog
- Lineage across tables, dashboards, workflows, notebooks, feature tables, files, and DLT



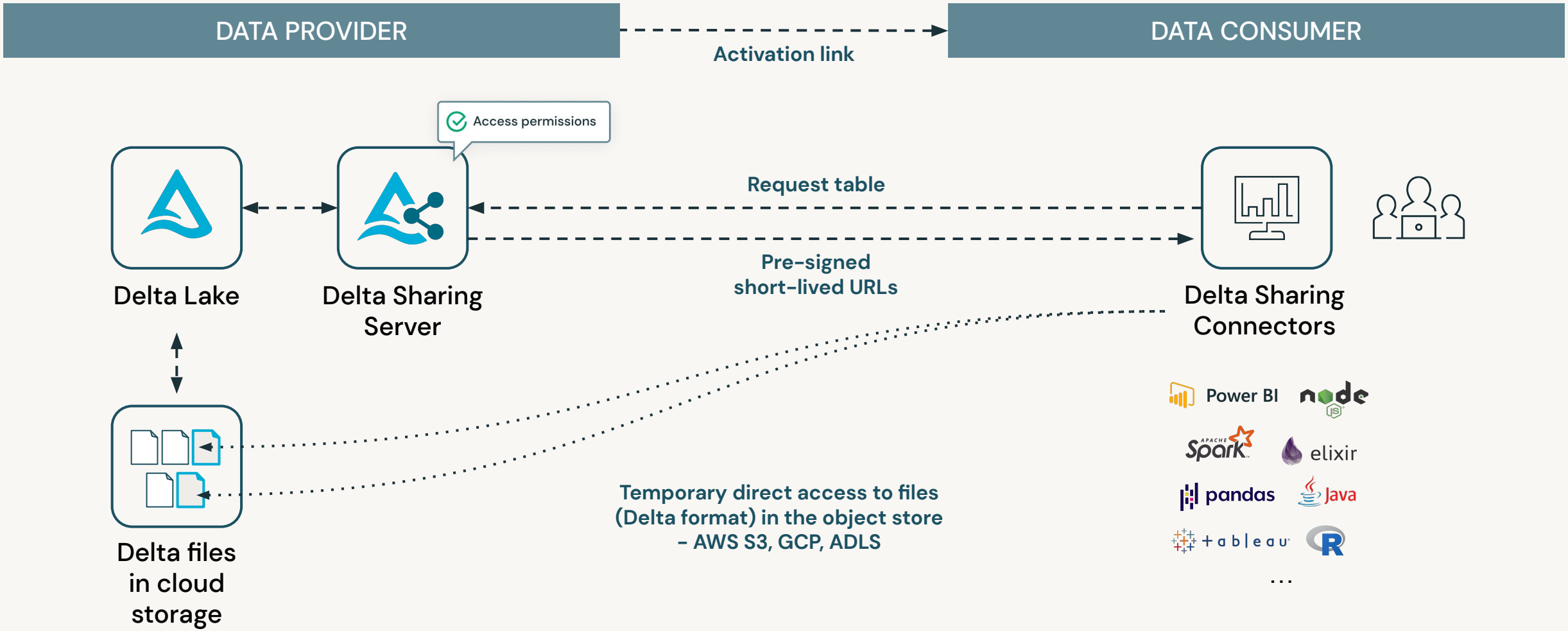
Built-in search and discovery

Accelerate time to value with low latency data discovery

- UI to search for data assets stored in Unity Catalog
- Unified UI across DSML + DBSQL
- Leverage common permission model from Unity Catalog
- Apply semantic tags to data and search across tags



Delta Sharing



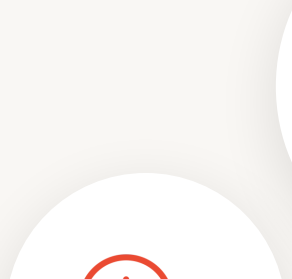
Unity Catalog Exercise

Databricks SQL

Warehouses can’t keep up with today’s world

Streaming, ML and AI require a new architecture

- Data Warehouses can’t keep up with the **variety** of data and use cases
- Innovation hinges on integrating **ML/AI** and predictive insights
- Business agility requires reliable, **real-time data**
- Data is **locked-in** and **duplicated**
- Not **cost** effective, especially at **scale**



Data Warehousing on the Lakehouse

Powered by Databricks SQL

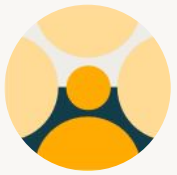
Databricks SQL (DB SQL) is a serverless data warehouse on the Databricks Lakehouse Platform that lets you run all your SQL and BI applications at scale with up to 12x better price/performance, a unified governance model, open formats and APIs, and your tools of choice - no lock-in.

Best price/performance

Lower costs, get world-class **performance**, and eliminate the need to manage, configure or scale cloud infrastructure with **serverless**.

Built-in governance

Establish one single copy for all your data using **open standards**, and one **unified governance layer** across all data teams using standard **SQL**.



Rich ecosystem

Use SQL and any tool like Fivetran, dbt, PowerBI, or Tableau along with Databricks to ingest, transform and query all your data in-place.

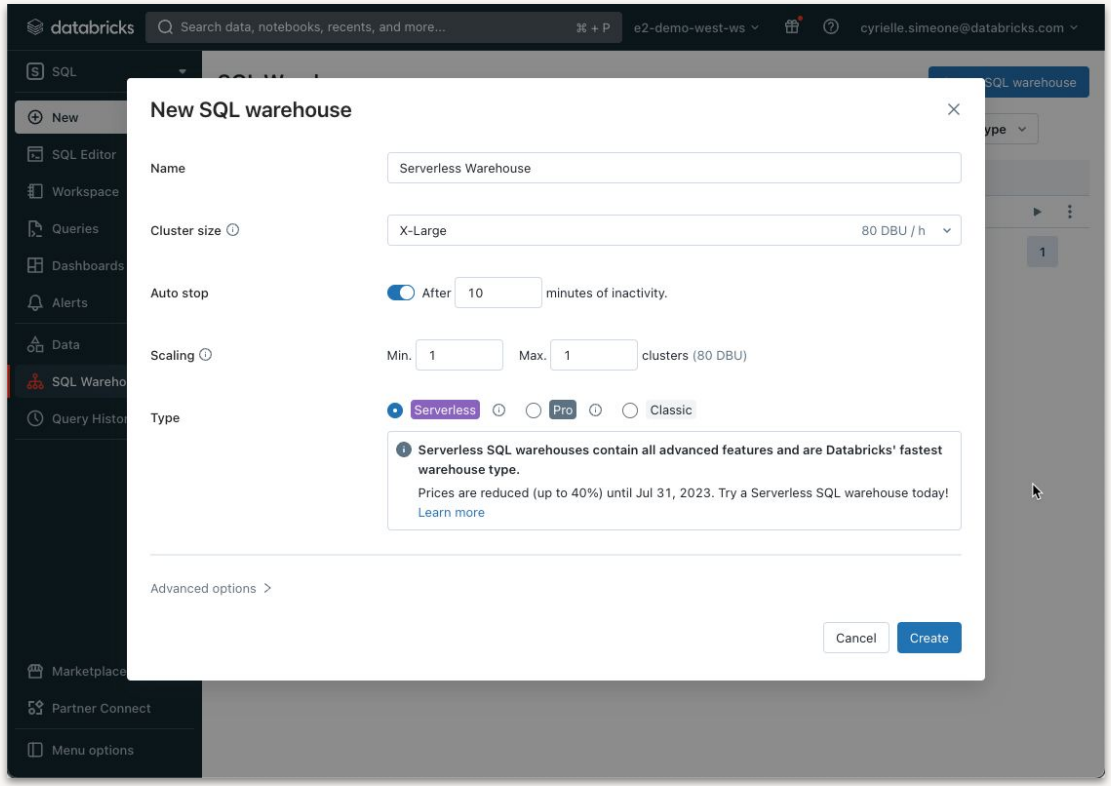
Break down silos

Empower every analyst to access the latest data faster for downstream **real-time analytics**, and go effortlessly from BI to ML.

Serverless compute for Databricks SQL

Instant, elastic & zero-management compute

- Quickly setup **instant**, **elastic SQL warehouse** - decoupled from storage - Powered by Photon
- Automatically determines instance types and configuration for best price/perf (up to 12x)
- **High concurrency** built-in, automatic load balancing
- Intelligent **workload management** and faster reads from cloud storage
- Instant startup, greater availability, and **40% average lower overall costs with serverless**



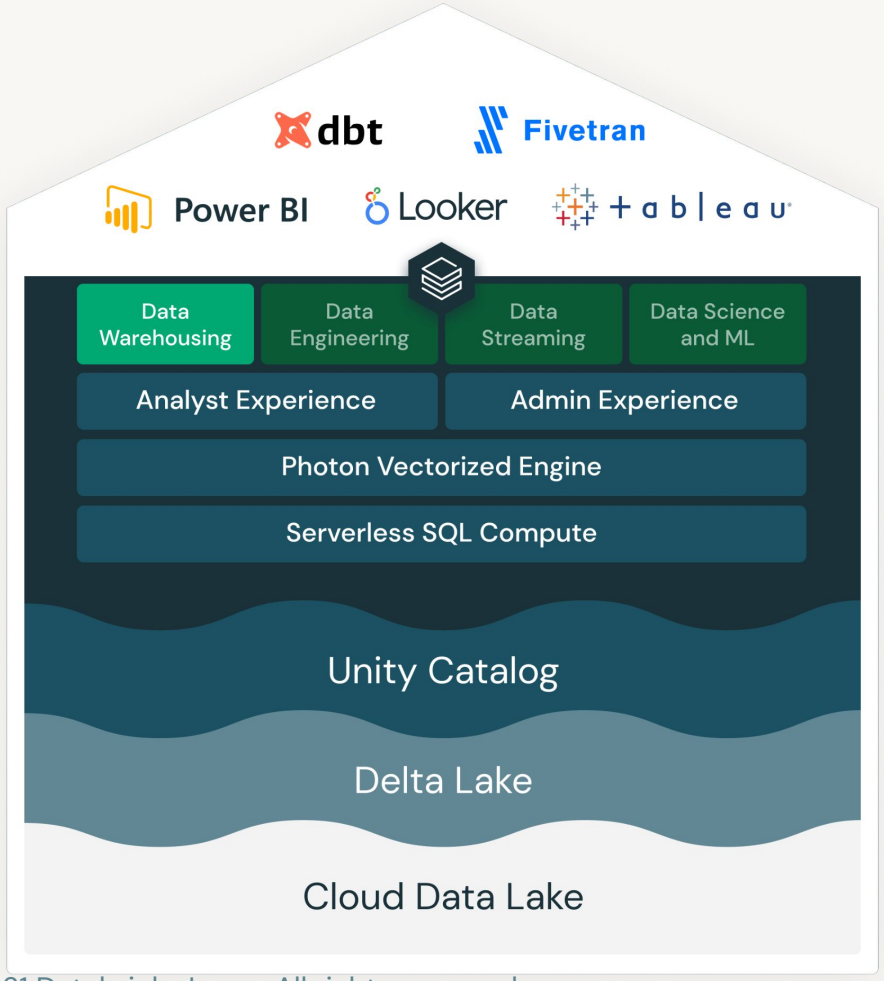
aws Public Preview!

Public Preview!

Coming Soon

The best data warehouse is a lakehouse

Powered by Databricks SQL



Seamless Integration with the Ecosystem

Ease of Use

Real-world Performance

Centralized Governance

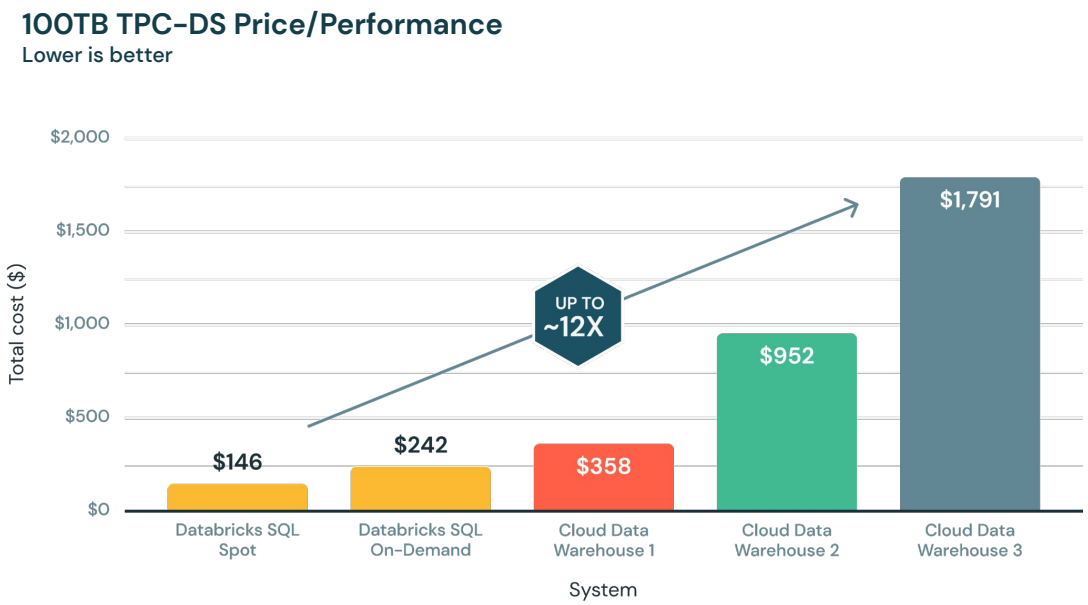
Open and Reliable Data Lake as the Foundation

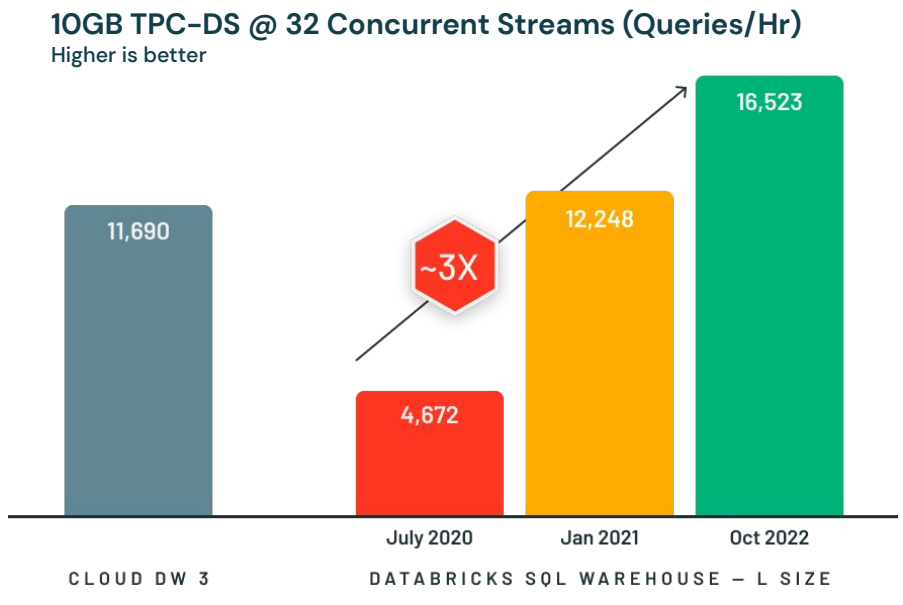
SQL workloads on Databricks

- Great performance and concurrency for BI and SQL workloads on Delta Lake
- Native SQL interface for analysts
- Support for BI tools to directly query your most recent data in Delta Lake

Built from the ground up for best performance

Lightning fast analytics for all queries

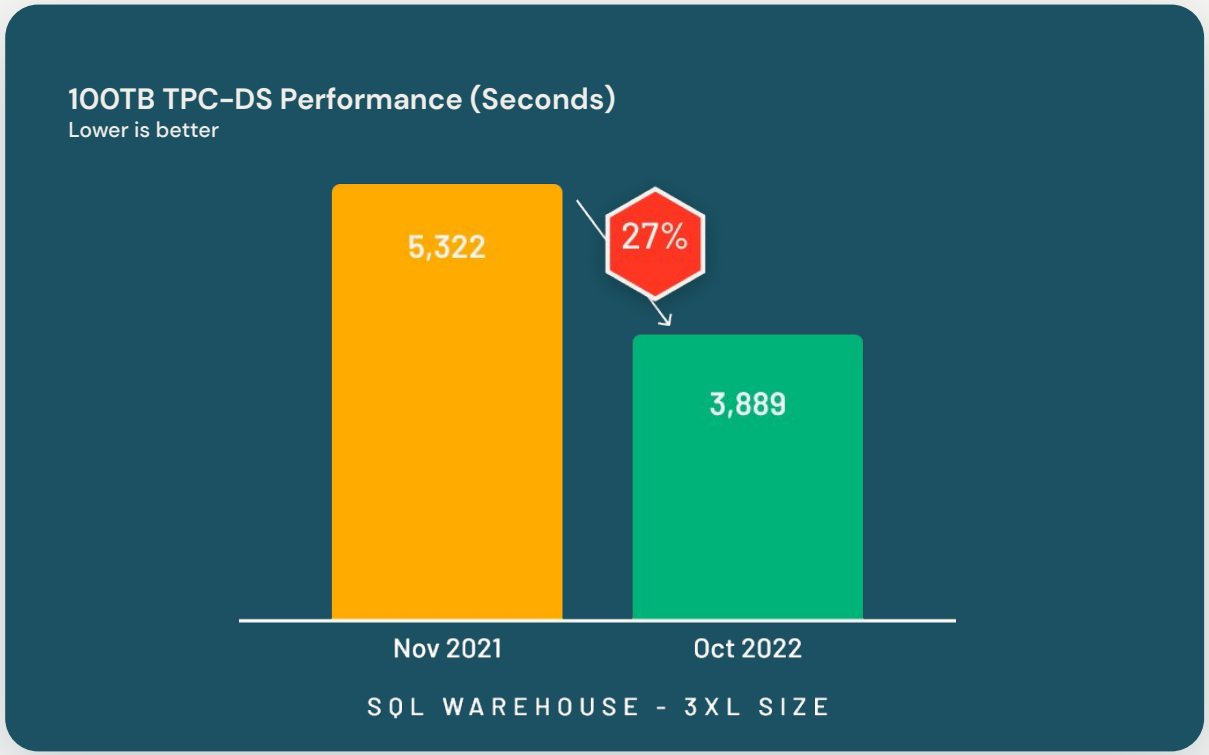




Databricks sets official data warehousing performance record
Learn more: **https://dbricks.co/benchmark**

Continuously Improving Performance

The same workload faster one year later



Databricks SQL is best with Serverless

Improved performance at lower cost

Users

Instant, Elastic Compute

Fastest query execution with instant compute

Scale fast and intelligently

Admins

Zero Management

Managed environment: capacity & pools managed by Databricks

Simple and predictable pricing model

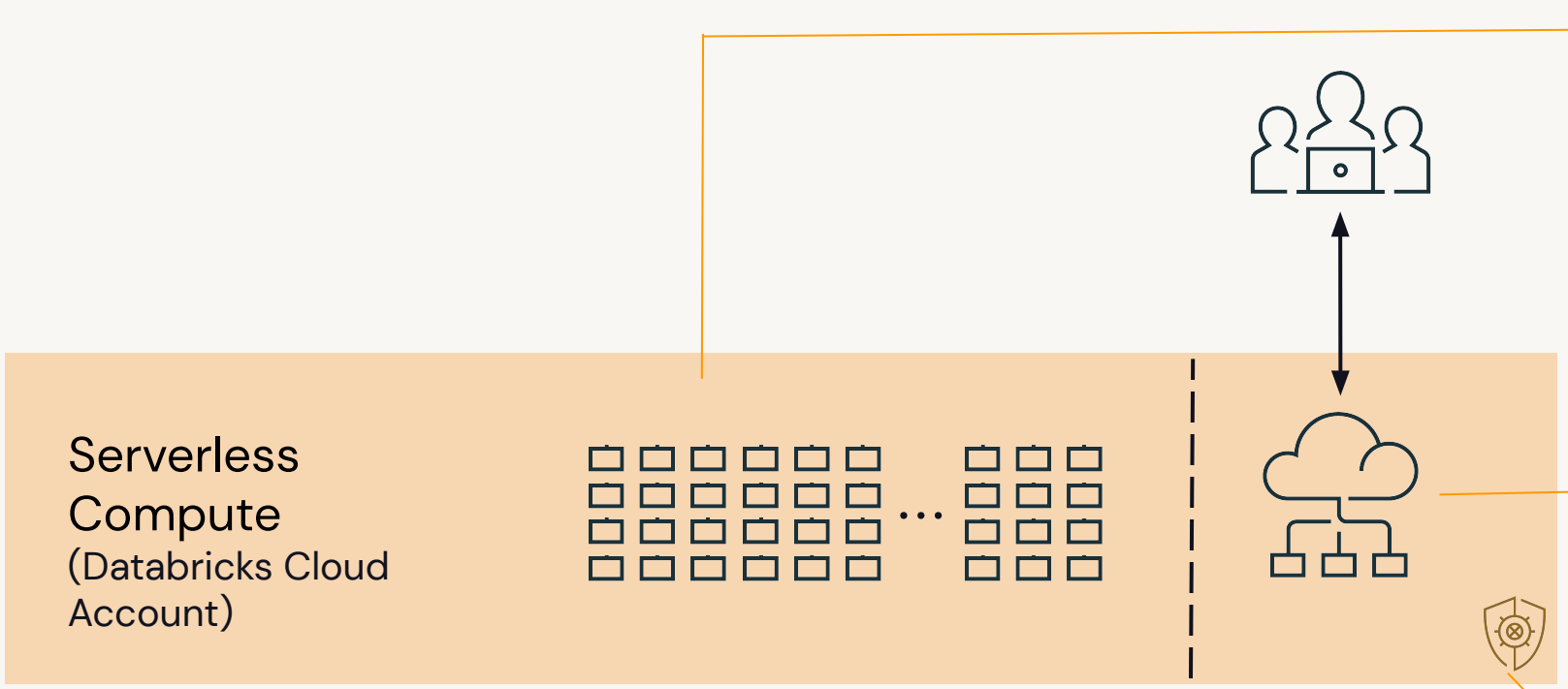
IT Budget

Lower TCO

Reduce idle time

Avoid over-provisioning

Databricks SQL Serverless



Managed Servers

Always-running server fleet
Patched and upgraded automatically

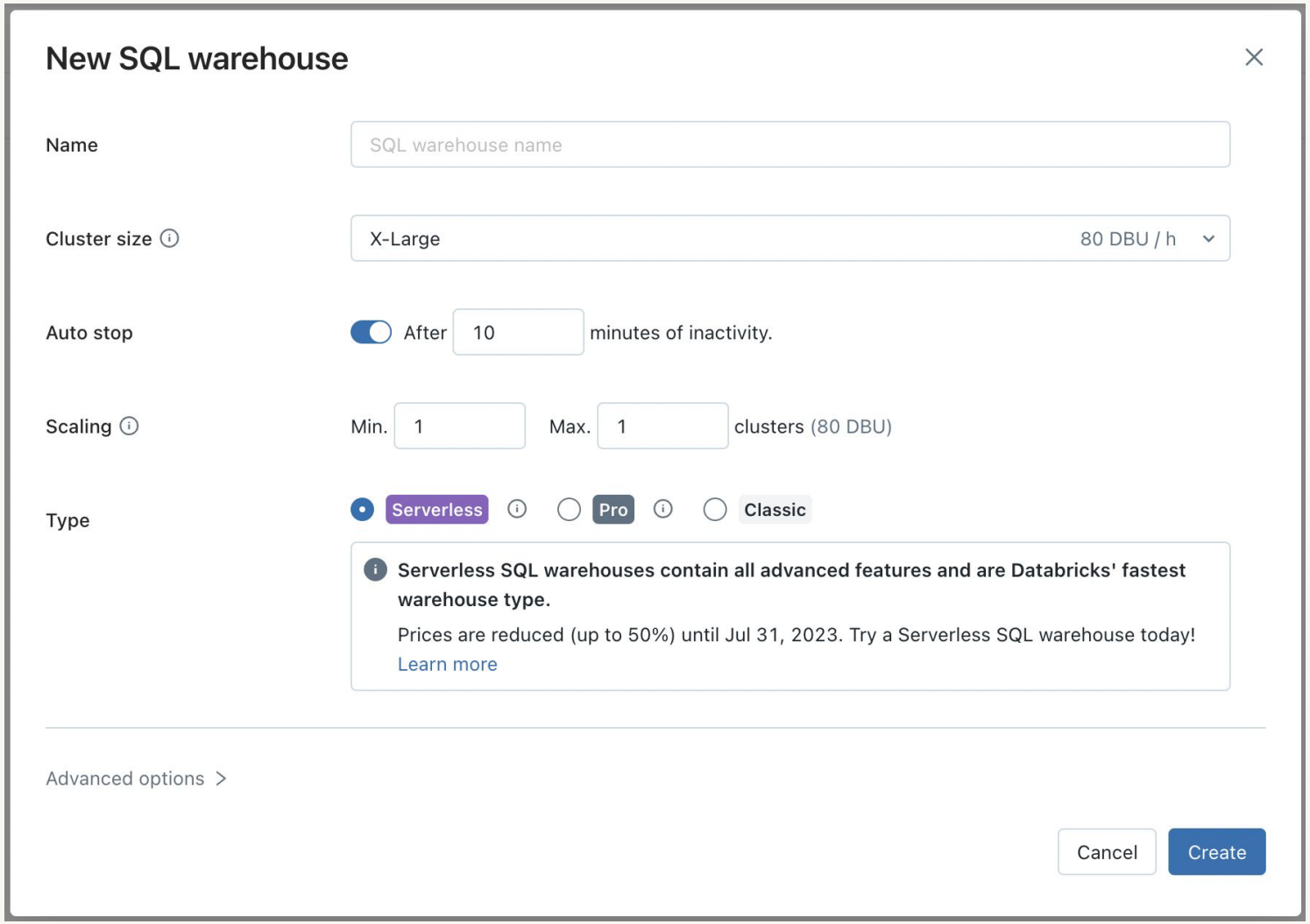
Elastic

Scale up and down automatically

Secure

3-layer isolation with data encryption

Serverless SQL – Simple to Configure



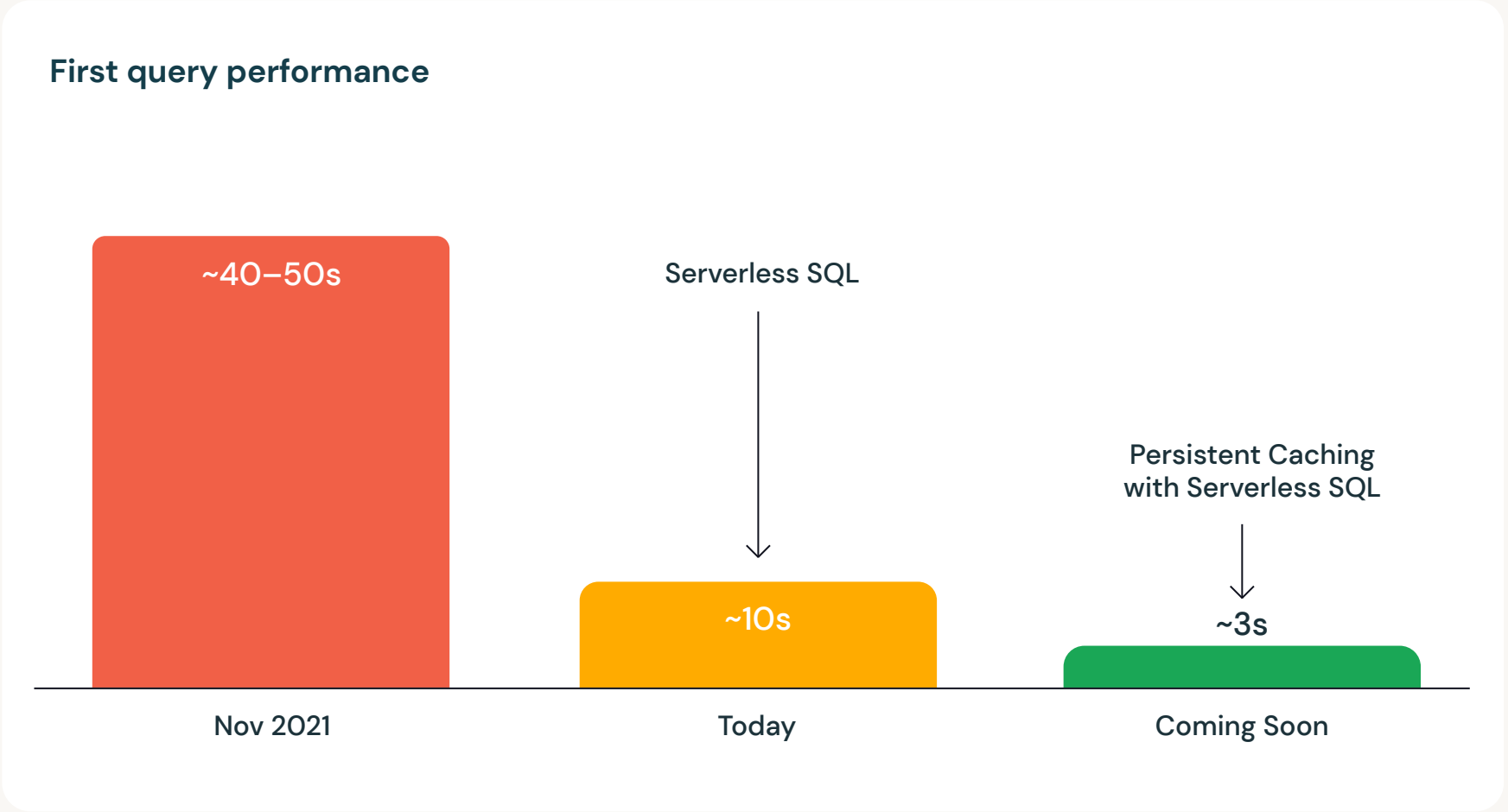
Azure Cluster Sizes for SQL Warehouses

Cluster size	Instance type for driver (applies only to pro and classic SQL warehouses)	Worker count
2X-Small	Standard_E8ds_v4	1 x Standard_E8ds_v4
X-Small	Standard_E8ds_v4	2 x Standard_E8ds_v4
Small	Standard_E16ds_v4	4 x Standard_E8ds_v4
Medium	Standard_E32ds_v4	8 x Standard_E8ds_v4
Large	Standard_E32ds_v4	16 x Standard_E8ds_v4
X-Large	Standard_E64ds_v4	32 x Standard_E8ds_v4
2X-Large	Standard_E64ds_v4	64 x Standard_E8ds_v4
3X-Large	Standard_E64ds_v4	128 x Standard_E8ds_v4
4X-Large	Standard_E64ds_v4	256 x Standard_E8ds_v4

Aggressively optimizing away idle costs

Improved first query warm-up

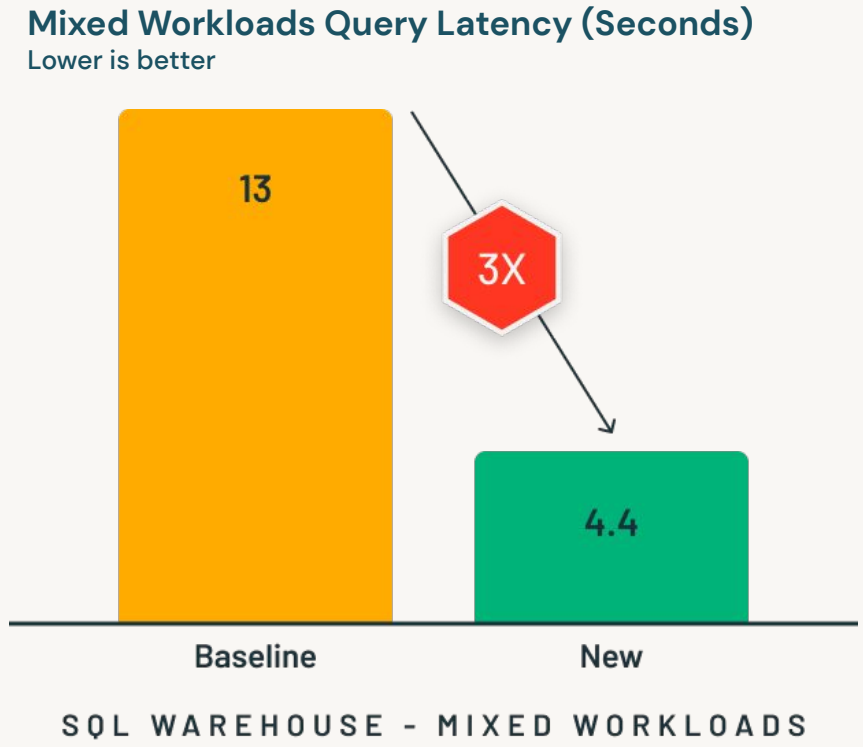
Databricks SQL Serverless provides **instant**, secure & zero-management compute.



AI Powered Simplicity

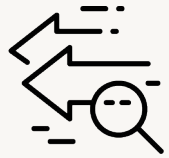
Intelligent workload management

- We have combined the elasticity of serverless with our years of experience building AI systems to enable **Intelligent Workload Management**
- Databricks SQL learns from the history of your workloads
- We use this history for new queries to determine if we should prioritize it to run immediately or scale up to run it without disrupting running queries.

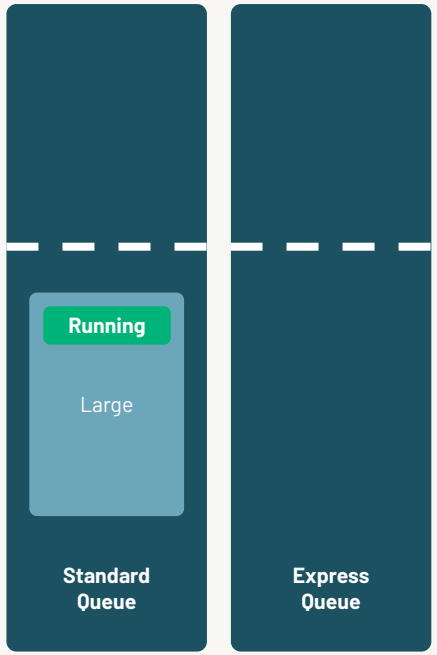


Adaptive Query Optimization

Out of the box performance for short queries

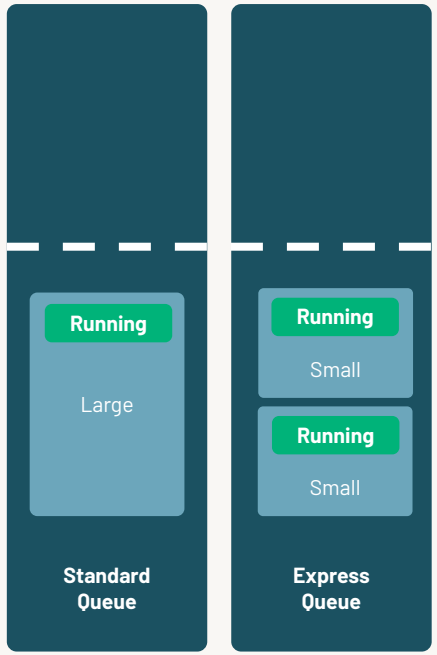


Dual queues **avoid large queries to block small ones**



5:00 PM

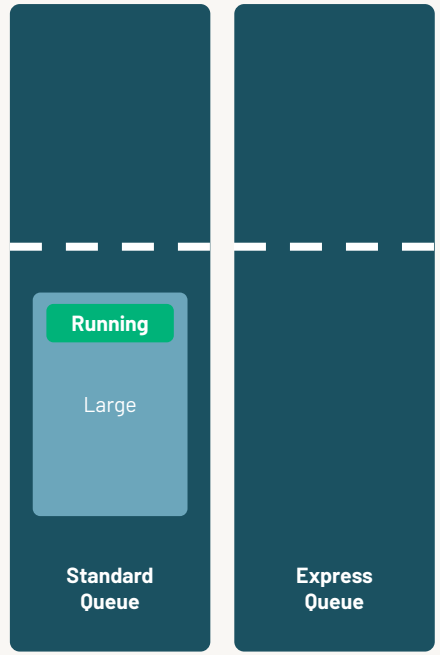
Large query in progress



5:05 PM

Large query in progress

New small queries immediately in-progress



5:05:10 PM

Large query in progress

Small queries complete!

An open and flexible warehouse

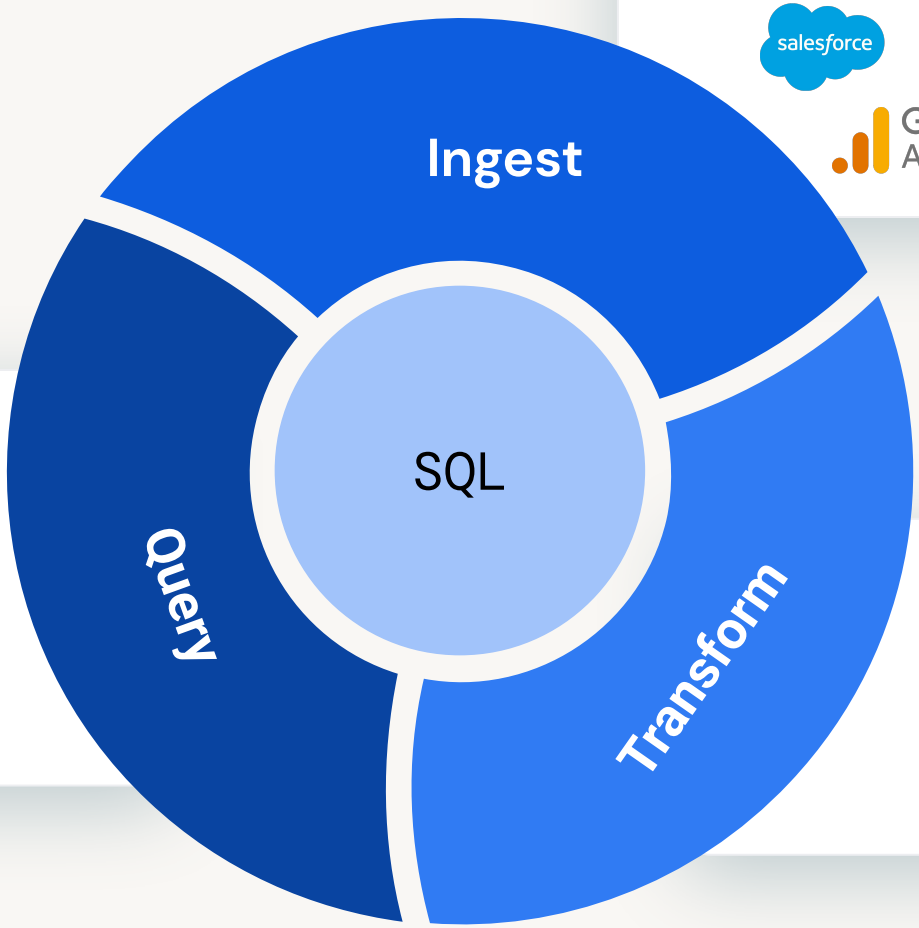
Use tools you already know – no lock in

Fivetran

salesforce

Google Analytics

Self-served data ingestion from **cloud storage**, **local files**, or **business critical applications**



Query the freshest data in SQL, and build **apps** and **dashboards** with **any tools** powered by the lakehouse

tableau

Power BI

python

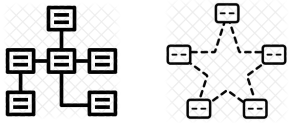
{REST:API}

node

GO

A **familiar toolkit** to discover and transform all your data in-place using **standard SQL**

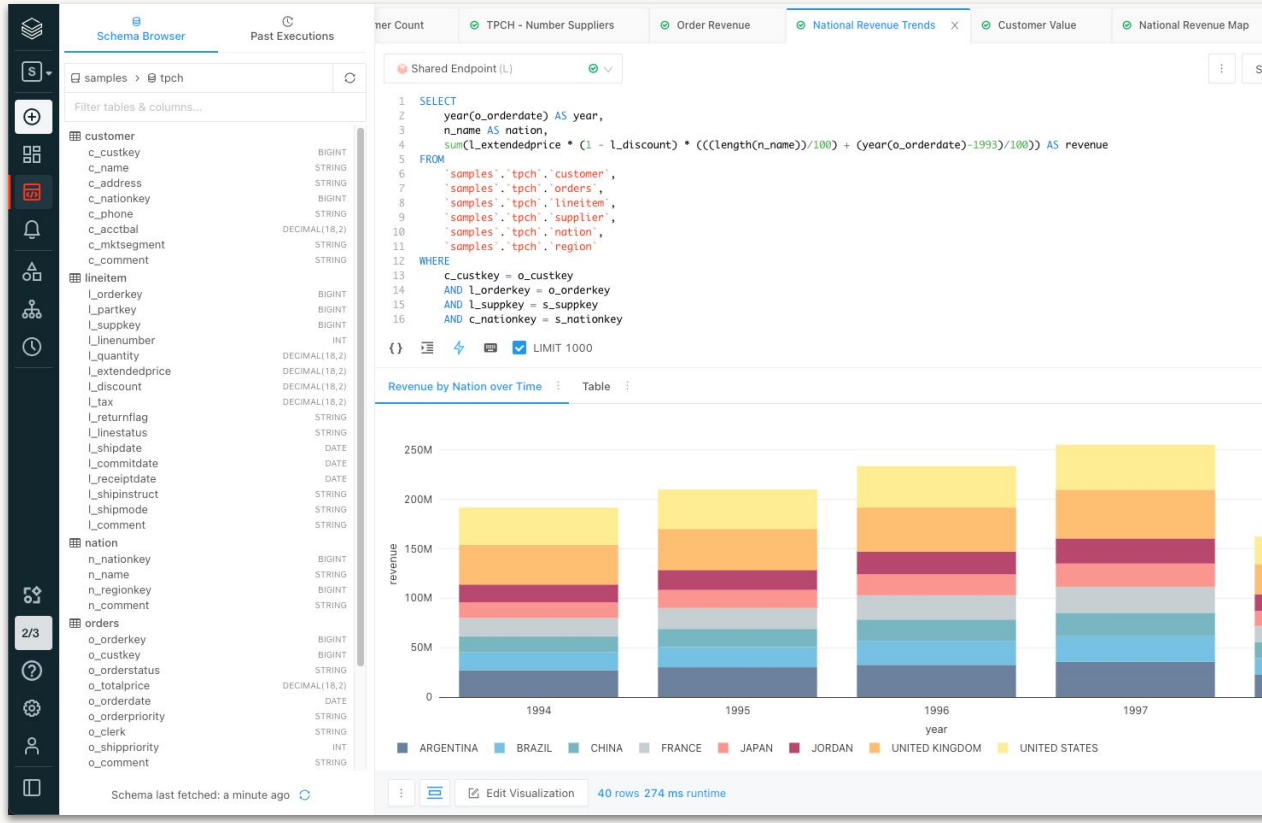
dbt



Databricks SQL Query Editor

Collaboratively query, explore, and transform data in-place

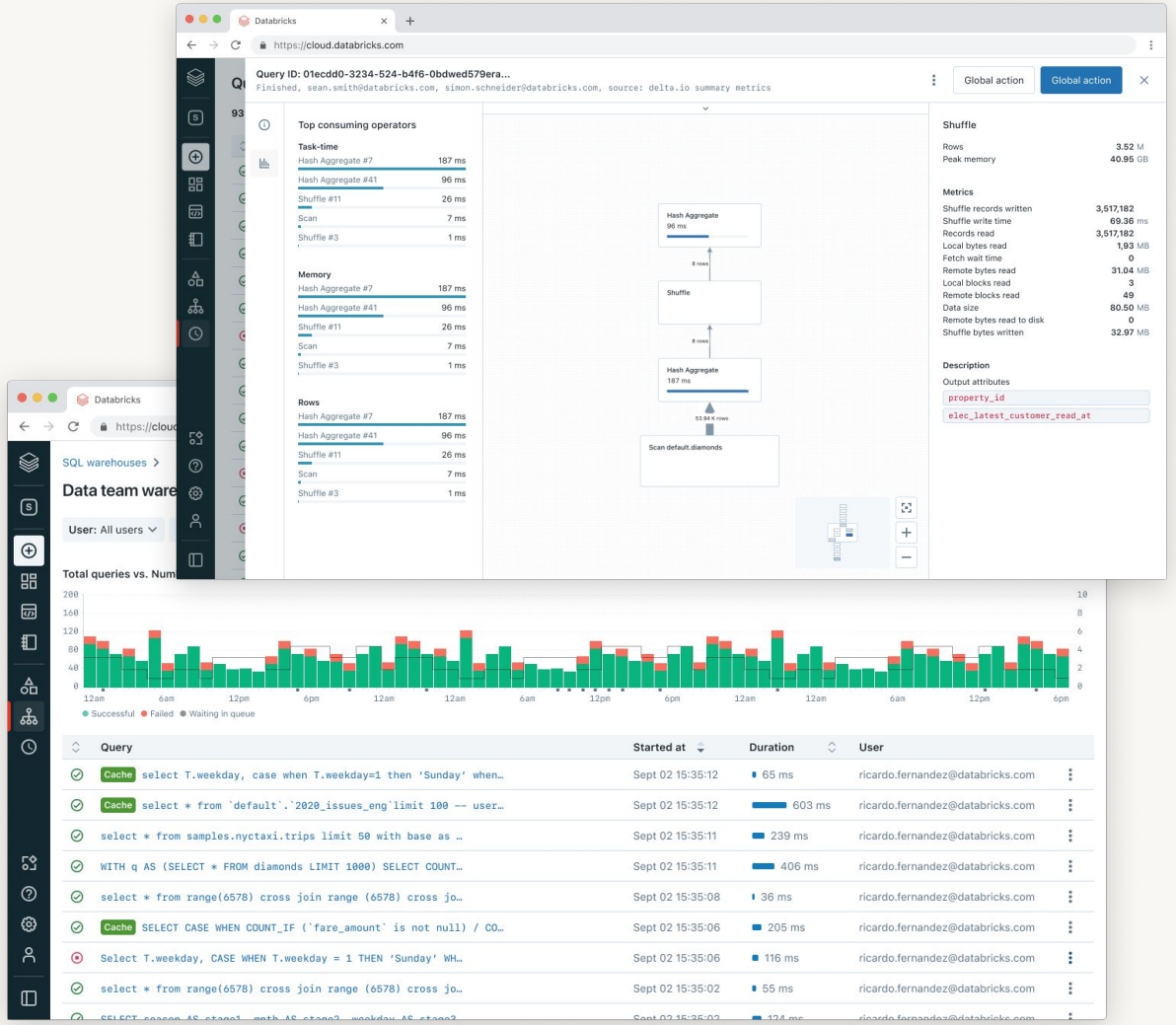
- Easily **ingest** data from cloud storage, local files, or business applications using Fivetran
- Discover data, explore database schema, and query data using **ANSI SQL**
- Save, share, and reuse queries across teams to get to results faster
- Orchestrate queries, alerts, and dashboards with automated **workflows**
- Stay up to date with alerts and automatic refresh schedules



Admin Experience

Monitoring & workload management

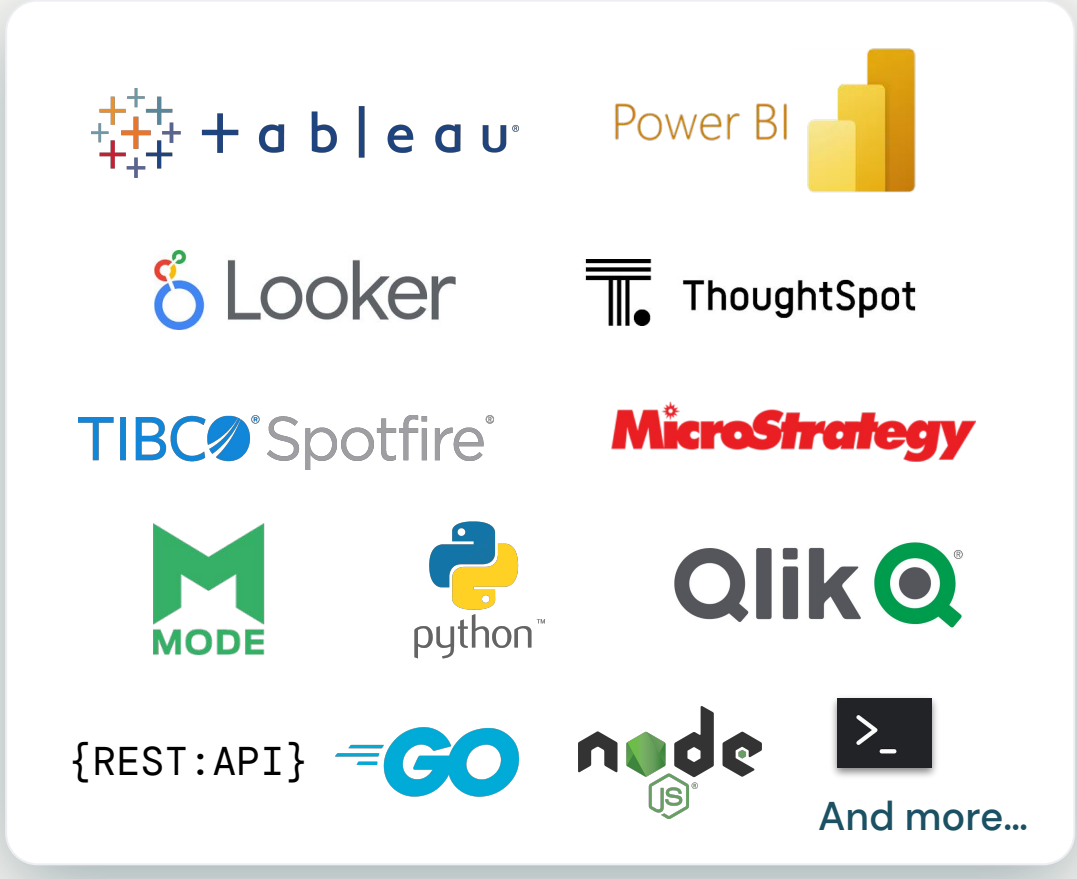
- Get full transparency and visibility into query execution with in-depth breakdown at operation level details
- Identify bottlenecks and expensive operations to enhance queries
- Analyze by time spent, data volume and resource usage
- Track & understand usage across virtual clusters, users & time



Data Consumption

Query from any tool

Connect existing dashboards or brand new ones to the latest and freshest data, leverage your favorite tools to find new insights, and build custom data apps powered by the lakehouse with tools and languages you already know.



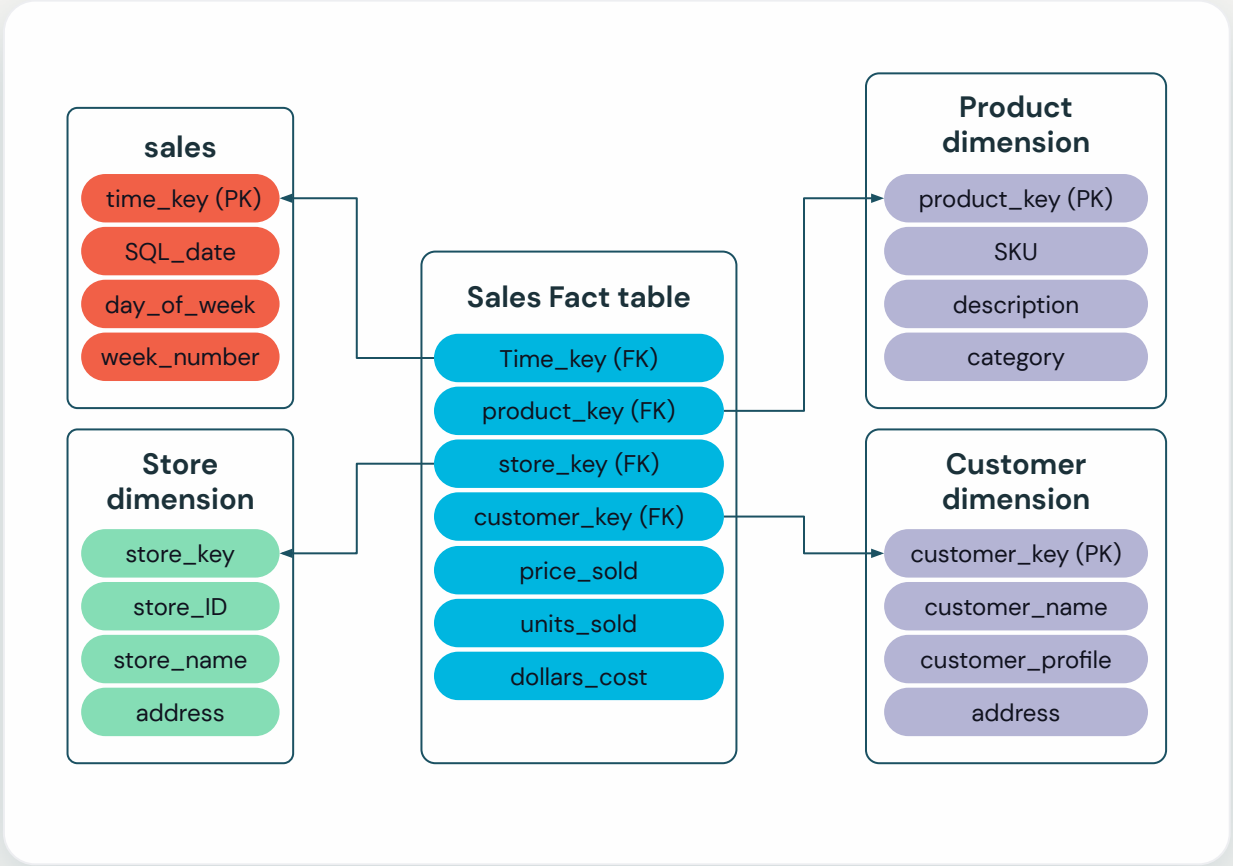
New: Data Modeling With Constraints

Familiar and easier schema modeling on the lakehouse

Primary + Foreign Key Constraints to allow end users to understand relationships between tables.

IDENTITY Columns automatically generate unique integer values when new rows are added.

Enforced CHECK Constraints to never worry about data quality or data correctness issues sneaking up on you.



New: INFORMATION_SCHEMA Support

All your metadata just a query away

SQL Standard information_schema.
Unified access to information about catalogs, schema, tables, locations, columns and their associated permissions.

```
SELECT table_name, column_name
FROM information_schema.columns
 WHERE data_type = 'DOUBLE'
   AND table_schema = 'information_schema'
```

New: Geospatial Support

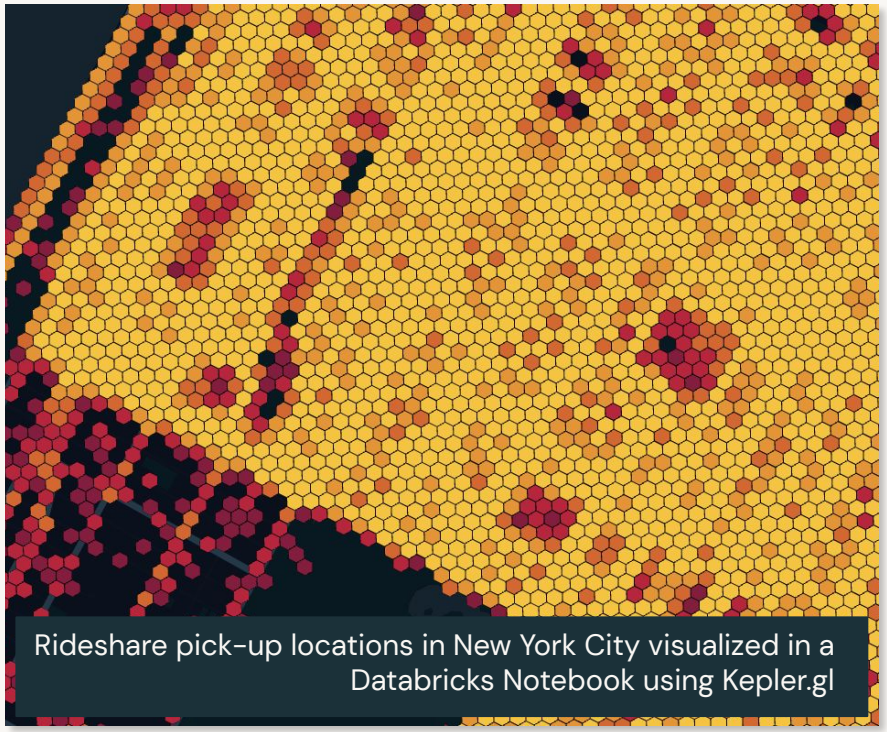
Supercharge your geospatial processing

Support for H3 global indexing

Efficient storage for spatial data in both large and small sizes.

Fast SPATIAL JOINs and binning support.

Easy to Visualize and integrate with ML you don't need to switch betweens tools to maximize geospatial data value.



Rideshare pick-up locations in New York City visualized in a Databricks Notebook using Kepler.gl

New: Query Federation

The lakehouse is home to all data sources

Directly connect to multiple data sources. No matter where your data lives it's just a simple query away.

- Query a remote database without worrying about ingesting
- Combine multiple data sources like PostgreSQL with Delta transparently
- Automatic and intelligent pushdown optimizations.

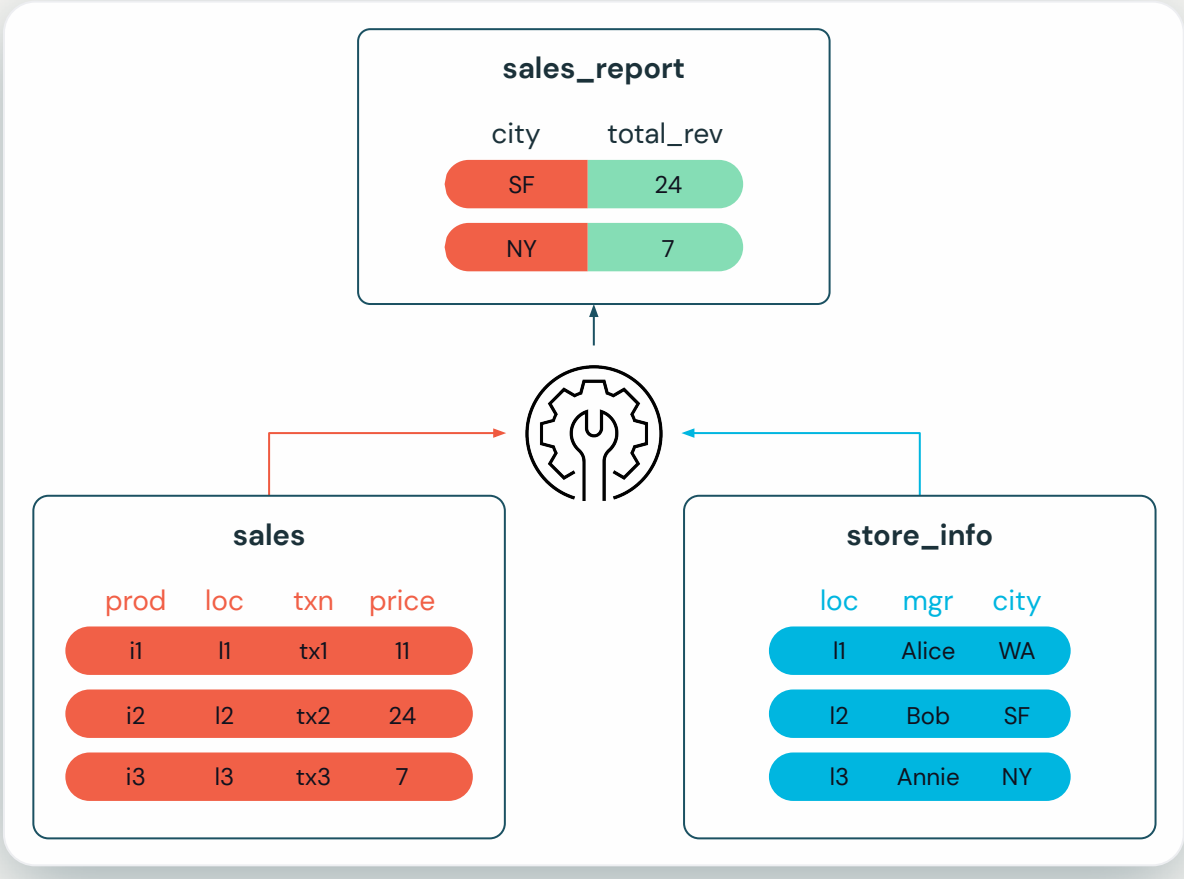
```
CREATE EXTERNAL TABLE
taxi_trips.taxi_transactions
USING postgresql OPTIONS
(
  dbtable 'taxi_trips',
  host secret("postgresdb","jost"),
  port '5432',
  database secret("postgresdb","db"),
  user secret(postgresdb","username"),
  password secret("postgresdb","password")
);
```

New: Materialized Views

Speed up queries with pre-computed results

Accelerate end-user queries and reduce infrastructure costs with efficient, incremental computation

- Accelerate **BI dashboards** and **ETL queries**
- **Streaming**: build MVs on top of live tables
- **Easy ELT**: Simplify reporting by cleaning, enriching, denormalizing the base tables
- **Data Sharing & Access Control**: control what info can be seen by internal and external users and organizations



New: Python User Defined Functions (UDFs)

Run Python UDFs from an isolated execution environment

Integrate **Machine Learning** models, **custom logic** & bring the flexibility of Python right into Databricks SQL!

```
CREATE FUNCTION redact(a STRING)
RETURNS STRING
LANGUAGE PYTHON
AS $$
import json
keys = ["email", "phone"]
obj = json.loads(a)
for k in obj:
   if k in keys:
       obj[k] = "REDACTED"
return json.dumps(obj)
$$;
```

Predictive I/O

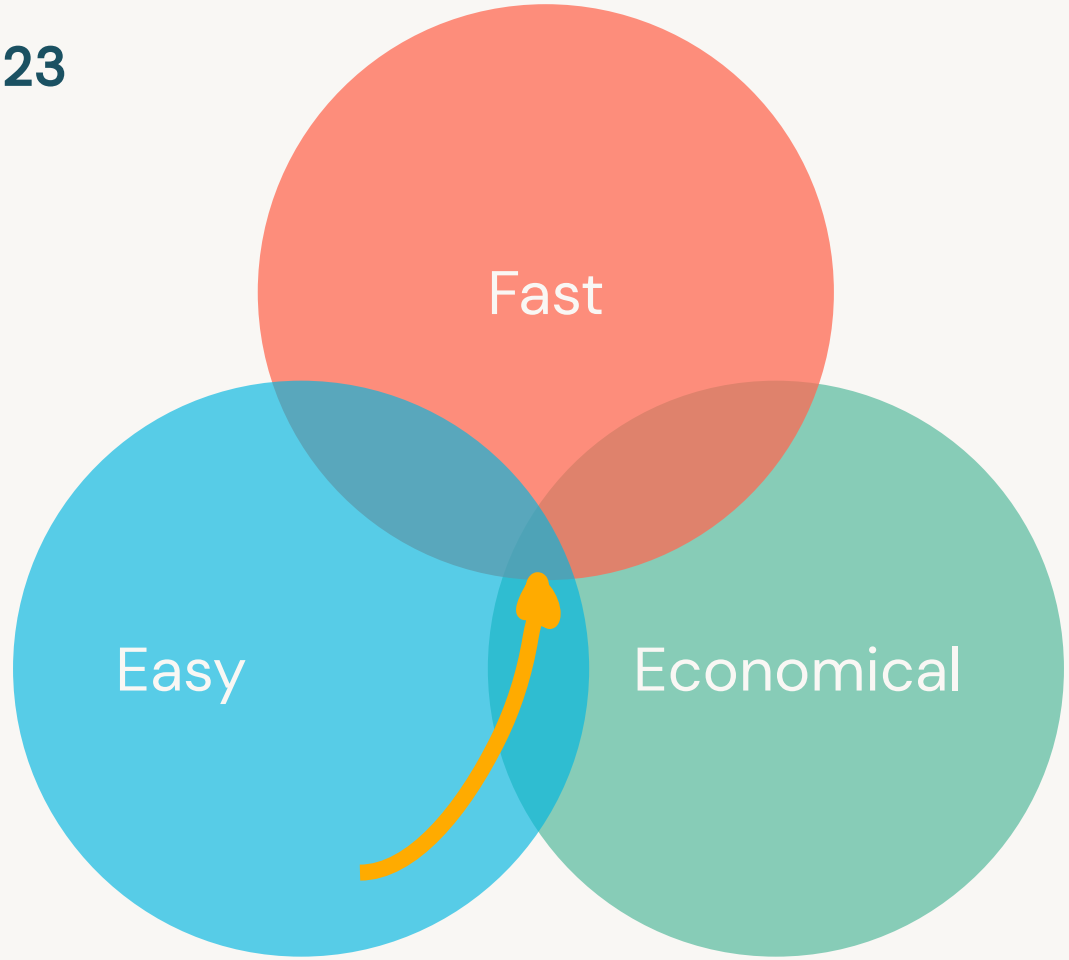
Selective queries without indexes

```
SELECT * from events where user_id = 123
```

What if you didn't have to make a copy of the data or create expensive indexes?

What if the system could learn which data is needed for your queries and anticipate what you'll need next?

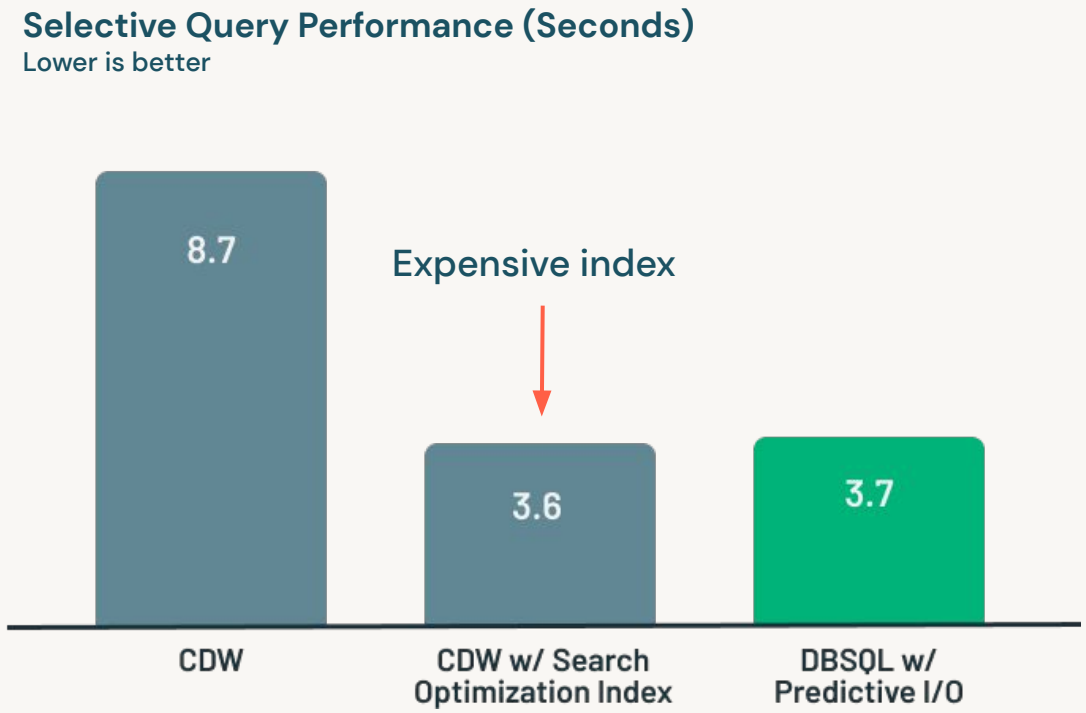
What if your queries were *just* fast without knobs?



Predictive I/O

Using ML to eliminate manual tuning for Performance

- Triangulate where your data is without having to do manual tuning (PARTITION BY, CLUSTER BY)
- Predictive I/O leverages the years of experience Databricks has in building large AI/ML systems to make the lakehouse a smarter data warehouse



Databricks SQL Workshop

DB SQL Exercises

Complete the following

1. Create a new SQL Warehouse (Size S, use other default settings)
2. Create a new query showing the top 3 energy producing states. This will be based on the hourly dataset current_turbine_metrics. Name this query "Top 3 Energy Producing States". Attach the SQL Warehouse you created in step 1.
3. Run the query and look at the query plan.
4. Create an alert on this query if the minimum value is < 200.
5. Create a visualization based on these results.
6. Create a new Dashboard and add this visualization to the dashboard.

DB SQL Exercises

Answers

1. Create a new SQL Warehouse (Size S, use other default settings)
2.
```
select sum(avg_energy),state from dbdemos_lakehouse_iot.current_turbine_metrics group by state order by sum(avg_energy) desc limit 3;
```
3. Run the query and look at the query plan.
4. Create an alert on this query if the minimum value is < 200.
5. Create a visualization based on these results.
6. Create a new Dashboard and add this visualization to the dashboard.

Machine Learning

Hardest Part of ML isn't ML, it's Data

"Hidden Technical Debt in Machine Learning Systems," *Google NIPS 2015*

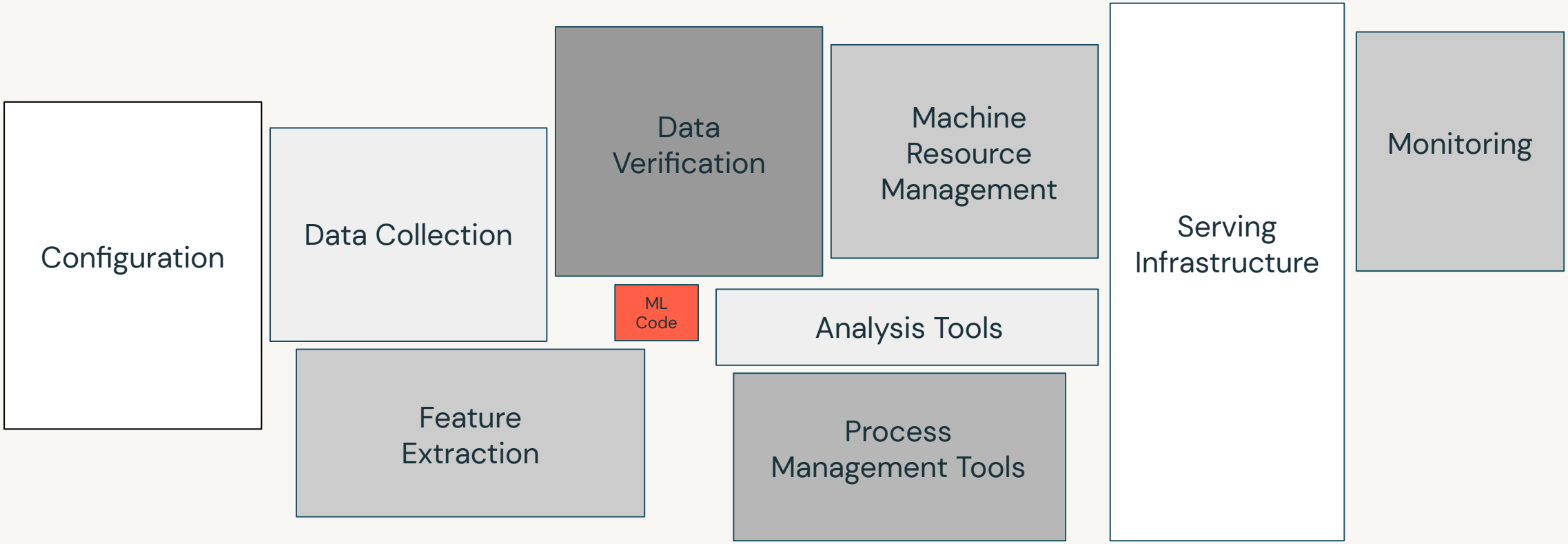


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small red box in the middle. The required surrounding infrastructure is vast and complex.

ML & data science workloads on Databricks

Machine Learning

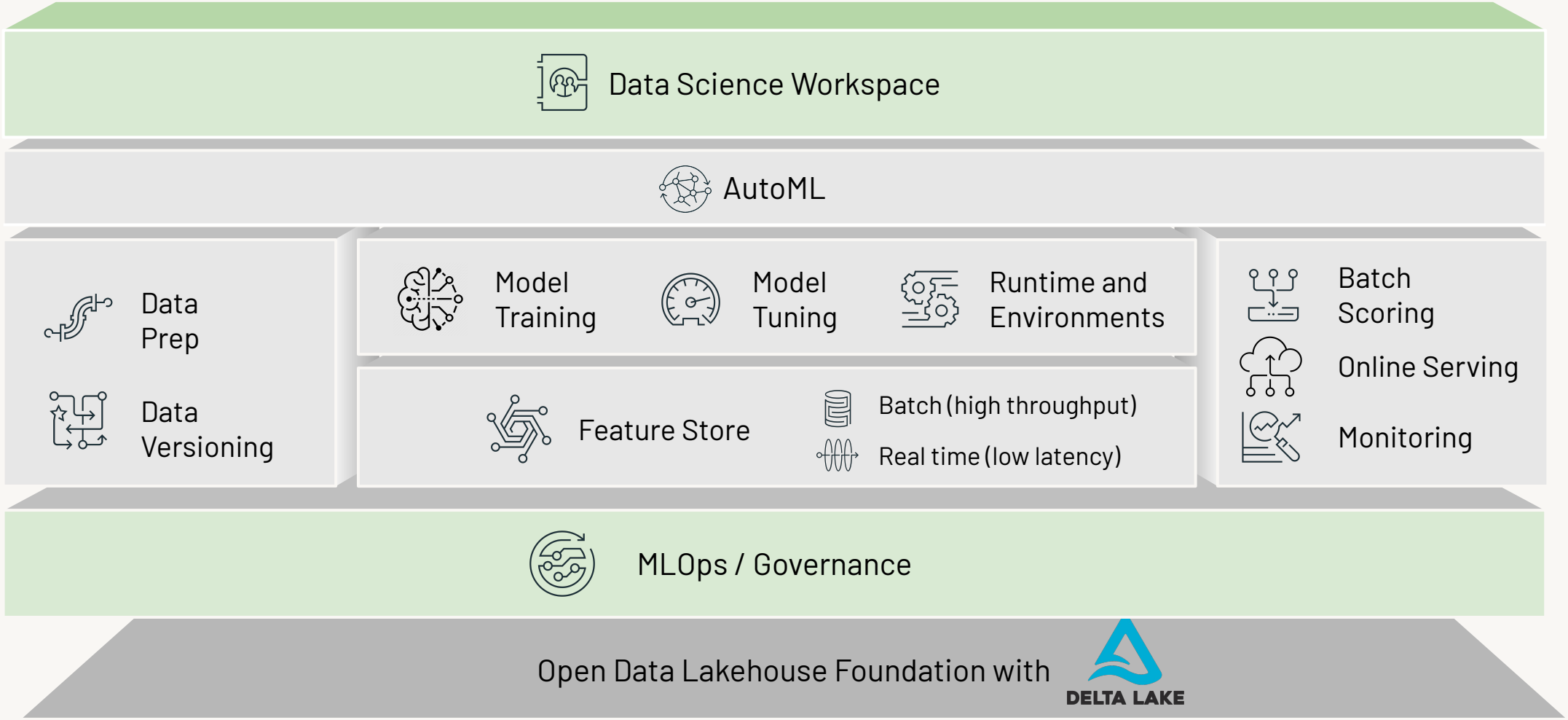
- Model registry, reproducibility, productionization
- Leverages Delta Lake for reproducibility
- AutoML for citizen data scientists

Data Science

- Collaborative notebooks and dashboards for interactive analysis
- Native support for Python, Java, R, Scala
- Delta Lake data natively supported

Databricks Machine Learning

A data-native and collaborative solution for the full ML lifecycle



Full ML Lifecycle: MLOps for Data Teams

MLOps = DataOps + DevOps + ModelOps

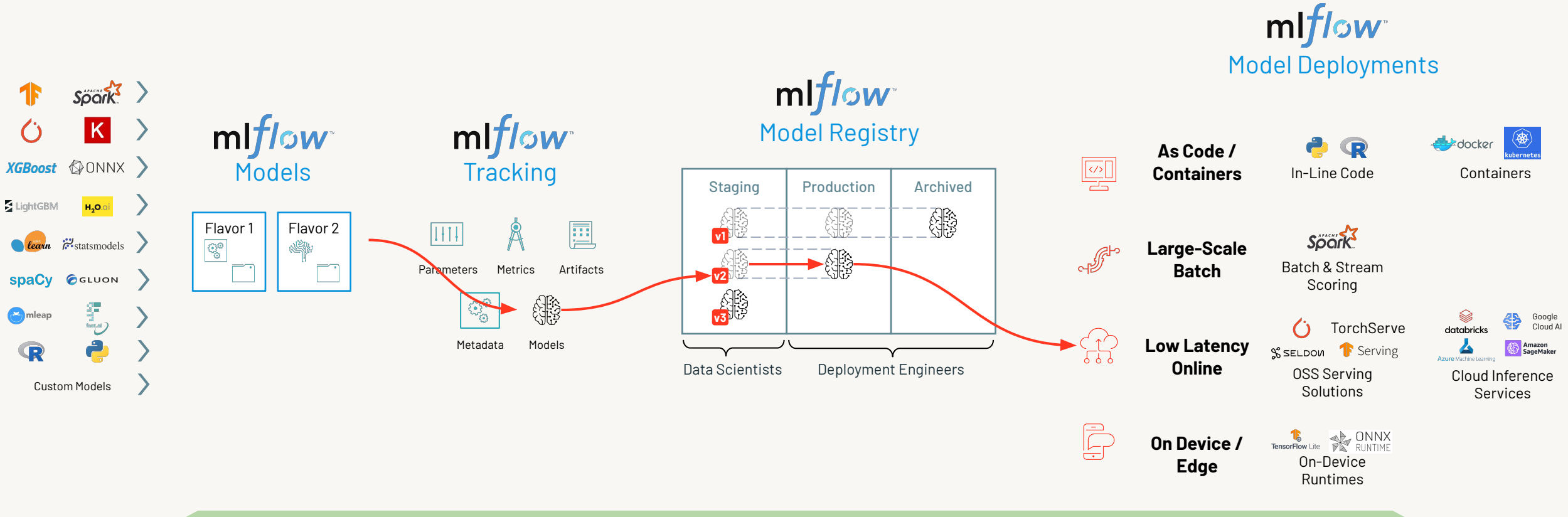
Data Versioning with Time Travel

Code Versioning with Git Integration

Model Lifecycle Management with Model Registry

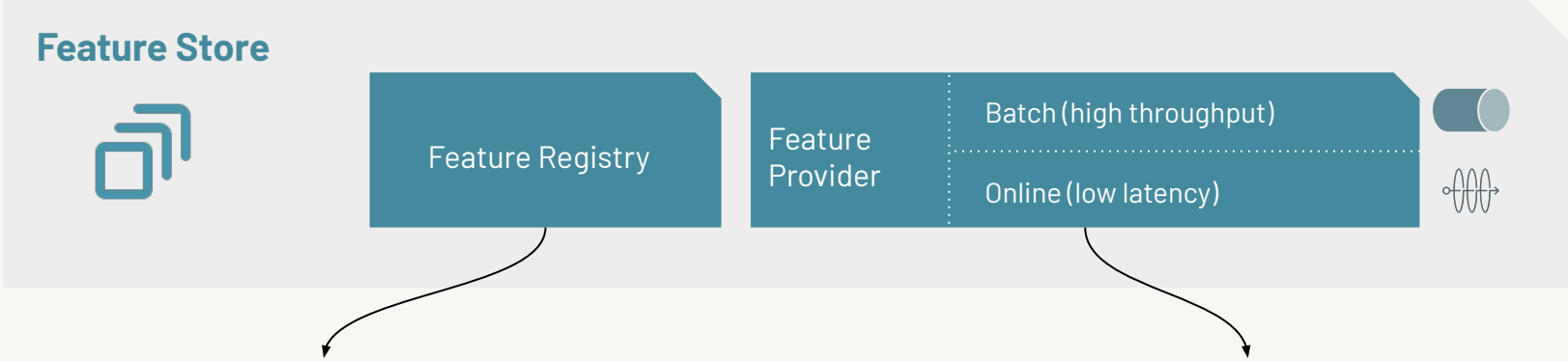
MLOps / Governance

Full ML Lifecycle: MLOps for Data Teams



Feature Store

The first Feature Store codesigned with a Data and MLOps Platform



Feature Registry

- Discoverability and Reusability
- Versioning
- Upstream and downstream Lineage

Feature Provider

- Batch and online access to Features
- Feature lookup packaged with Models
- Simplified deployment process

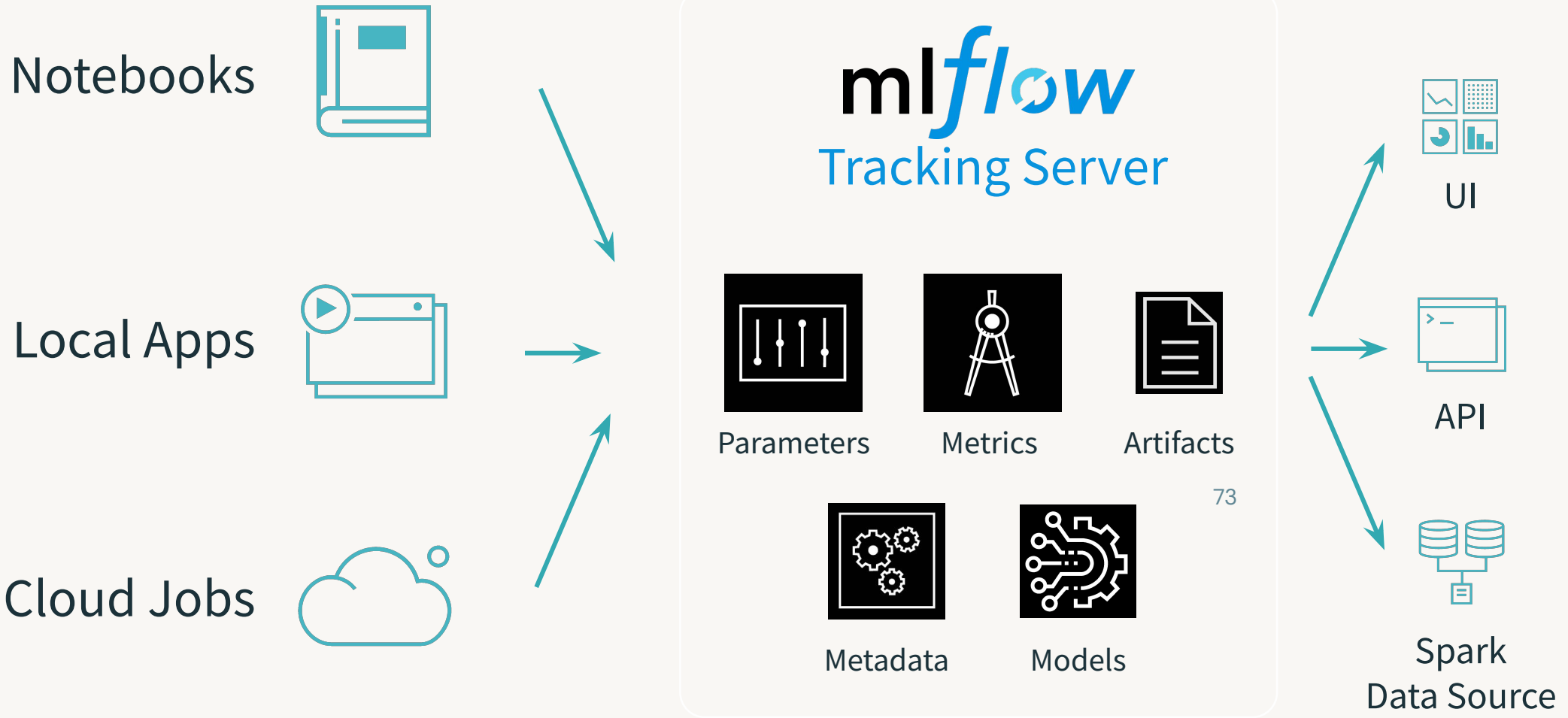
Co-designed with DELTA LAKE

- Open format
- Built-in data versioning and governance
- Native access through PySpark, SQL, etc.

Co-designed with mlflow

- Open model format that supports all ML frameworks
- Feature version and lookup logic hermetically logged with Model

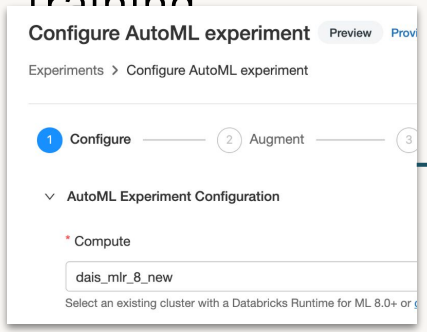
mlflow Tracking

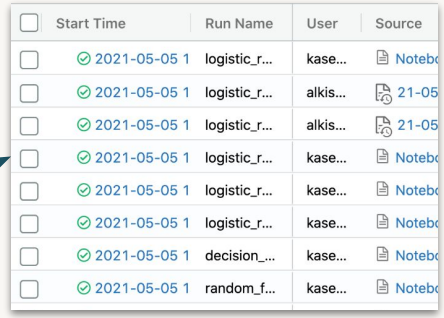


Databricks AutoML

A glass-box solution that empowers data teams without taking away control

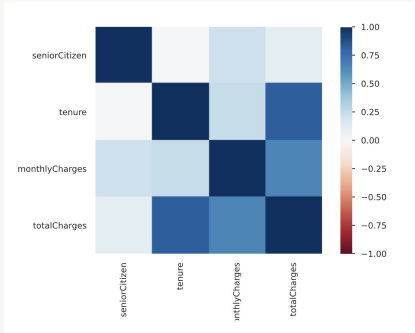
UI and API to start AutoML training





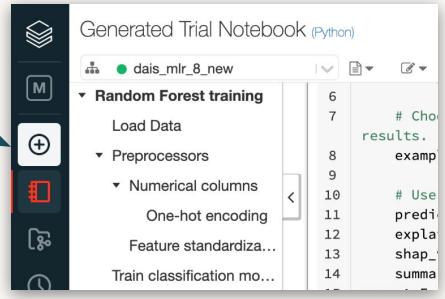
MLflow experiment
Auto-created MLflow Experiment to track models and metrics

→ Easily deploy to Model Registry



Data exploration notebook
Generated notebook with feature summary statistics and distributions

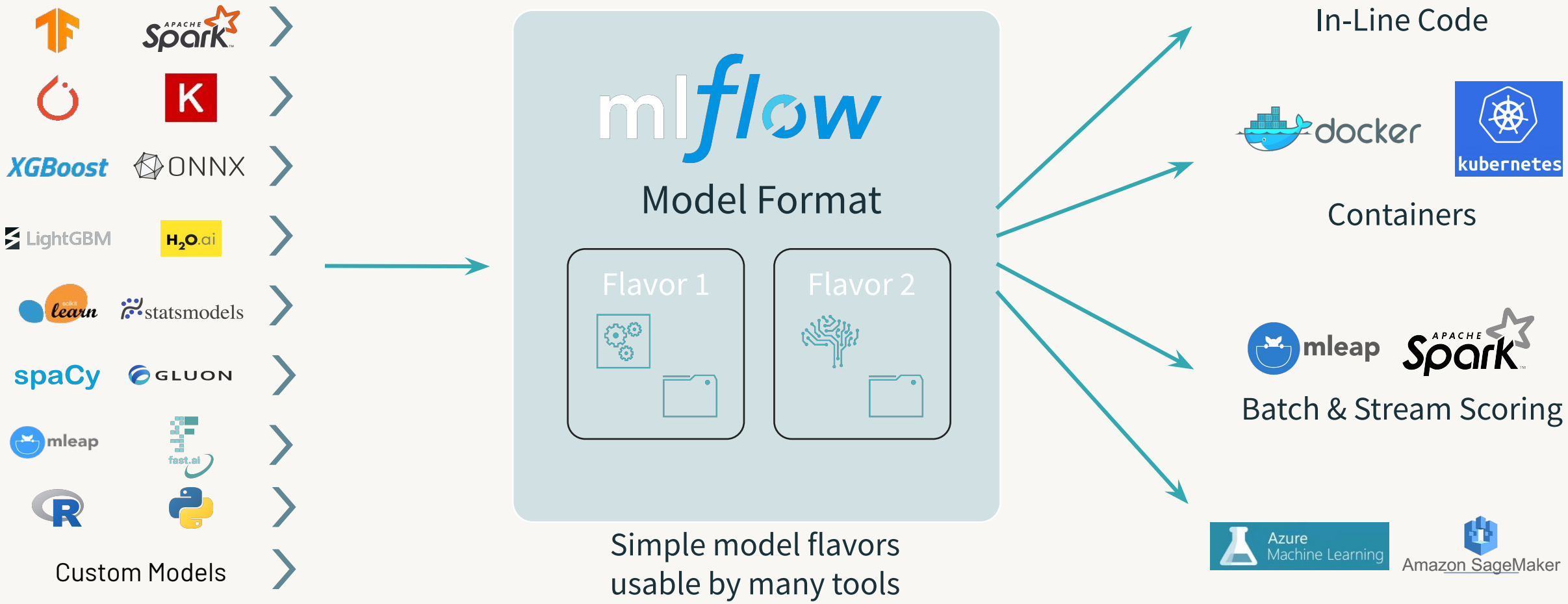
→ Understand and debug data quality and preprocessing



Reproducible trial notebooks
Generated notebooks with source code for every model

→ Iterate further on models from AutoML, adding your expertise

mlflow Models

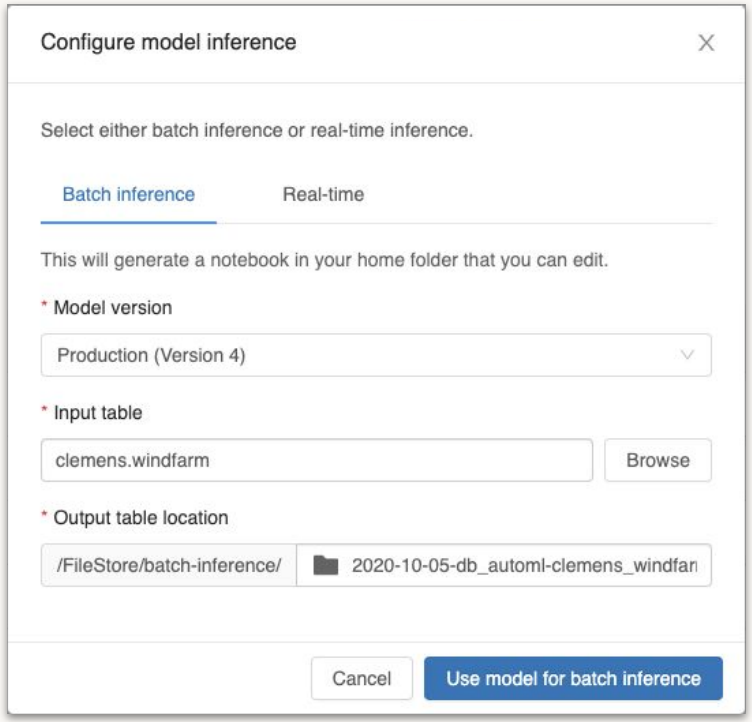


Model Deployment / Serving Options

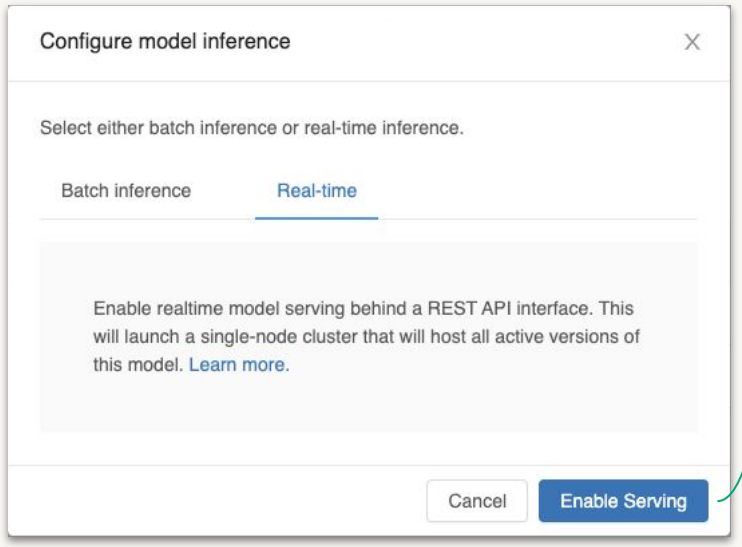
Deploy any ML model at large scale AND low latency

Use model for inference — One click model deployment directly from the Model Registry

Large-scale batch scoring



Low-latency online serving



Classic - Available in All Clouds

Serverless - Private Preview in AWS

New Serverless Model Endpoints

- Low latency REST endpoints
- High availability SLAs
- Serverless and autoscaling
- Built-in observability and connectors

Machine Learning Demo