databricks

Delta Live Tables

Automatic reliable ETL on Delta Lake

Chris Hoshino-Fish

Lead Solutions Architect, Databricks since 2017

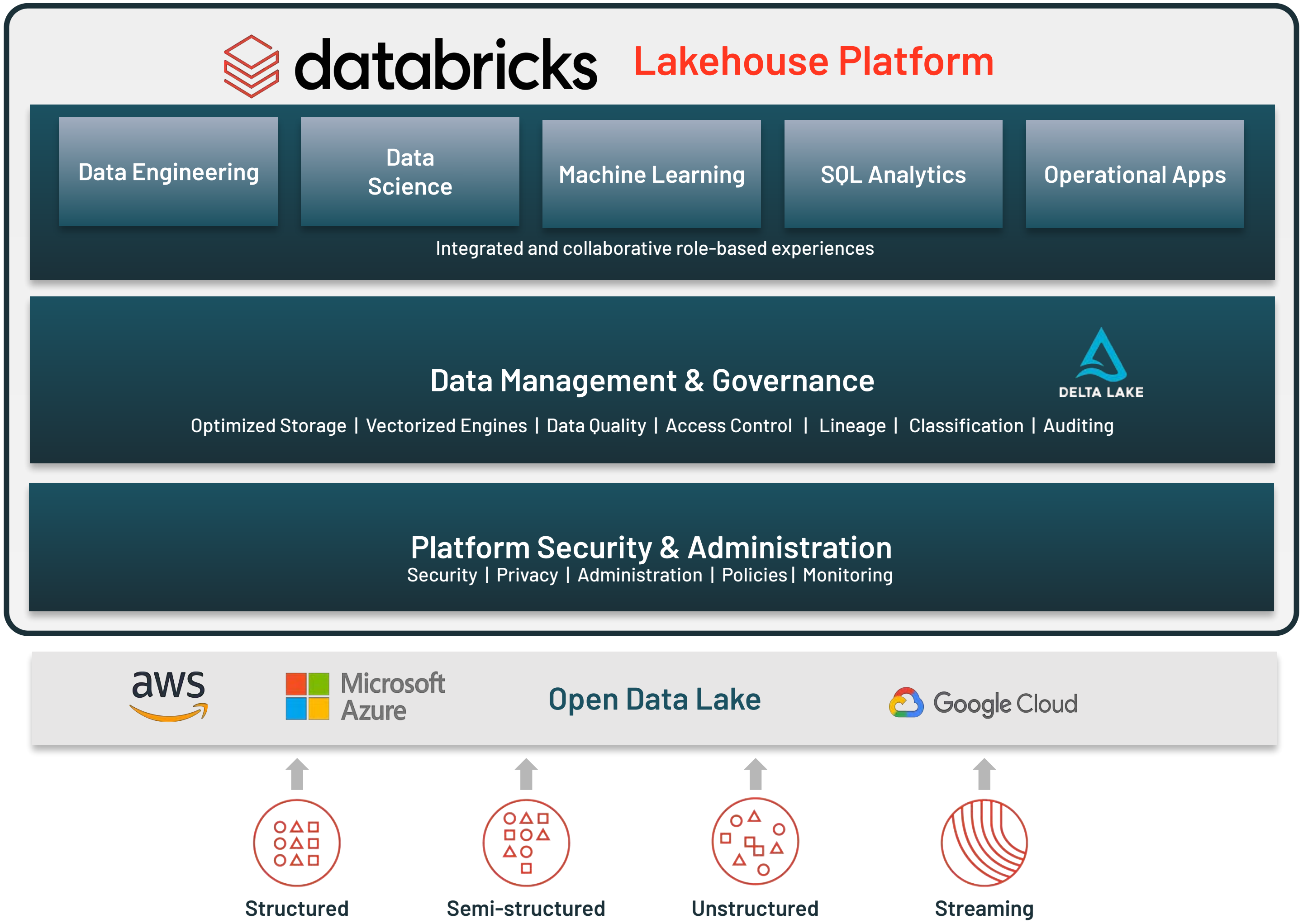
Specialize in Real-Time Data systems & Performance Engineering

Data Engineer since 2014

B.A. Computational Mathematics, UC Santa Cruz 2012

fish@databricks.com

UNIVERSITY OF CALIFORNIA SANTA CRUZ



databricks Lakehouse Platform

Data Engineering

Data Science

Machine Learning

SQL Analytics

Operational Apps

Integrated and collaborative role-based experiences

Data Management & Governance

DELTA LAKE

Optimized Storage | Vectorized Engines | Data Quality | Access Control | Lineage | Classification | Auditing

Platform Security & Administration

Security | Privacy | Administration | Policies | Monitoring

aws

Microsoft Azure

Open Data Lake

Google Cloud

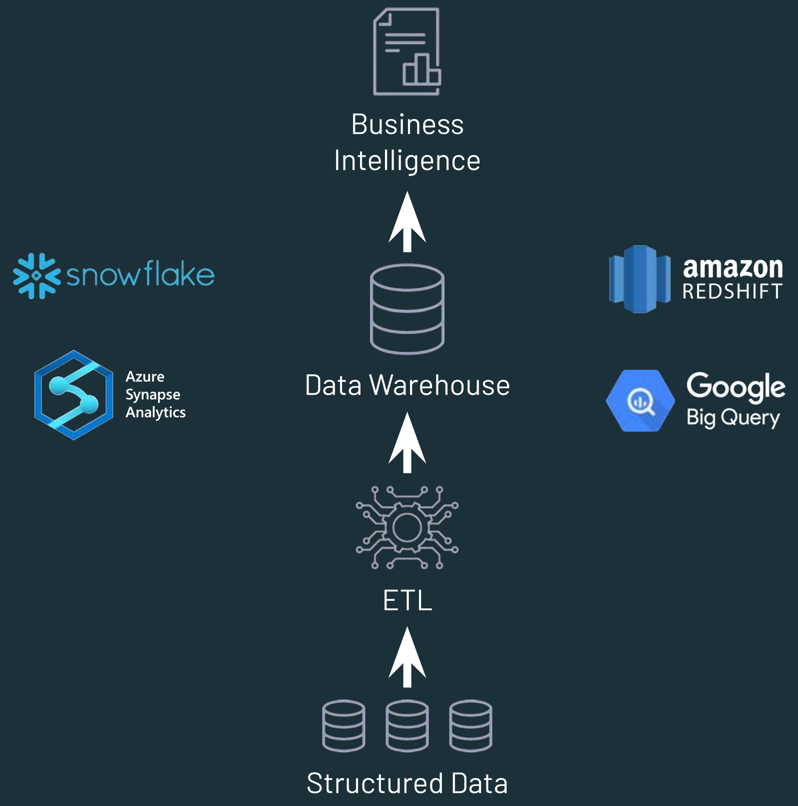
Structured

Semi-structured

Unstructured

Streaming

Data Warehouses



Business Intelligence

Data Warehouse

ETL

Structured Data

Pros

- Great for Business Intelligence (BI) applications

Cons

- Limited support for Machine Learning (ML) workloads
- Proprietary systems with only a SQL interface

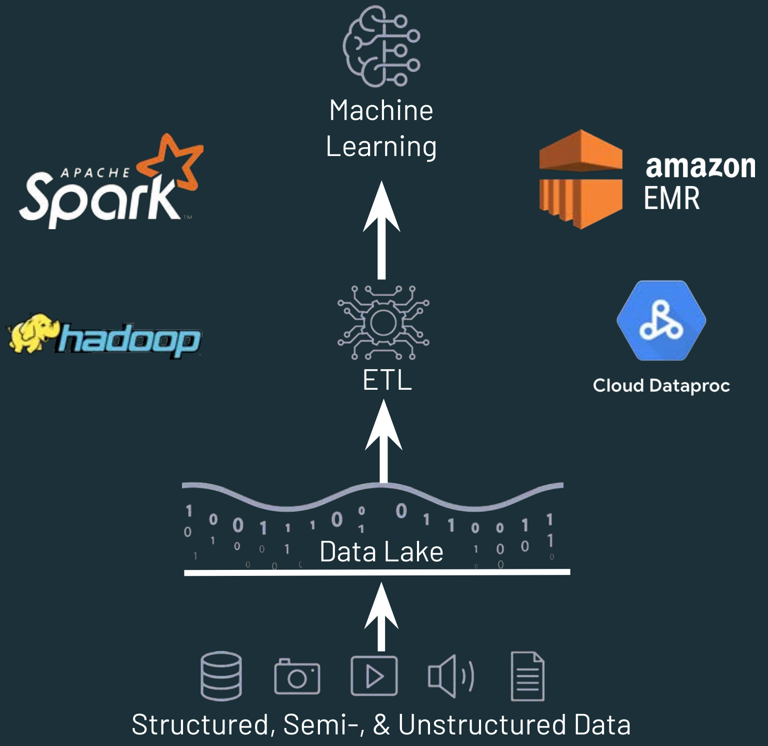
Data Lakes

Pros

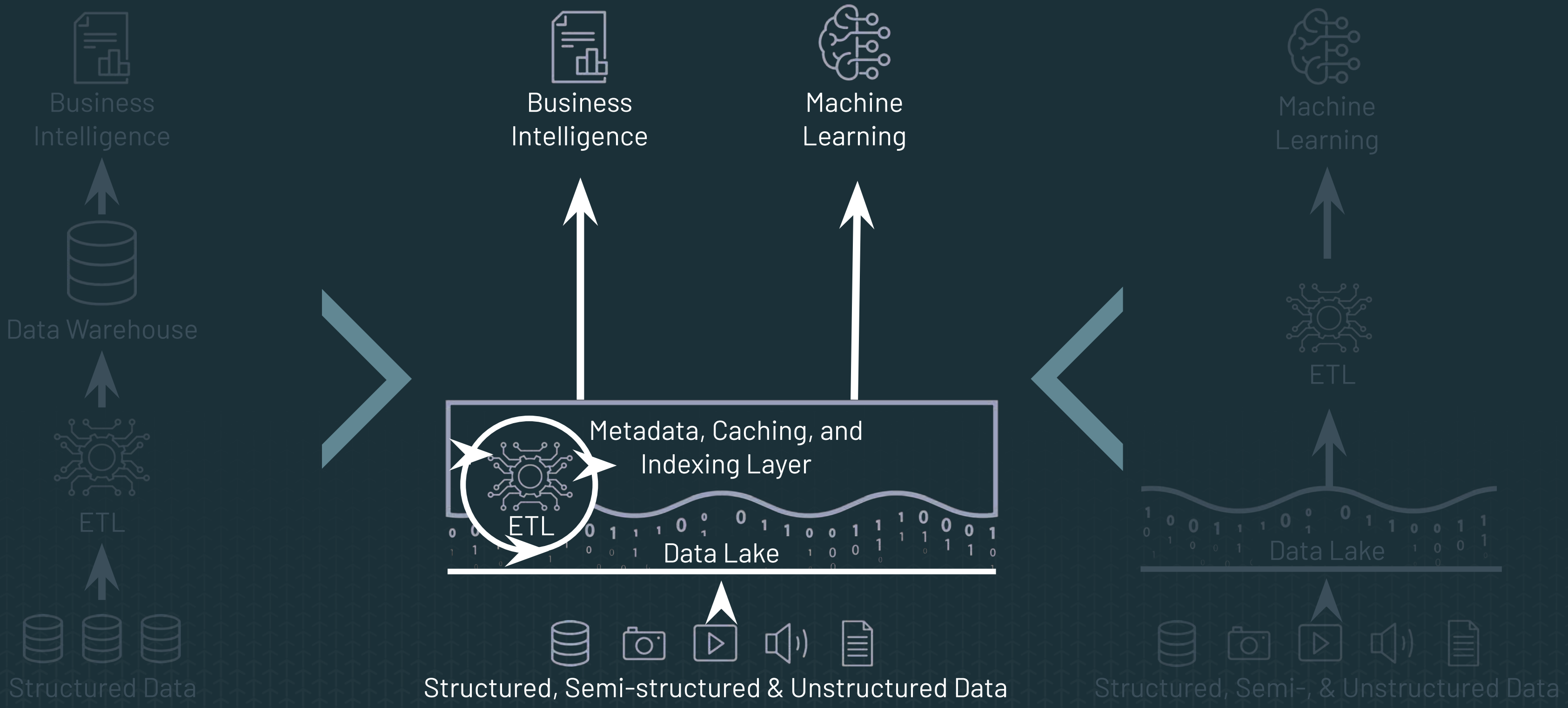
- Supports ML
- Open formats and big ecosystem

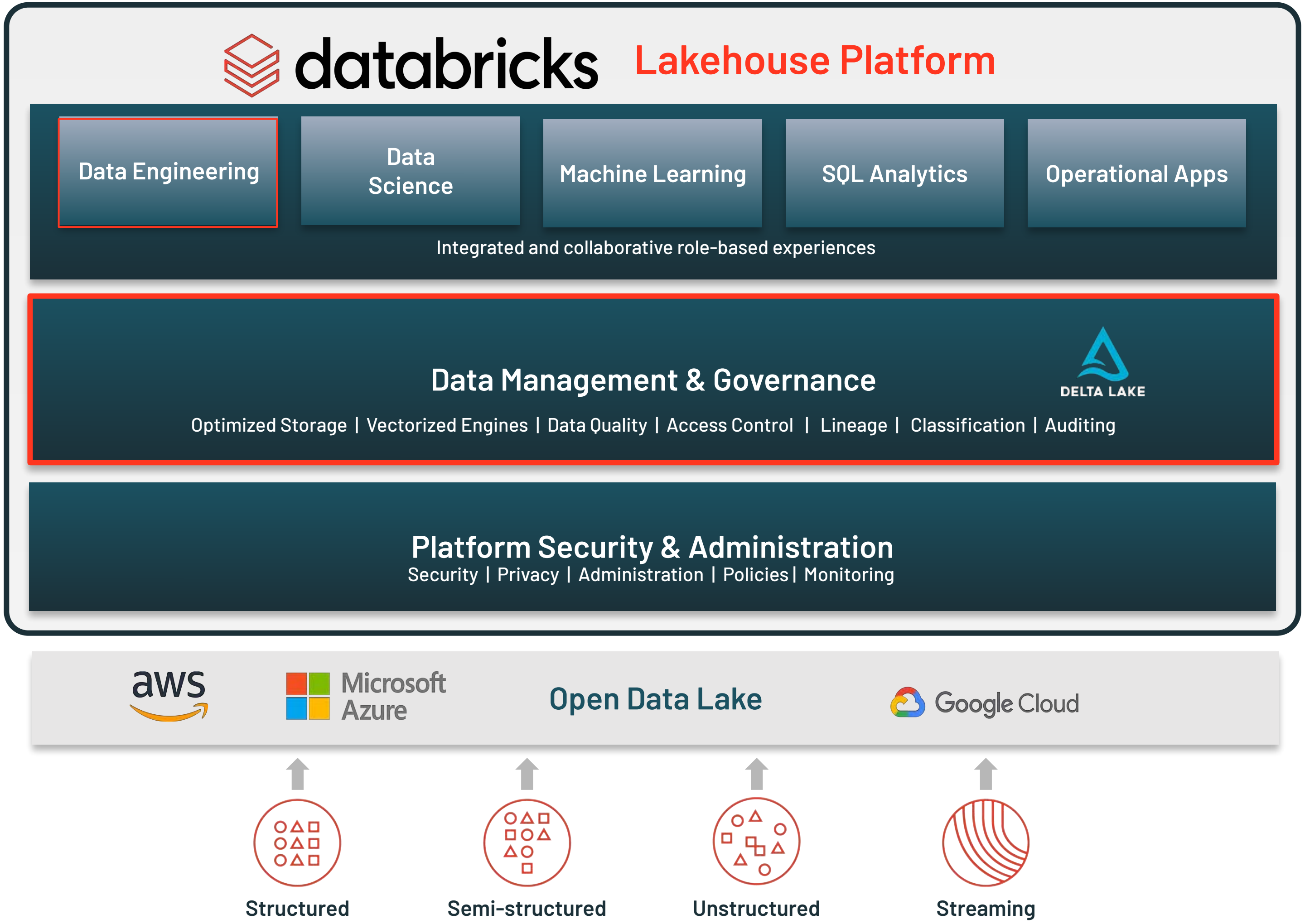
Cons

- Poor support for BI
- Complex data quality problems



New Way Forward: Lakehouse





databricks **Lakehouse Platform**

Data Engineering

Data Science

Machine Learning

SQL Analytics

Operational Apps

Integrated and collaborative role-based experiences

Data Management & Governance

DELTA LAKE

Optimized Storage | Vectorized Engines | Data Quality | Access Control | Lineage | Classification | Auditing

Platform Security & Administration

Security | Privacy | Administration | Policies | Monitoring

aws

Microsoft Azure

Open Data Lake

Google Cloud

Structured

Semi-structured

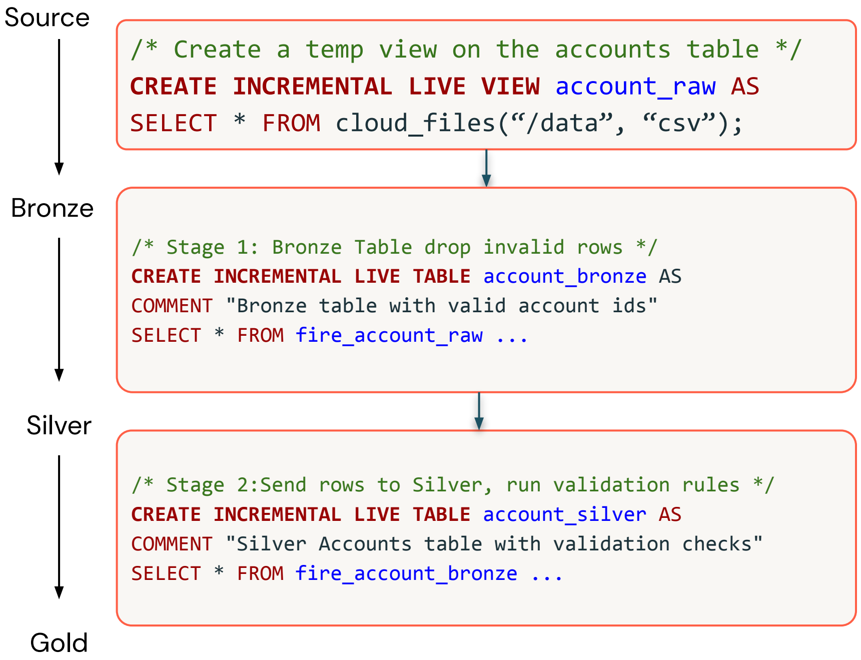
Unstructured

Streaming

Key differentiators for successful data engineering

- Continuous or scheduled data ingestion
- Declarative ETL pipelines
- Change Data Capture
- Data quality validation and monitoring
- Data pipeline observability
- Automated scaling and fault tolerance
- Automatic deployments and operations
- Orchestrate pipelines & workflows

Declarative ETL pipelines with Delta Live Tables



Source

```
/* Create a temp view on the accounts table */
CREATE INCREMENTAL LIVE VIEW account_raw AS
SELECT * FROM cloud_files(“/data”, “csv”);
```

Bronze

```
/* Stage 1: Bronze Table drop invalid rows */
CREATE INCREMENTAL LIVE TABLE account_bronze AS
COMMENT "Bronze table with valid account ids"
SELECT * FROM fire_account_raw ...
```

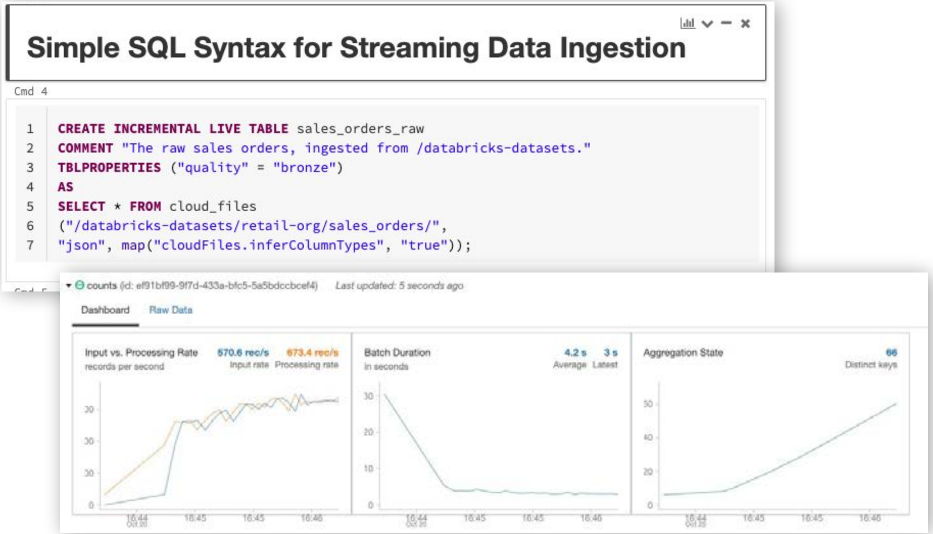
Silver

```
/* Stage 2:Send rows to Silver, run validation rules */
CREATE INCREMENTAL LIVE TABLE account_silver AS
COMMENT "Silver Accounts table with validation checks"
SELECT * FROM fire_account_bronze ...
```

Gold

- Use intent-driven declarative development to abstract away the **“how”** and define **“what”** to solve
- Automatically create high-quality lineage and manage table dependencies across the data pipeline
- Automatically checks for errors, missing dependencies and syntax errors, and manage pipeline recovery

Continuous or scheduled data ingestion with Auto Loader



Simple SQL Syntax for Streaming Data Ingestion

```sql
CREATE INCREMENTAL LIVE TABLE sales_orders_raw
COMMENT "The raw sales orders, ingested from /databricks-datasets."
TBLPROPERTIES ("quality" = "bronze")
AS
SELECT * FROM cloud_files
("/databricks-datasets/retail-org/sales_orders/",
"json", map("cloudFiles.inferColumnTypes", "true"));
```

- **Incrementally and efficiently** process new data files as they arrive in cloud storage
- 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

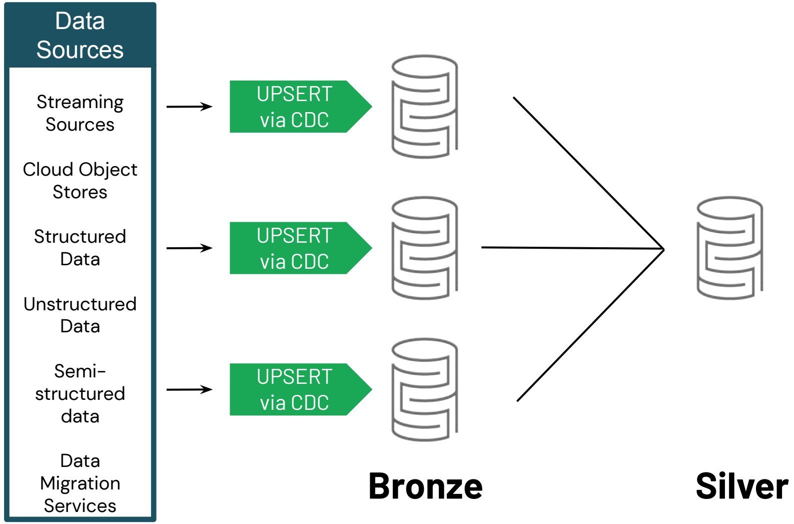
✅ JSON

✅ CSV

Coming Soon
AVRO

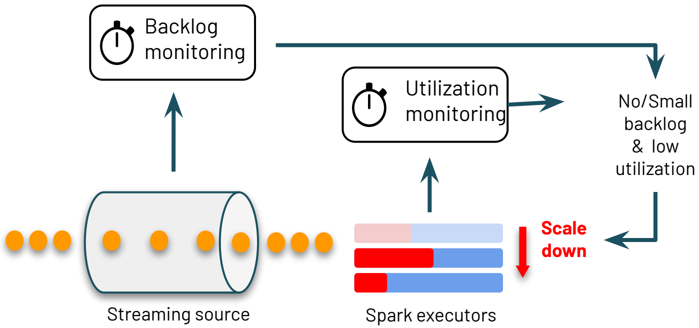
Coming Soon
PARQUET

Change data capture (CDC) with Delta Live Tables



- Capture row-level changes from any data source supported by DBR, cloud storage, or DBFS
- Simpler architecture: build, simple incremental pipelines
- Handles out-of-order events
- Schema evolution
- Process change records (inserts, updates, deletes) incrementally using a simple, declarative "APPLY CHANGES INTO" SQL API

Automated scaling and fault tolerance with Delta Live Tables



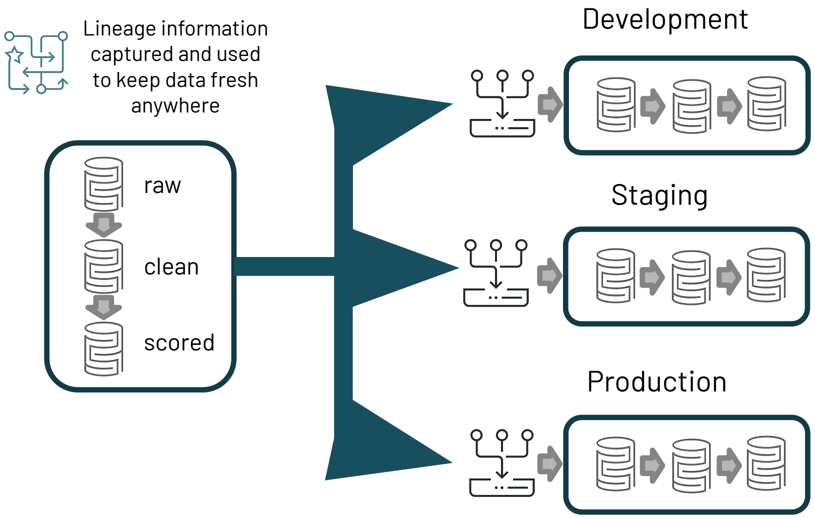
- Meet streaming SLOs with backlog-aware scaling decisions - Monitor both, **backlog metrics and cluster utilization** to scale up or down
- **Reduce down time** with automatic error handling and easy replay
- **Eliminate maintenance** with automatic optimizations of all Delta Live Tables
- Execute data pipeline workload on **automatically provisioned** elastic Apache Spark™-based compute clusters that parallelize jobs as well as minimize data movement

Automatic deployments and operations with Delta Lives Table

- Complete, parameterized and automated deployment for the continuous data delivery

- **Reuse ETL pipelines** across environments with config files and parameterization

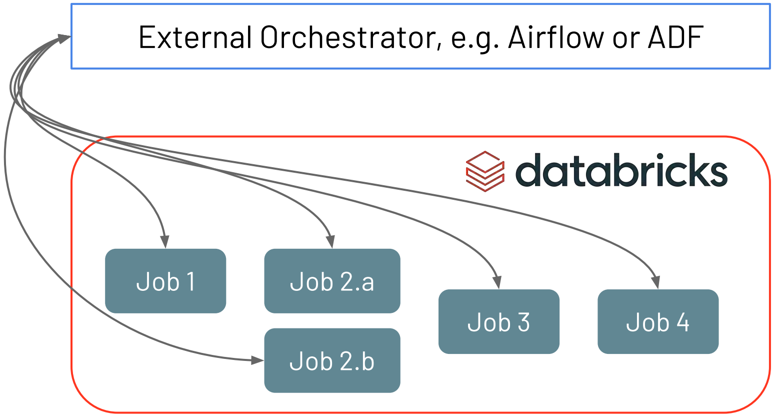
- Orchestrates, tests, and monitor end-to-end the data pipeline


Lineage information captured and used to keep data fresh anywhere
raw
clean
scored
Development
Staging
Production

Workflow Management on Databricks

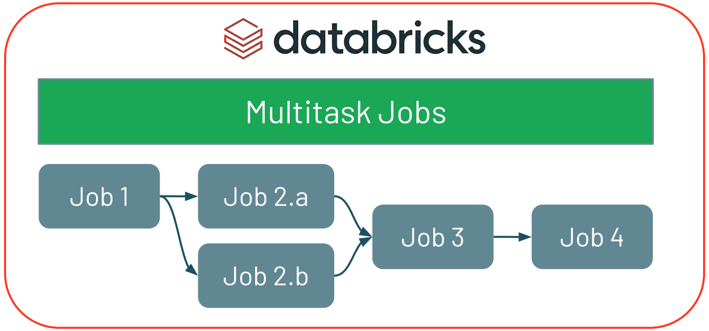
Simplify orchestration and management of multi-step workflows

Before



- Cost/complexity of maintaining external orchestrator
- Hard to monitor/debug

After



- Turnkey orchestration within Databricks
- Visibility into job dependencies, debugging, etc.
- Airflow and ADF integrations will continue to be supported

Demo

Additional Resources

- Getting Started with Delta Live Tables
- 5 Steps to Implementing Intelligent Data Pipelines With Delta Live Tables
- Product Page
- Documentation
- Spark’s Structured Streaming
- Delta Lake
- Great Expectations