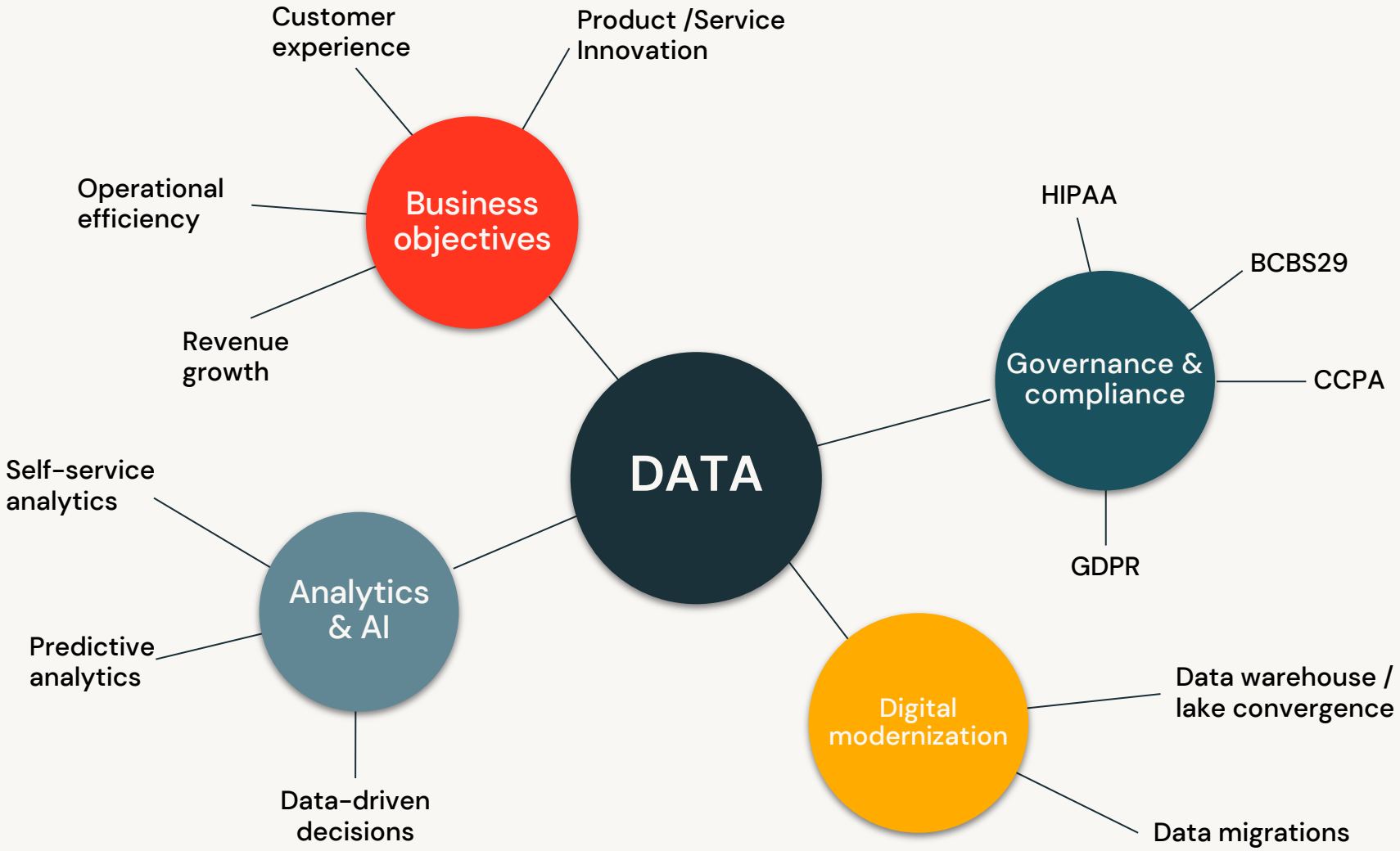
databricks

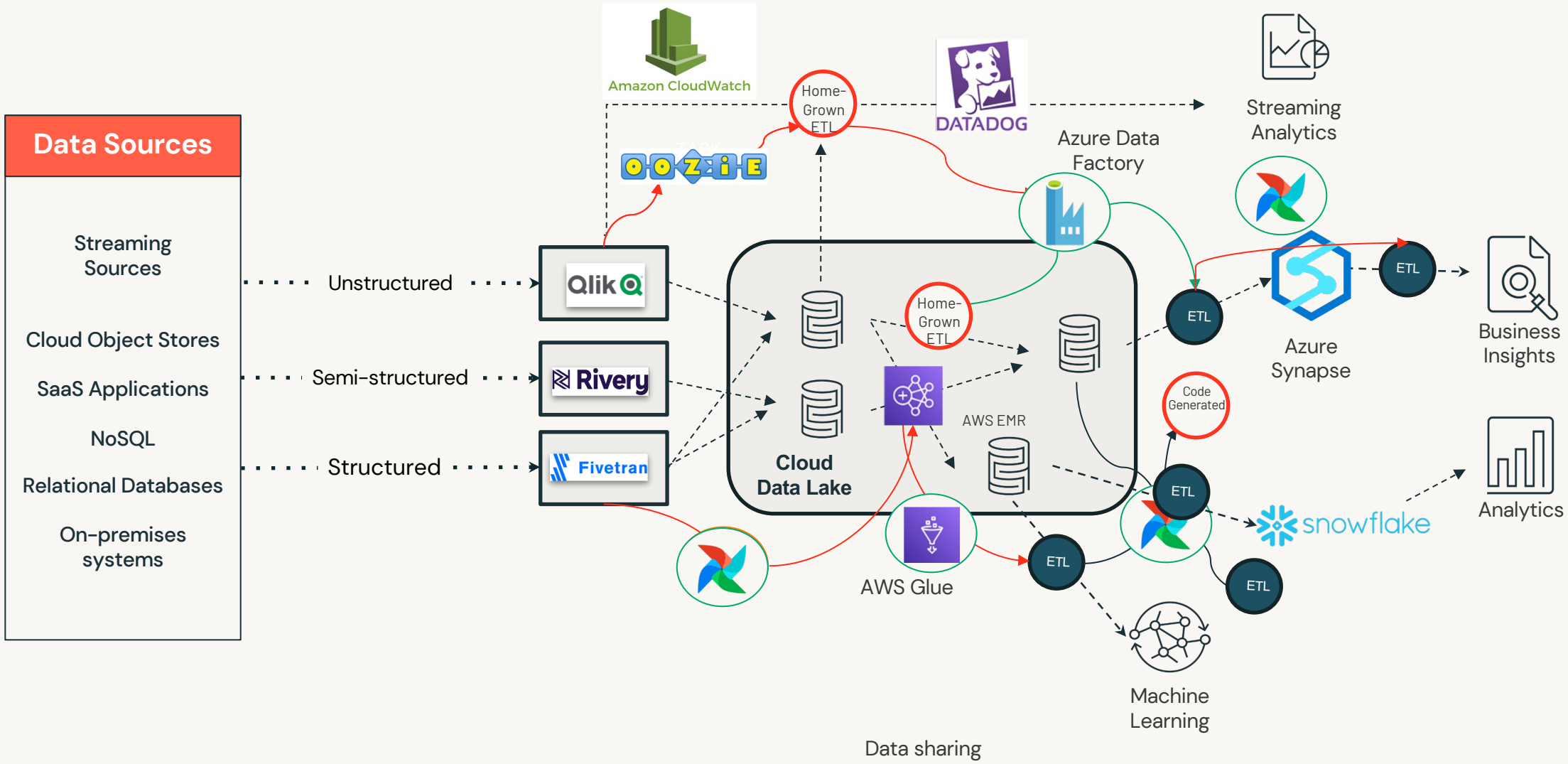
ETL with Databricks Delta Live Tables

What's the problem with Data Engineering?

We know data is critical to business outcomes

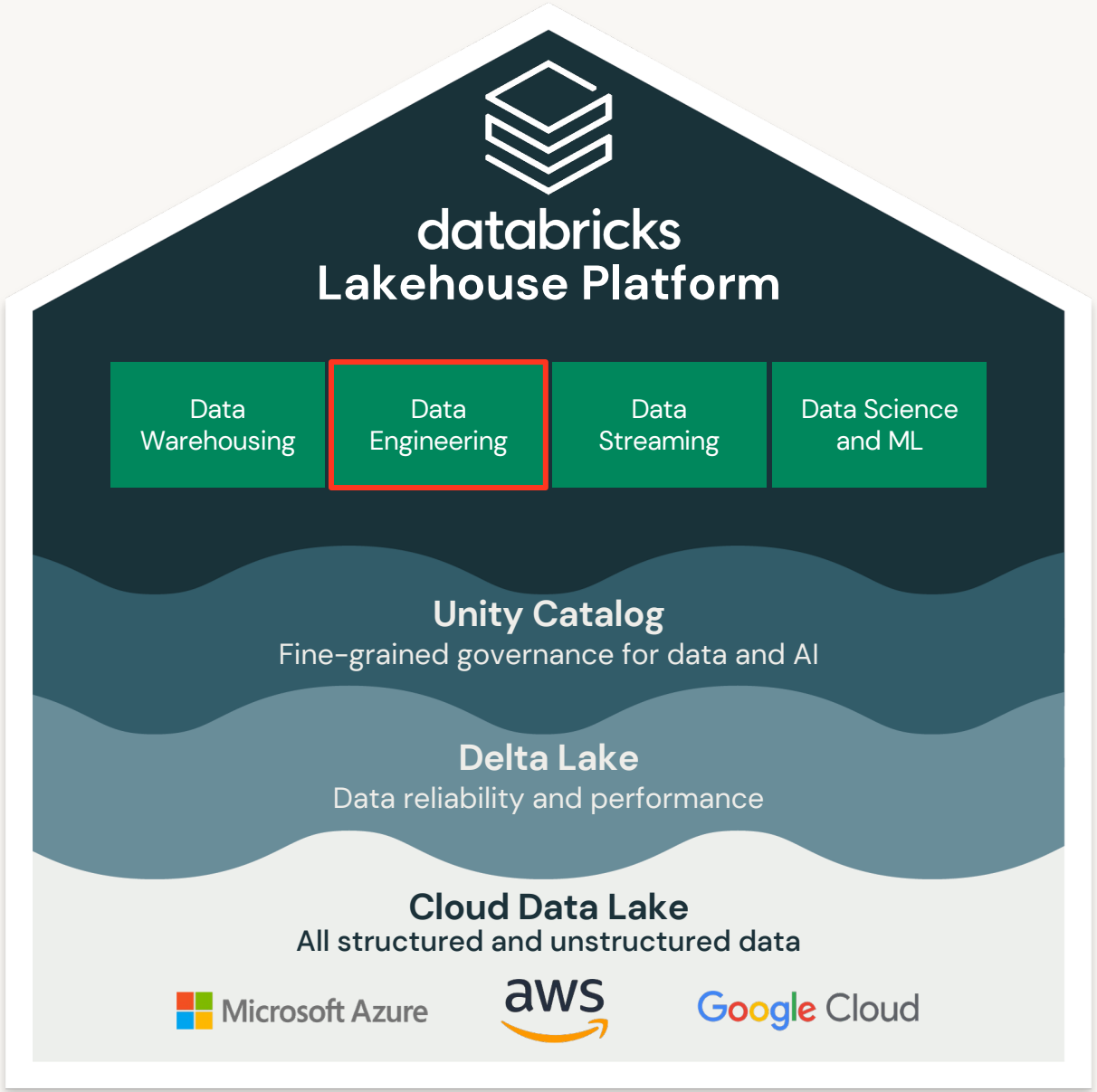


But there is complexity in data engineering....



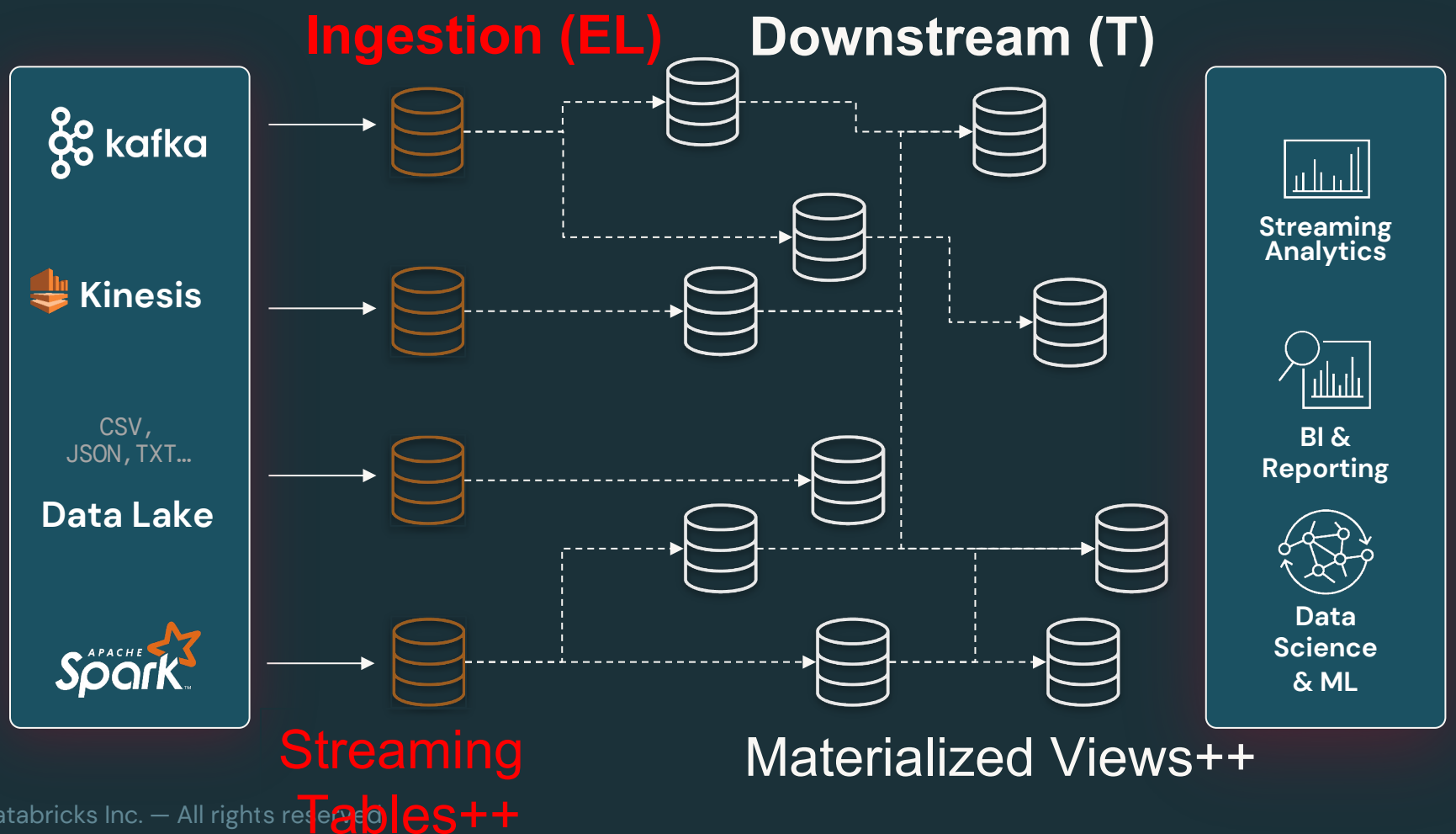
How does Databricks Help?

Databricks Lakehouse Platform is the foundation for **Data Engineering**



Declarative Pipelines for ExtractLoad-Transform

-> script defining a DAG of ingestion and downstream transform



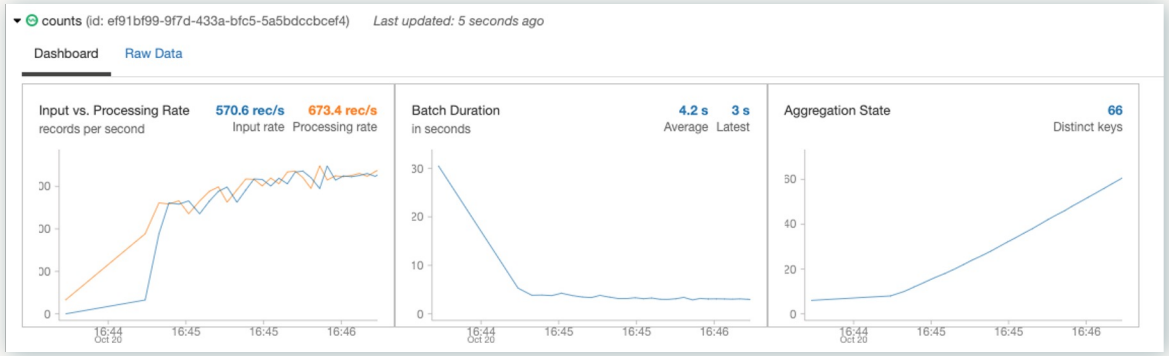
Key Differentiators:
Databases meet Software Engineering and Systems

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
```

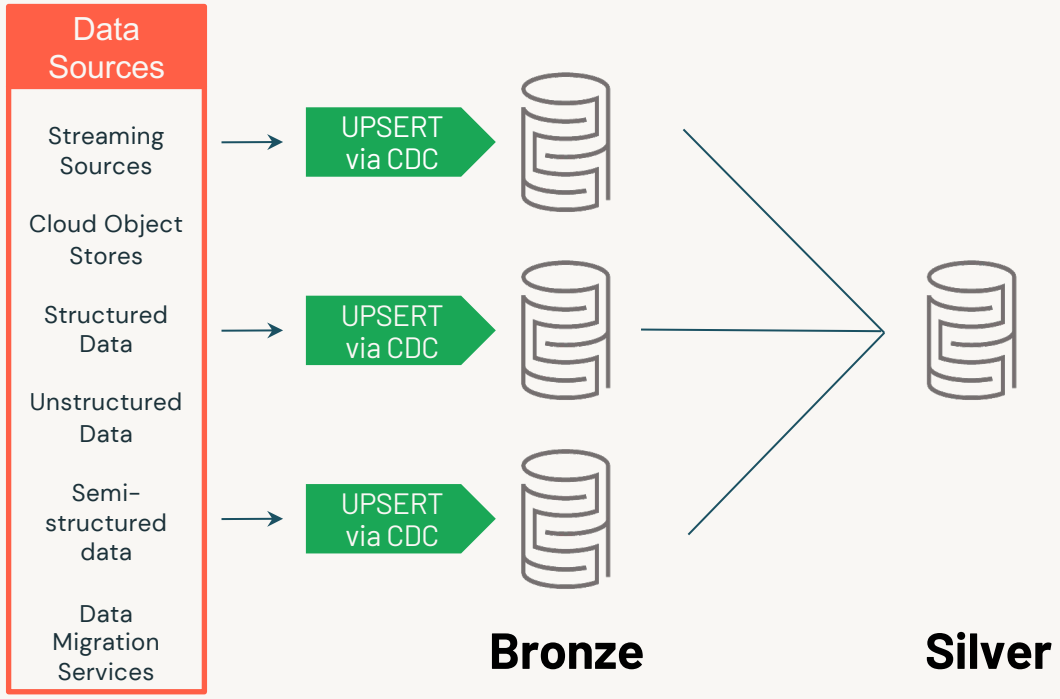


- Incrementally and efficiently process new data files as they arrive in cloud storage using Auto Loader
- Automatically infer schema of incoming files or superimpose what you know with Schema Hints
- Automatic schema evolution
- Rescue data column – never lose data again

Schema Evolution

✅ JSON ✅ CSV ✅ AVRO ✅ PARQUET

Change data capture (CDC)



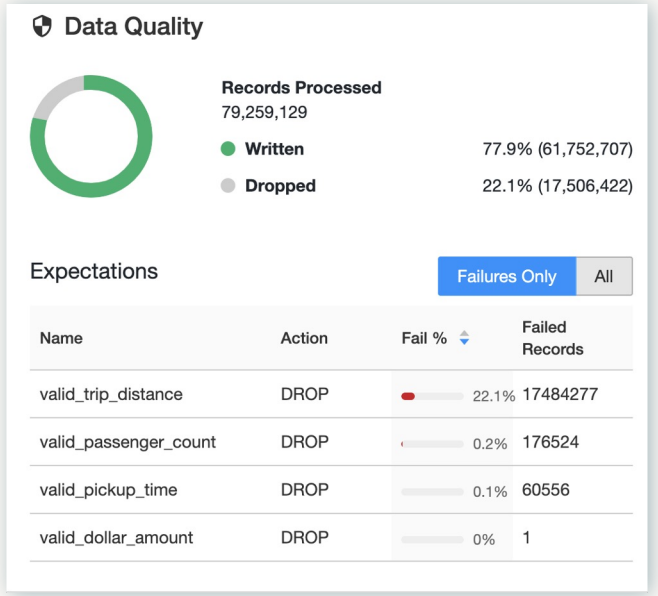
- Stream change records (inserts, updates, deletes) from any data source supported by DBR, cloud storage, or DBFS
- Simple, declarative "APPLY CHANGES INTO" API for SQL or Python
- Handles out-of-order events
- Partial Updates
- SCD2 support

Data Engineers don’t just code: collaborate, version, test, validate, monitor

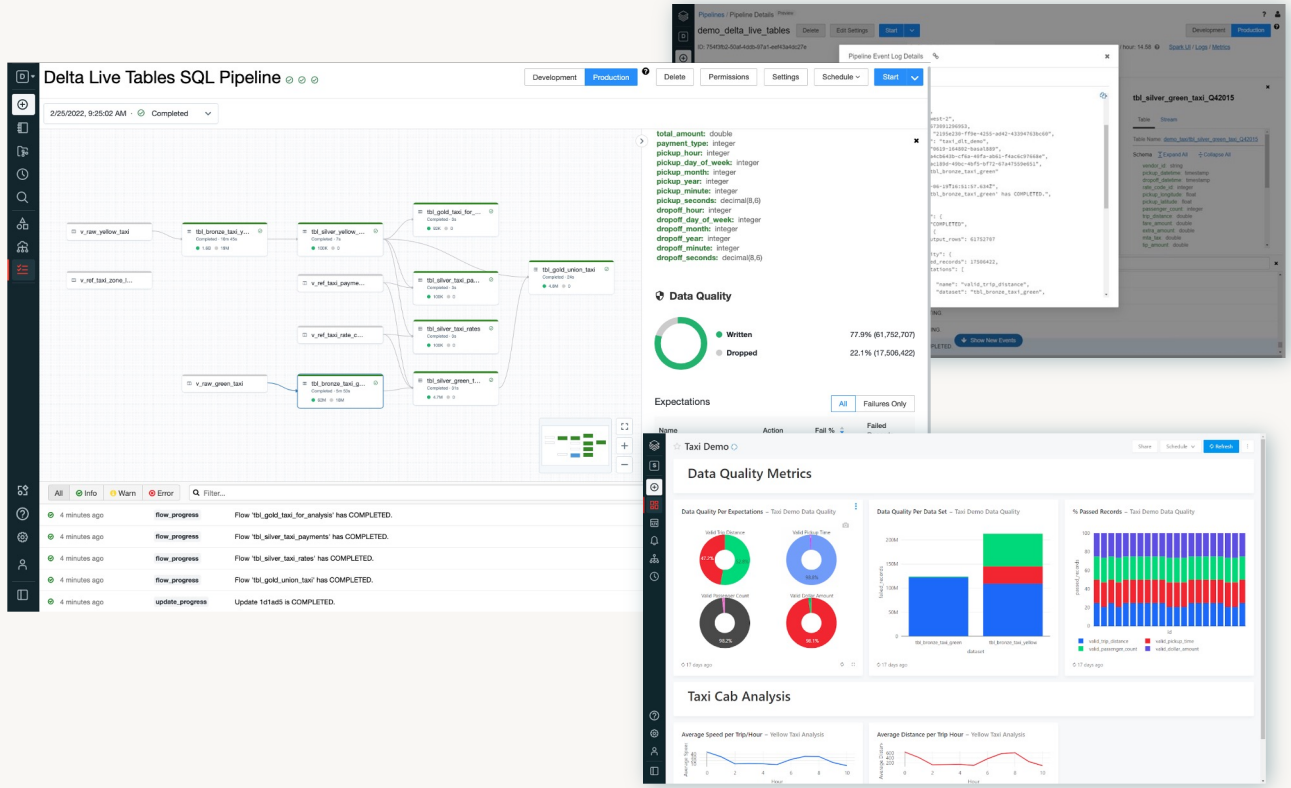
Data quality validation and monitoring

- 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 ...
```



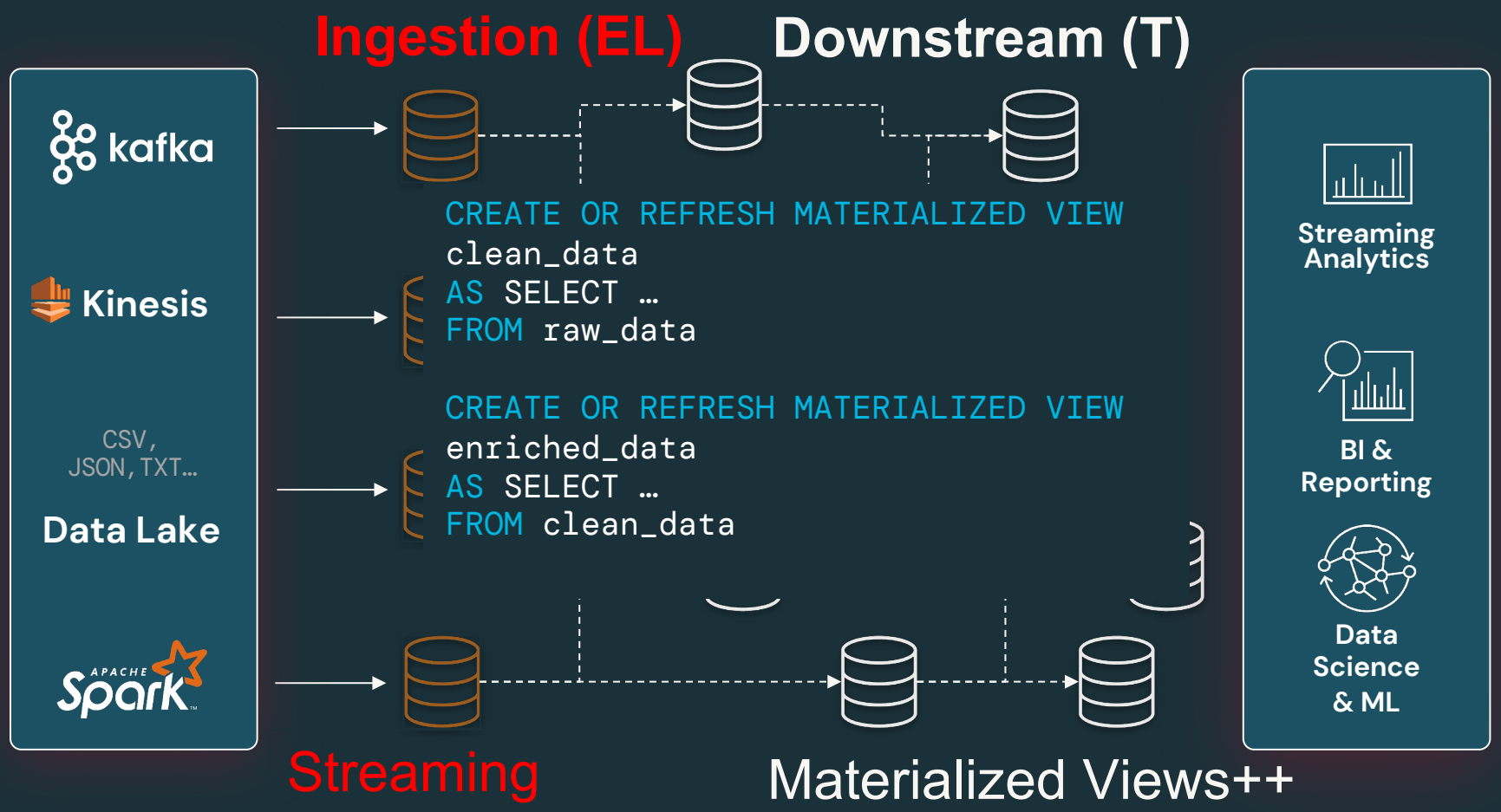
Data pipeline observability



- 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

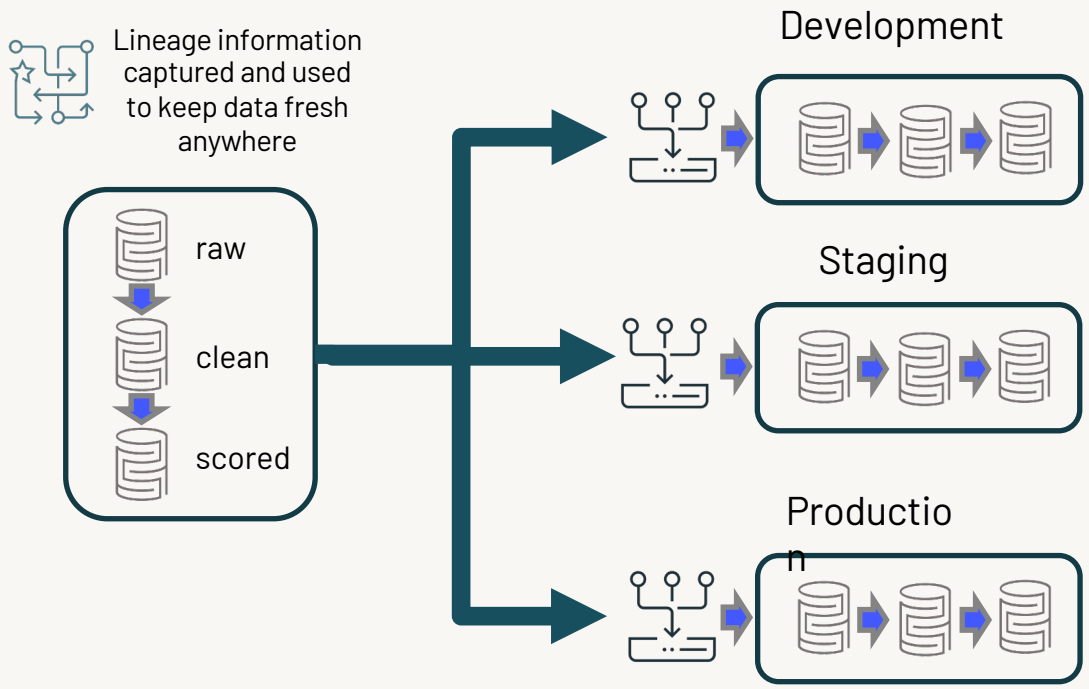
Declarative Pipelines CICD

-> "script" is conceptually executed from scratch

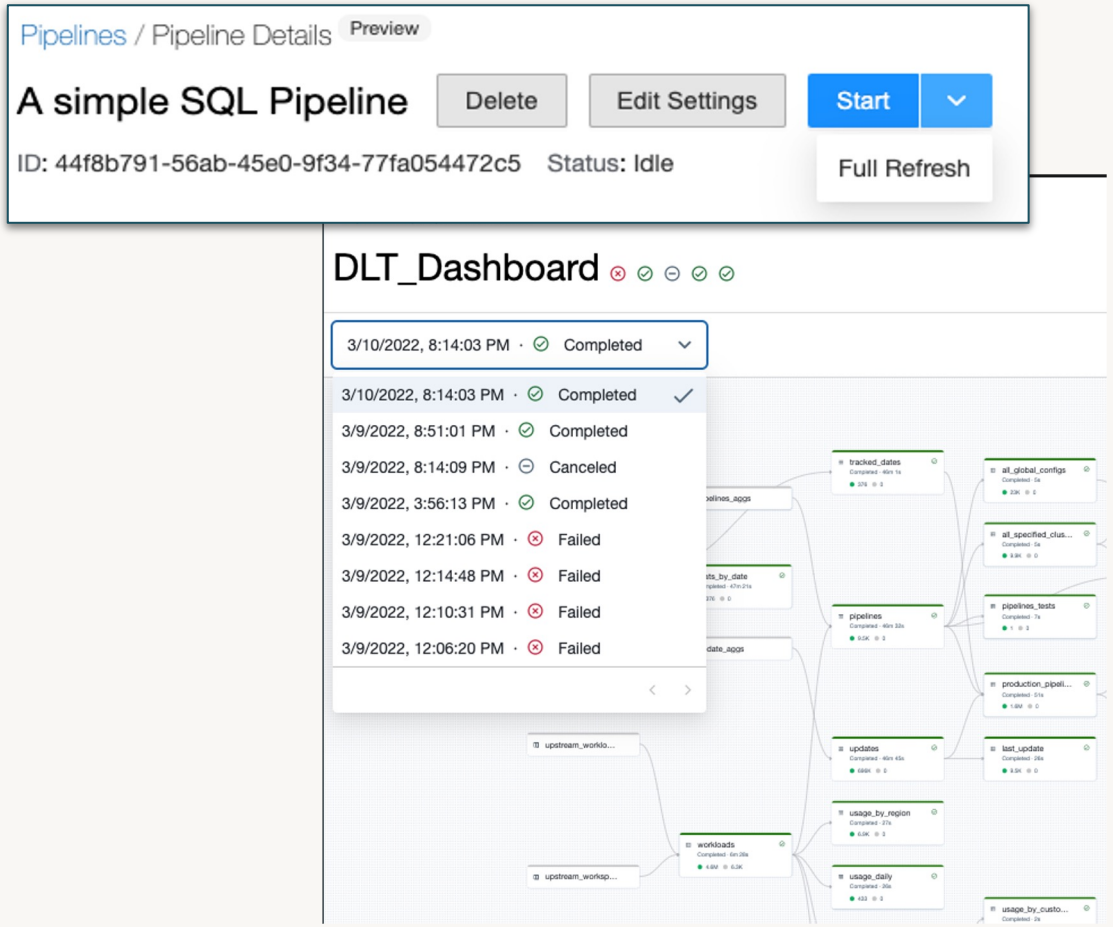


Automated ETL development lifecycle

- Develop in environment(s) separate from production with the ability to easily test it before deploying - entirely in SQL
- Deploy and manage environments using parameterization
- Unit testing and documentation
- Enables metadata-driven ability to programatically scale to 100s of tables/pipelines dynamically



Automated ETL operations

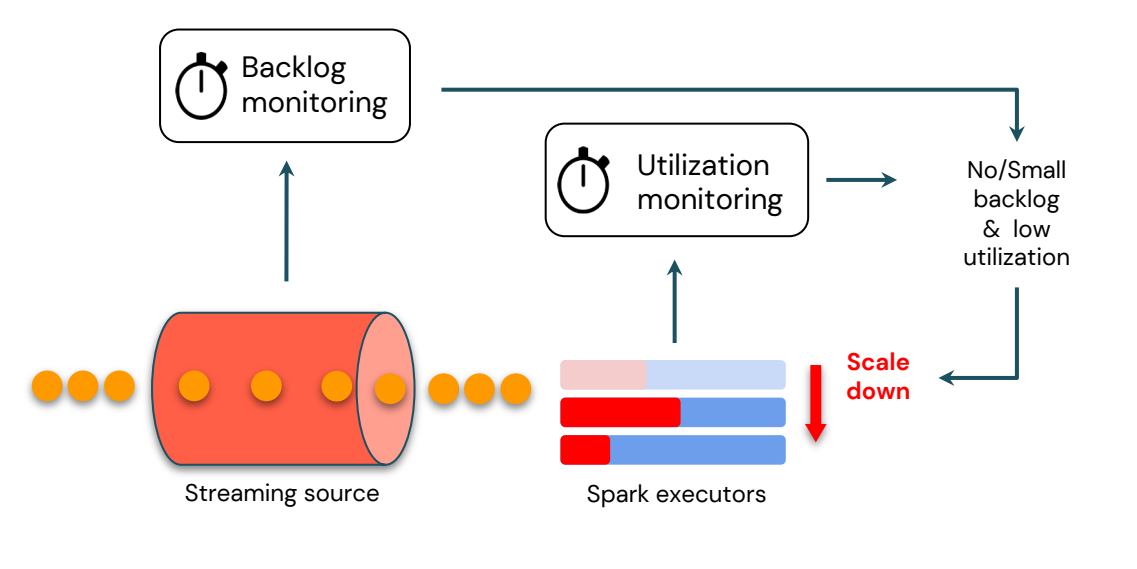


- Reduce down time with automatic error handling and easy replay
- Eliminate maintenance with automatic optimizations of all Delta Live Tables
- Auto-scaling adds more resources automatically when needed.

Enhanced Autoscaling

Save infrastructure costs while maintaining end-to-end latency SLAs for streaming workloads

Problem
Optimize infrastructure spend when making scaling decisions for streaming workloads



- Built to handle streaming workloads which are spiky and unpredictable
- Shuts down nodes when utilization is low while guaranteeing task execution
- Only scales up to needed # of nodes

AWS	Azure	GCP
Generally Available	Generally Available	Public Preview GA Coming Soon

Thank you