BRICK

GUIDE

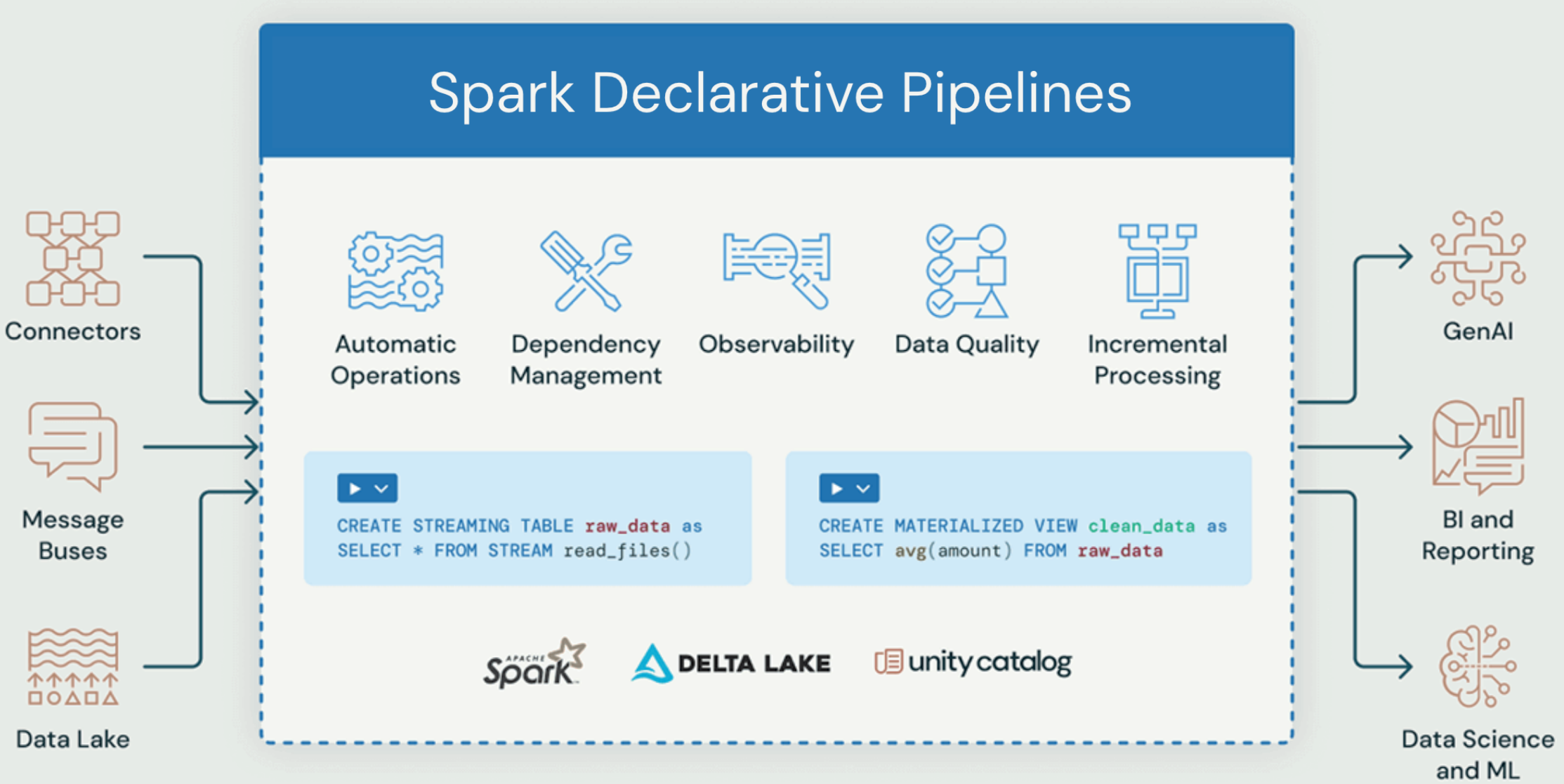
#1

Spark Declarative Pipelines
An Overview

Nov 2025

Lingeshwaran Kanniappan

Develop, Operate and Monitor pipelines ALL in one place



Simple Declarative Framework ⚙️

Infrastructure is managed for you ✅

Build Custom Pipelines in no-time 🚀

NOV 2025

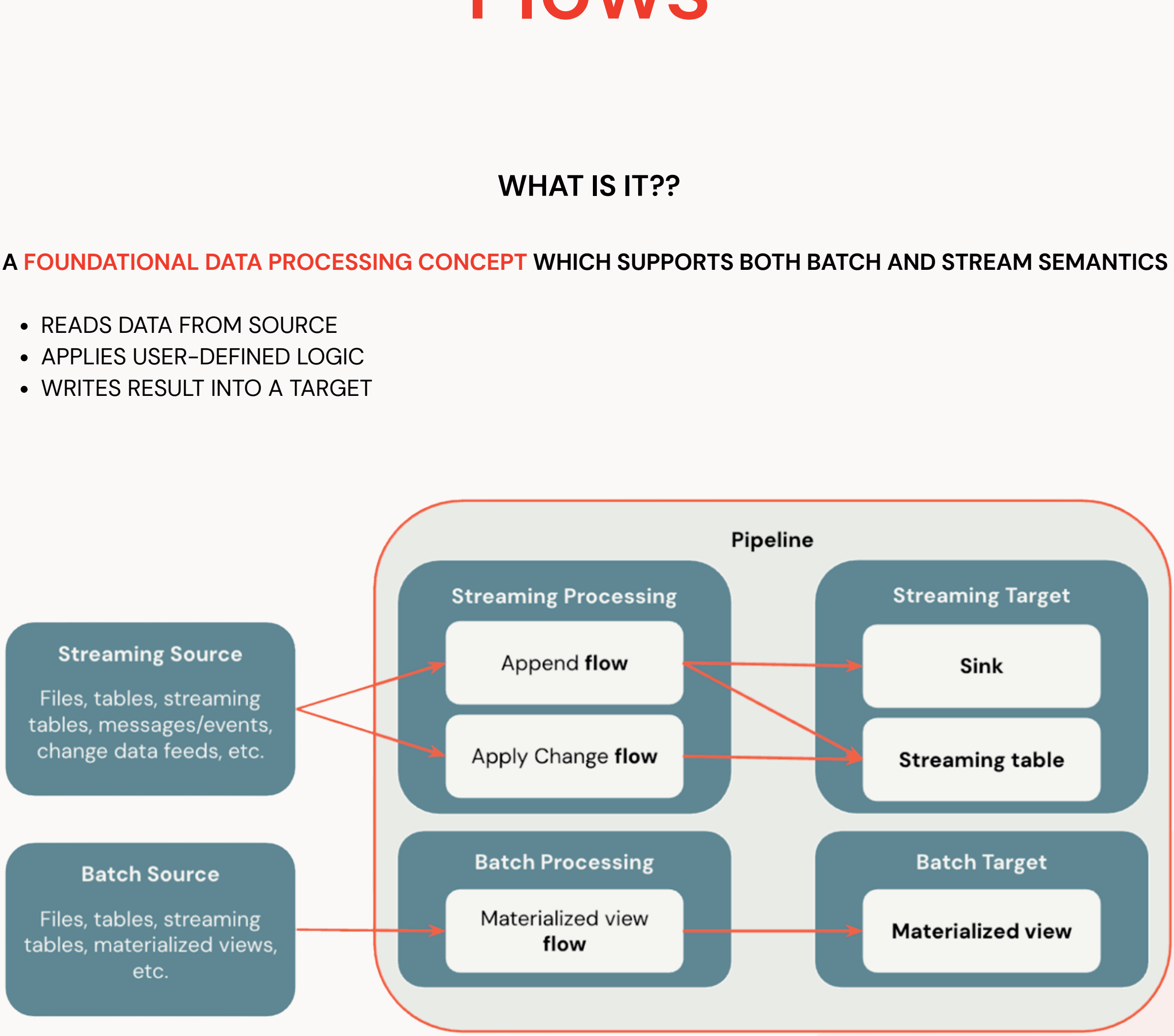
Lingeshwaran Kanniappan

Flows

WHAT IS IT??

A FOUNDATIONAL DATA PROCESSING CONCEPT WHICH SUPPORTS BOTH BATCH AND STREAM SEMANTICS

- READS DATA FROM SOURCE
- APPLIES USER-DEFINED LOGIC
- WRITES RESULT INTO A TARGET



Lingeshwaran Kanniappan

Streaming Tables (STs)

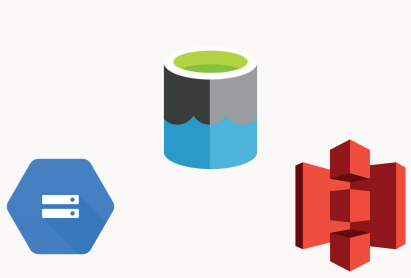
WHAT IS IT??

A SPECIAL TYPE OF TABLE FOR INGESTING AND PROCESSING STREAMING DATA

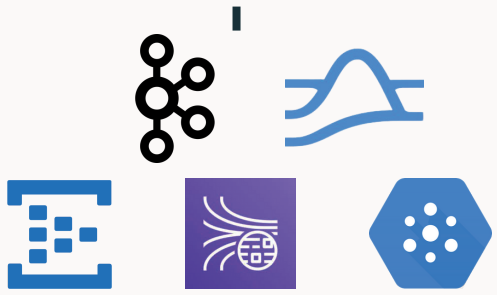
- SUPPORTS REAL-TIME ANALYTICS
- HIGH VOLUME , INCREMENTAL INGESTION
- SIMPLE SQL SYNTAX

```
CREATE STREAMING TABLE web_clicks
AS
SELECT *
FROM STREAM
  read_files('s3://mybucket')
```

```
CREATE STREAMING TABLE server_logs
AS
SELECT from_json(...) data
FROM STREAM
  read_kafka(...)
```



Cloud Storage
(S3, ADLS, GCS)



Message Queues
(Kafka, Pub/Sub, Kinesis, etc.)

Lingeshwaran Kanniappan

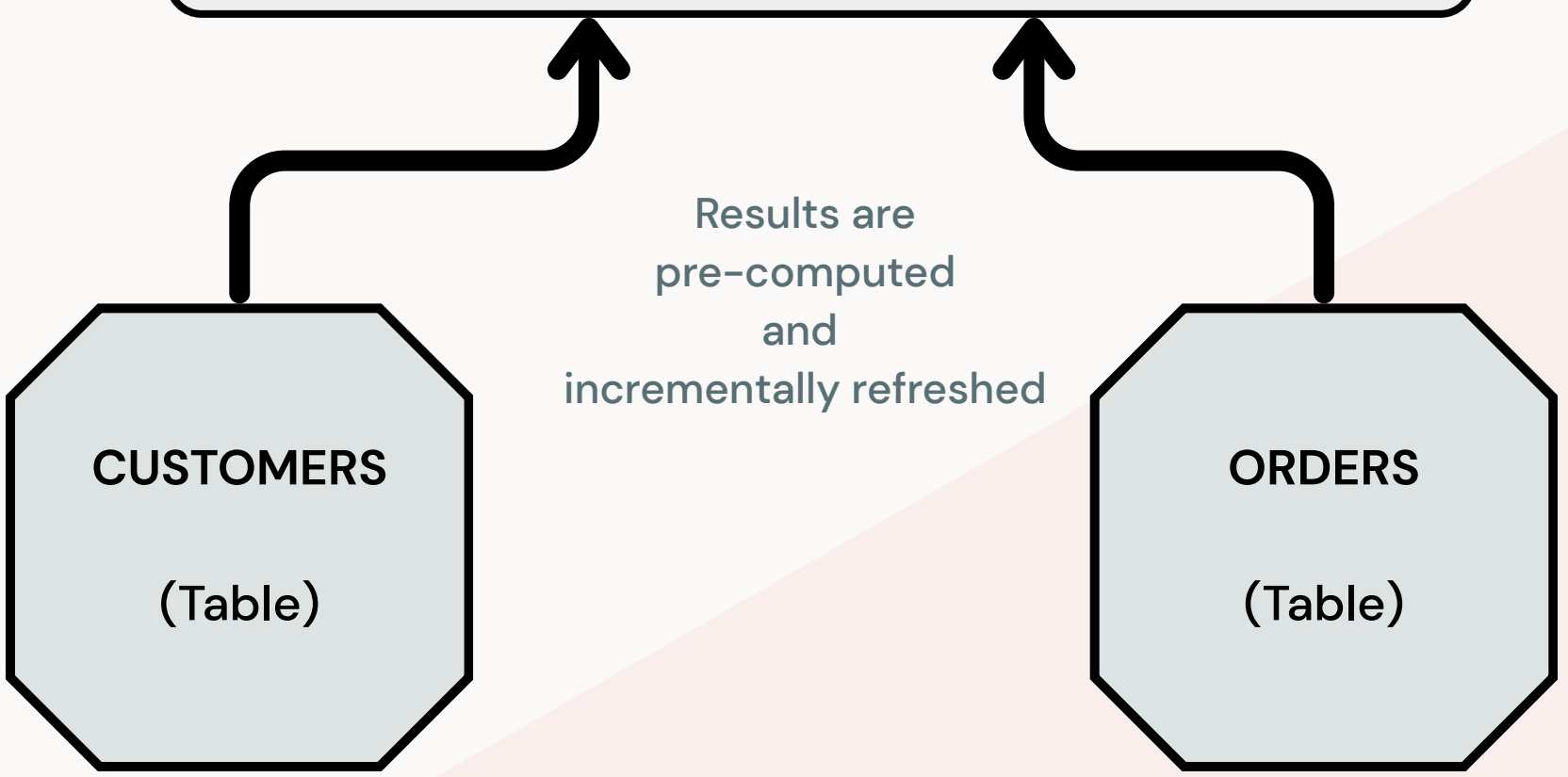
Materialized Views (MVs)

WHAT IS IT??

IT STORES THE RESULT OF A QUERY AS A PHYSICAL TABLE.

- IT IS NOT A REGULAR DATABASE VIEW, BUT DERIVES THEIR DATA FROM THE UNDERLYING TABLE
- USE WHEN COMPLEX JOINS AND AGGREGATIONS ARE FREQUENTED
- FASTER RESPONSE TIMES ARE ACHIEVED.

```
CREATE MATERIALIZED VIEW customer_orders
AS
SELECT
  customers.name,
  sum(orders.amount),
  orders.orderdate
FROM orders
  LEFT JOIN customers ON
    orders.custkey = customers.c_custkey
GROUP BY
  name,
  orderdate;
```



Lingeshwaran Kanniappan

Streaming Tables

vs.

Materialized Views

vs.

Flows

Streaming tables

- Incremental processing
- Exactly once

Best For

- Data ingestion
- Append-only workloads
- Low latency streaming

Materialized views

- Cached query results
- Pre-computed for fast access
- Can be incremental

Best For

- Complex transformations
- Analytical queries
- Always-correct results

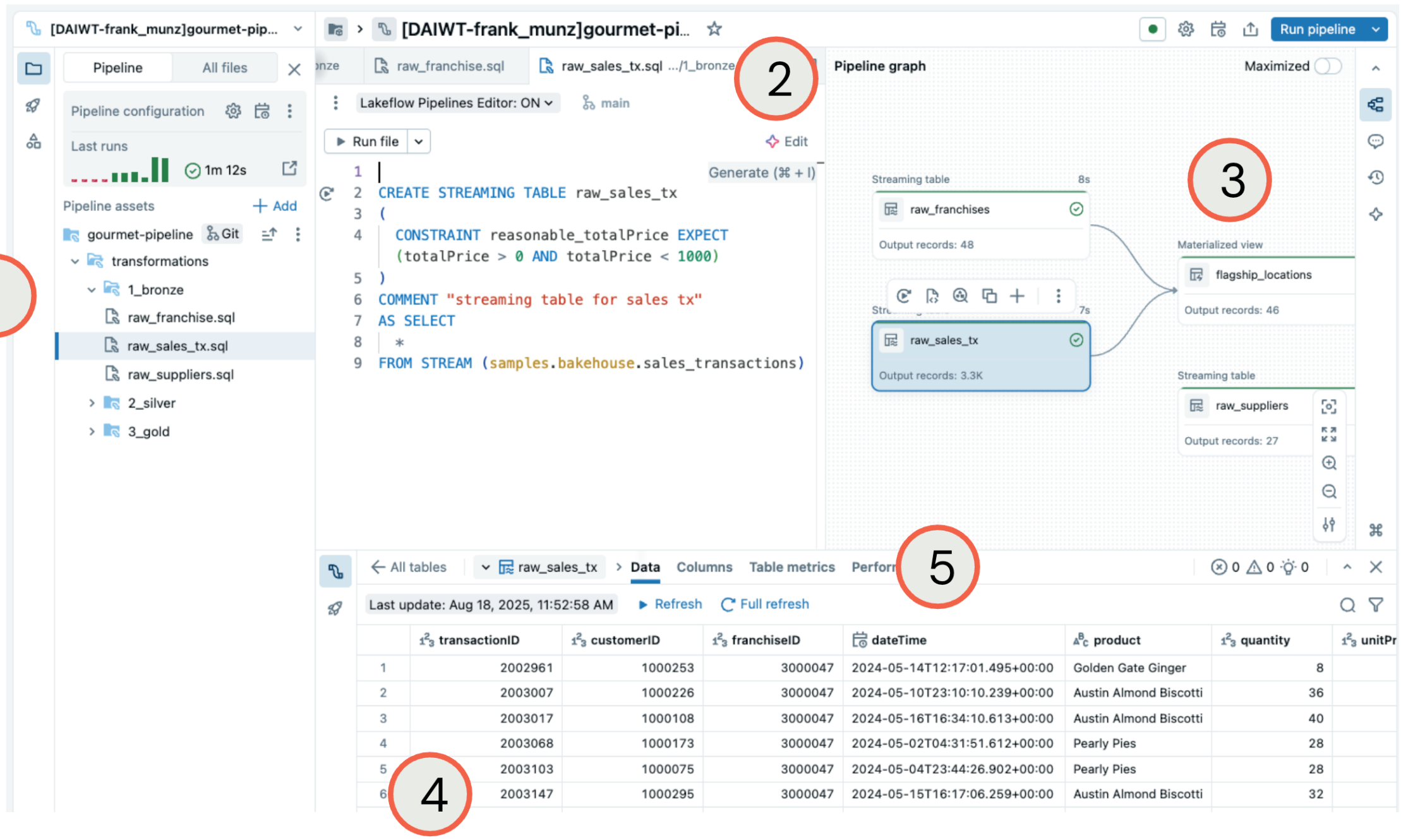
Flows

- Powers both Streaming Tables & Materialized Views
- Supports both streaming and batch semantics

NOV 2025

Lingeshwaran Kanniappan

The Experience



1. **Pipeline asset browser**
2. **Multi-file code editor with features for step-by-step development**
   a. **Soft tabs**
   b. **Gutter actions**
   c. **Inline error indicators**
3. **Interactive DAG**
4. **Data previews**
5. **Execution insights panels**

NOV 2025

Lingeshwaran Kanniappan

Say what to be done

NOT how to do it

Spark imperative program

```
def upsertToDelta(microBatchOutputDF: DataFrame, batchId:
Long) {
  microBatchOutputDF
    .groupBy("key")
    .agg(max_by("ts", struct("*").alias("row"))
    .select("row.*")
    .createOrReplaceTempView("updates")

  microBatchOutputDF.sparkSession.sql(s"""
    MERGE INTO cdc_data_raw t
    USING updates s
    ON s.key = t.key
    WHEN MATCHED AND s.is_delete THEN
      UPDATE SET DELETED_AT=now()
    WHEN MATCHED THEN UPDATE SET
      A=CASE WHEN s.ts > t.ts THEN s.a ELSE t.a,
      B=CASE WHEN s.ts > t.ts THEN s.b ELSE t.b,
      … for every column …
    WHEN NOT MATCHED THEN INSERT *
  """)
}

cdcData.writeStream
  .foreachBatch(upsertToDelta _)
  .outputMode("append")
  .start()
```

Declarative Pipelines Program

```
AUTO CDC INTO cdc_data
FROM source_data
KEYS (id)
SEQUENCE BY ts
APPLY AS DELETE WHEN is_deleted
```

A super HUGE Spark method

is now

JUST 5 LINES on SDP

NOV 2025

Lingeshwaran Kanniappan

Important Considerations

- Function used to define a dataset MUST return a Spark DataFrame
- Never use methods that save or write to files or tables as part of your SDP dataset code e.g. collect(), count(), toTable()
- When using the for loop pattern to define datasets in Python, ensure that the list of values passed to the for loop is always additive.
- In Lakeflow (managed), work that incurs cost is the pipeline update execution when the pipeline is updated. Planning evaluations don't incur costs.

Lingeshwaran Kanniappan

Remember the limitations!

- 200 concurrent pipeline updates per workspace
- Datasets can only be defined ONCE (except streaming tables with append flow)
- Identity columns not supported with AUTO CDC targets yet!
- pivot() function is not supported
- Materialized views/streaming tables accessible ONLY via Databricks clients (use sink API for external access)
- Delta time travel works ONLY with streaming tables, not materialized views

SDP Serverless is now 26% better TCO than Classic

Full Support for Unity Catalog
Leverage optimisations like Liquid Clustering
Event logs are captured in System Tables

Check out the stats below!!

52%

Avg. productivity gains

47%

Avg. use case time-to-market gain

**Based on 2025 customer survey by Databricks (n= 3,350+)

NOV 2025

Lingeshwaran Kanniappan

Why wait ?!

We are a Leader in ETL and Streaming Space

Forrester Wave: Data Lakehouse

Forrester Wave: Cloud Data Pipelines

IDC MarketScape: Analytic Stream Processing

NOV 2025

Lingeshwaran Kanniappan

THANK YOU!!

Follow for more tips!!!