



POWERED BY  DUG

DYNAMICS CON LIVE

MAY 2024

A “House on the Lake”

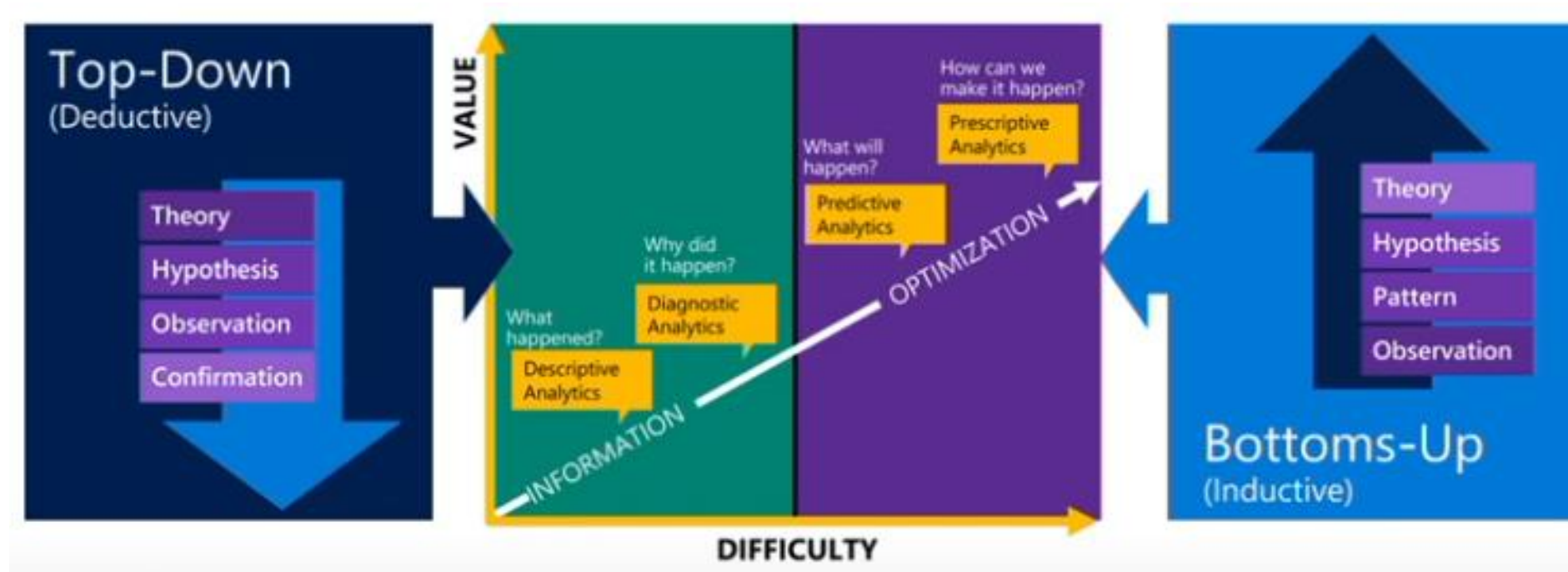
Bill Spurling

Software Architect - Ellipse Solutions



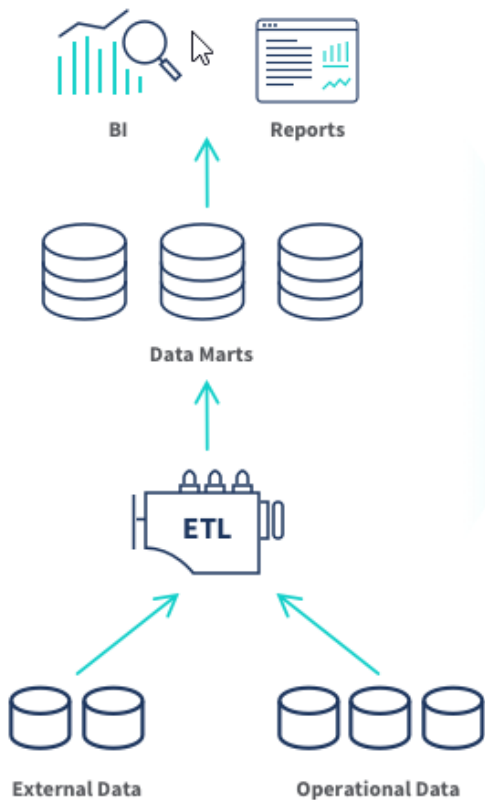


Data Value



Data warehouse

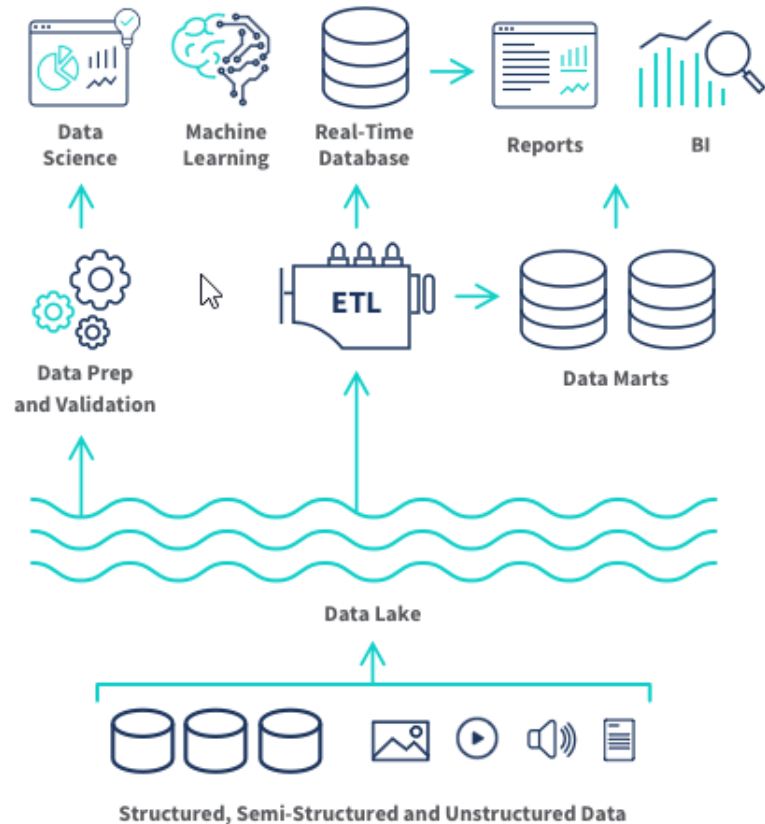
Data Warehouse



- Data warehouses hold highly structured and unified data to support specific business intelligence and analytics needs
- Data is usually transformed and fit into defined schemas
- Not appropriate for storing unstructured or semi-structured data.

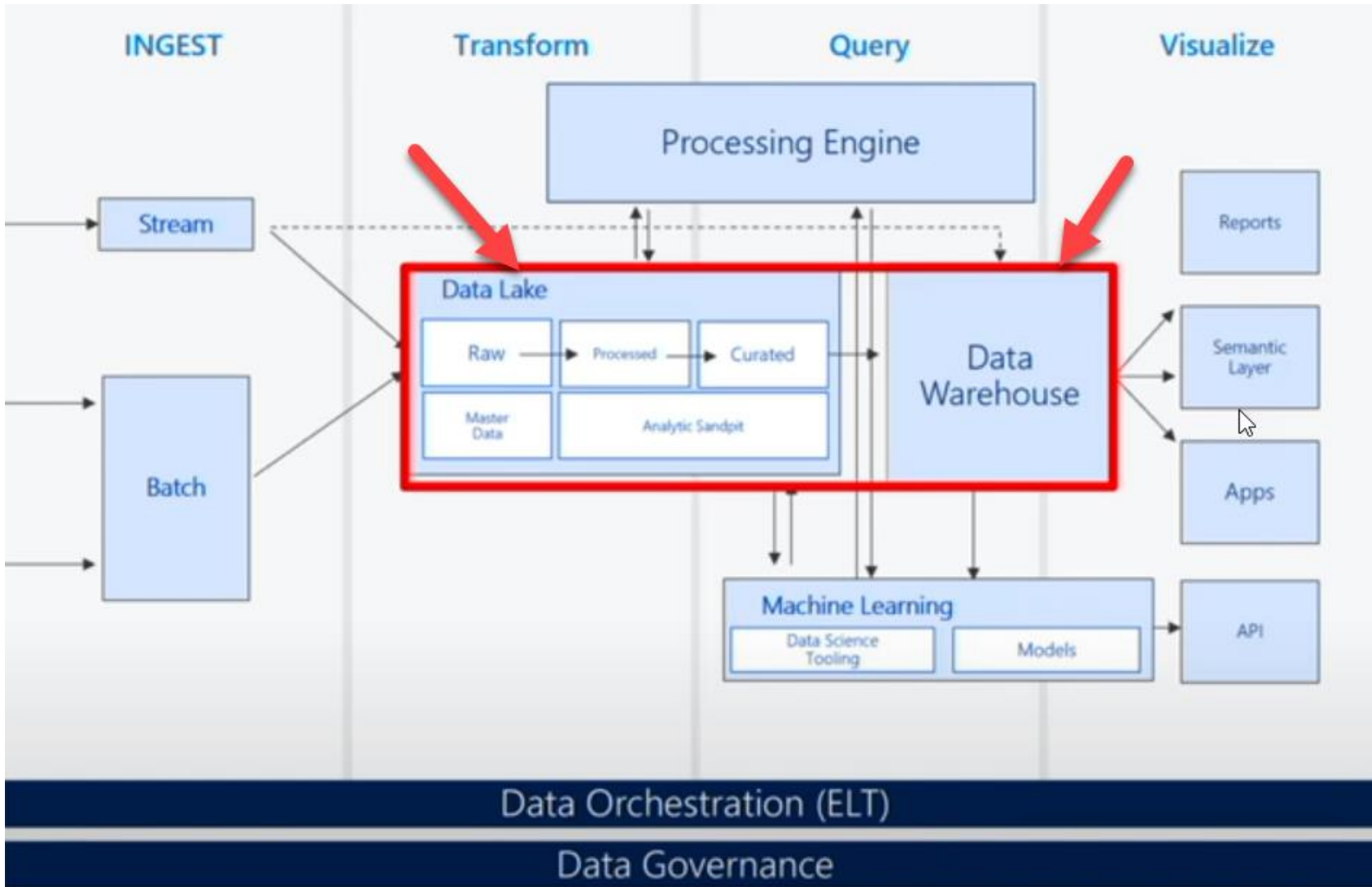


Modern data warehouse pattern



- Massive volumes of structured and unstructured data
- Data is available for use far faster by keeping it in a raw state
- Data teams can build data pipeline transformations





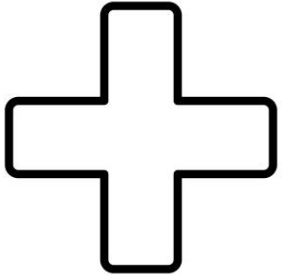
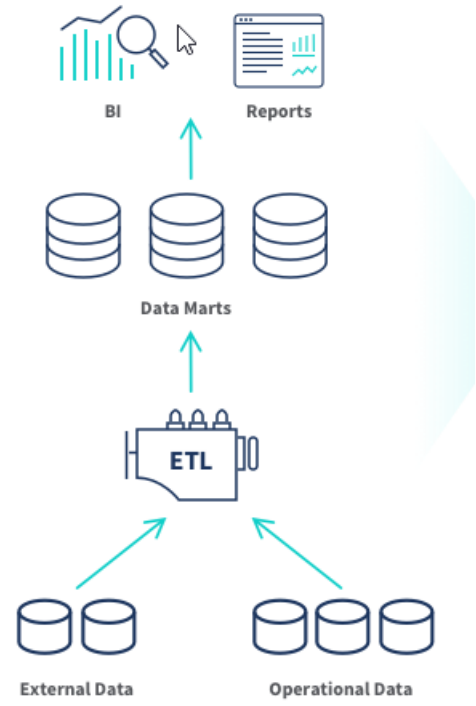
Challenges of the data lake

- Lack of transaction support
- Hard to enforce data quality
- Its hard/complicated to mix appends, updates, and deletes in the data lake
- It can lead to challenges around data governance in the lake itself, leading to data swamps and not data lakes
- It has multiple storage layers – different zones and file types in the lake PLUS the data warehouse itself PLUS often the BI tools.

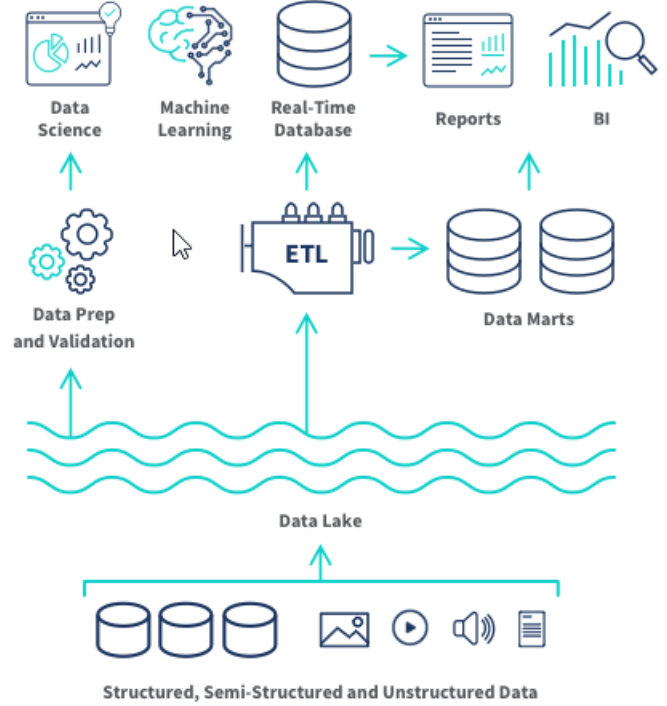


So, what is a Lakehouse

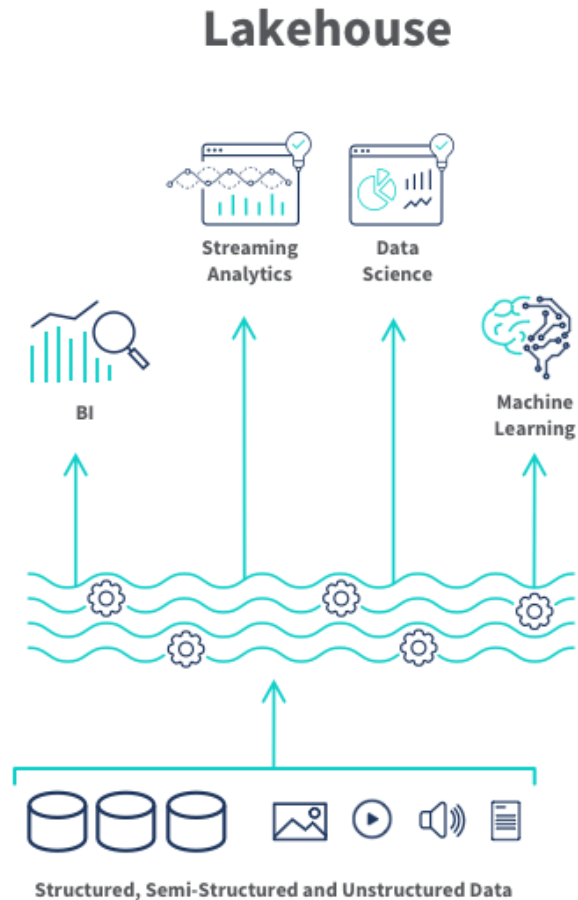
Data Warehouse



Data Lake



So, what is a Lakehouse

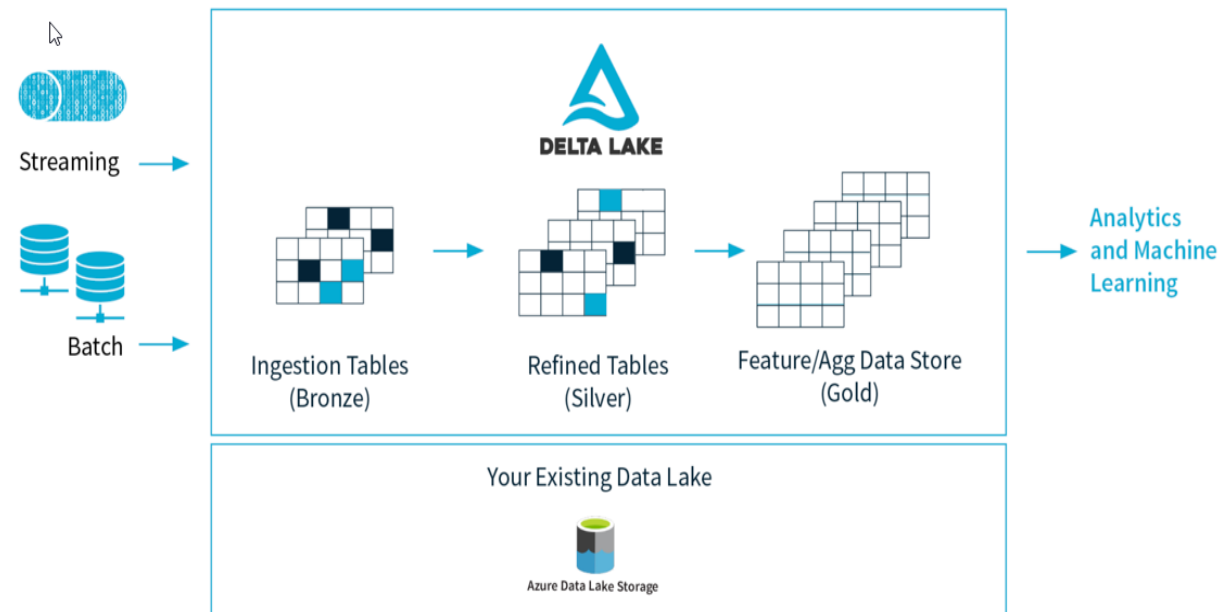


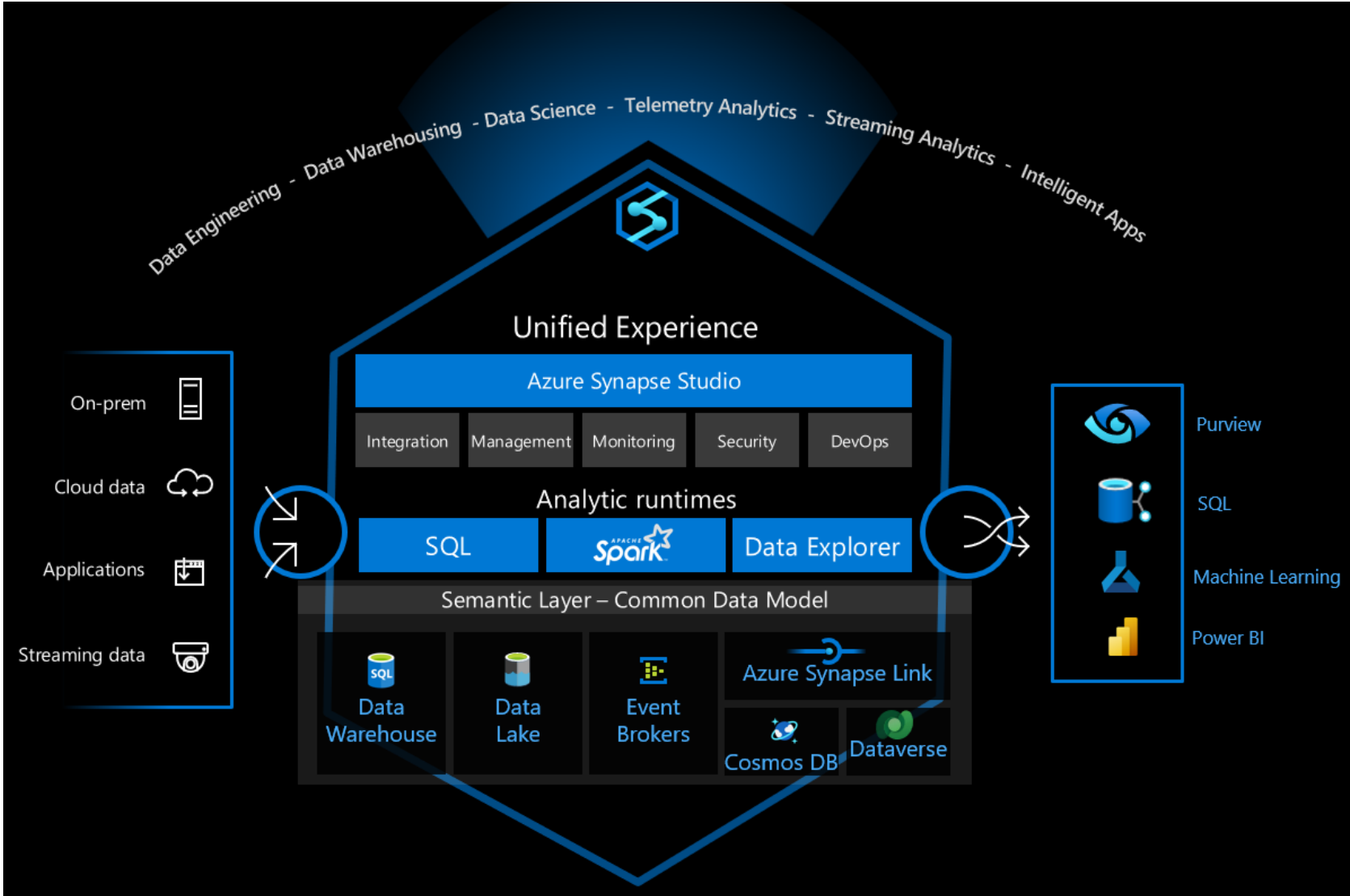
- There is no separate data warehouse and data lake. The data lake IS the data warehouse.
- It uses open file formats (Parquet) and metadata layers (Delta Lake) to store and manage data in cloud object storage (Azure blob storage)
- It provides high-performance SQL analysis and optimized access for data science and machine learning tools
- It supports data governance, data sharing, data auditing, and data discovery.



Huh? Delta Lake?

- The secret sauce of the Lakehouse pattern and looks to resolve the challenges highlighted earlier.
- The Delta Lake is an open-source project that allows data warehouse like functionality directly ON the data lake.





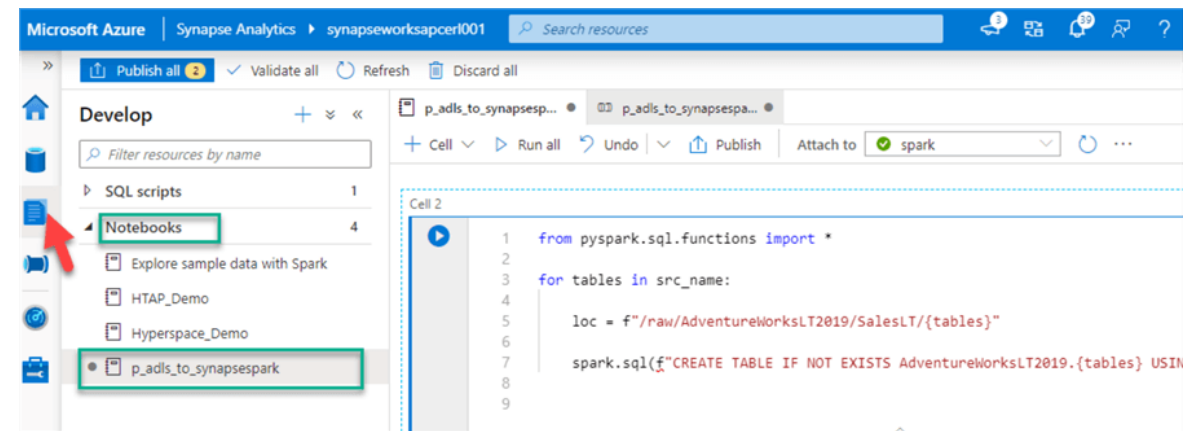
Synapse Pipelines & Data Flows

- No/low code transformation
- Graphical utility for building data pipelines
- Handles data orchestration , data movement, and data transformations
- Powerful scale out engine, built on top of Apache Spark
- Supports Lakehouse pattern via Delta inline Datasets.



Data Engineering with Spark

- Code first Data Engineering
- PySpark, SQL, and C# language supported
- Author multiple languages in a single notebook
- Analyze & transform data from data warehouse, data lake, and real-time operational data from one place
- Synapse Spark uses Spark 3.2 runtime, which includes Delta Lake 1.0



The screenshot shows the Microsoft Azure Synapse Analytics interface. The left sidebar displays the 'Develop' section with a search bar and a list of resources: 'SQL scripts' (1), 'Notebooks' (4), 'Explore sample data with Spark', 'HTAP_Demo', 'Hyperspace_Demo', and 'p_adls_to_synapsepark'. A red arrow points to the 'Notebooks' section. The main area shows a notebook cell with the following PySpark code:

```
1 from pyspark.sql.functions import *
2
3 for tables in src_name:
4
5     loc = f"/raw/AdventureWorksLT2019/SalesLT/{tables}"
6
7     spark.sql(f"CREATE TABLE IF NOT EXISTS AdventureWorksLT2019.{tables} USIN
8
9
```

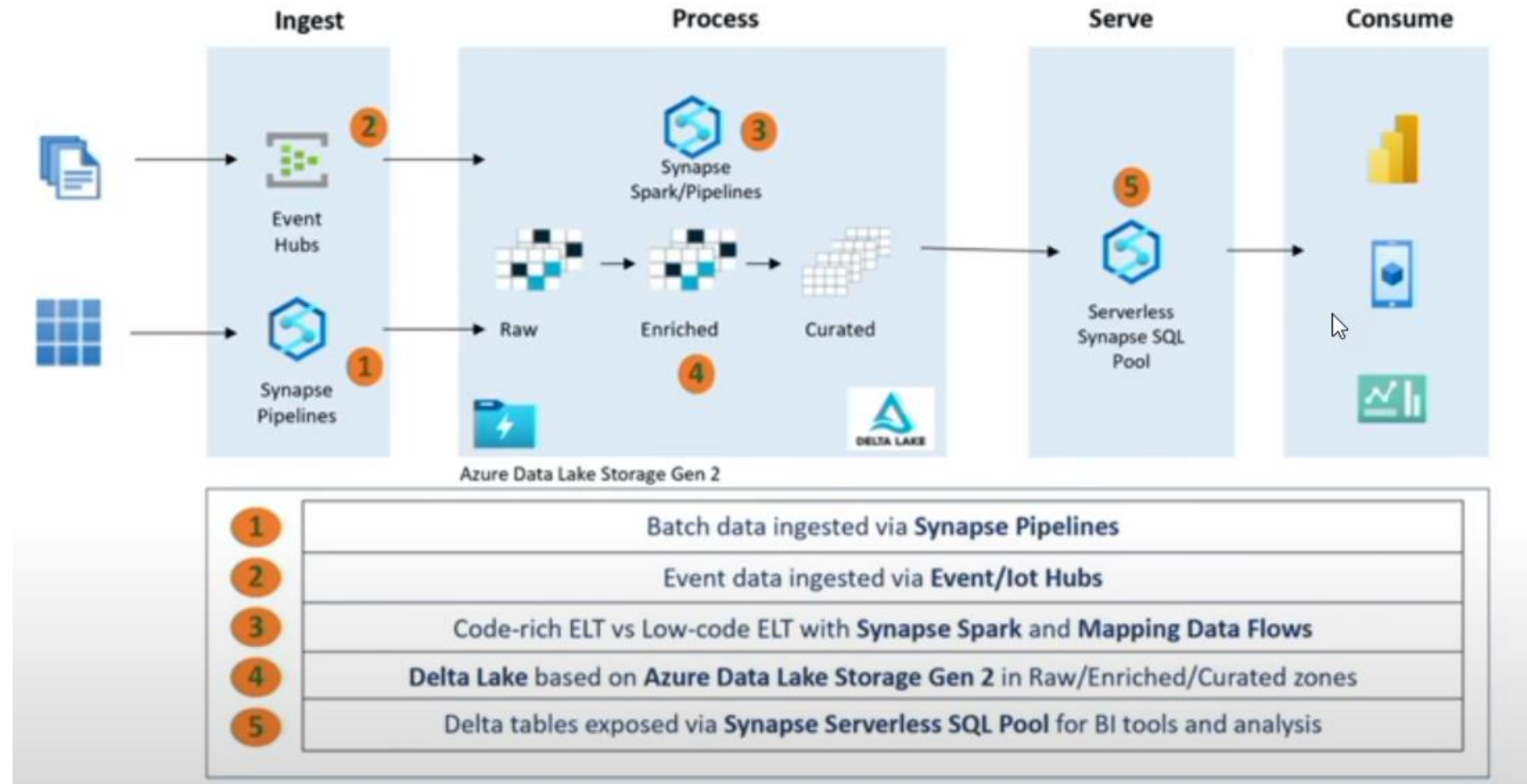


Serverless SQL Pools

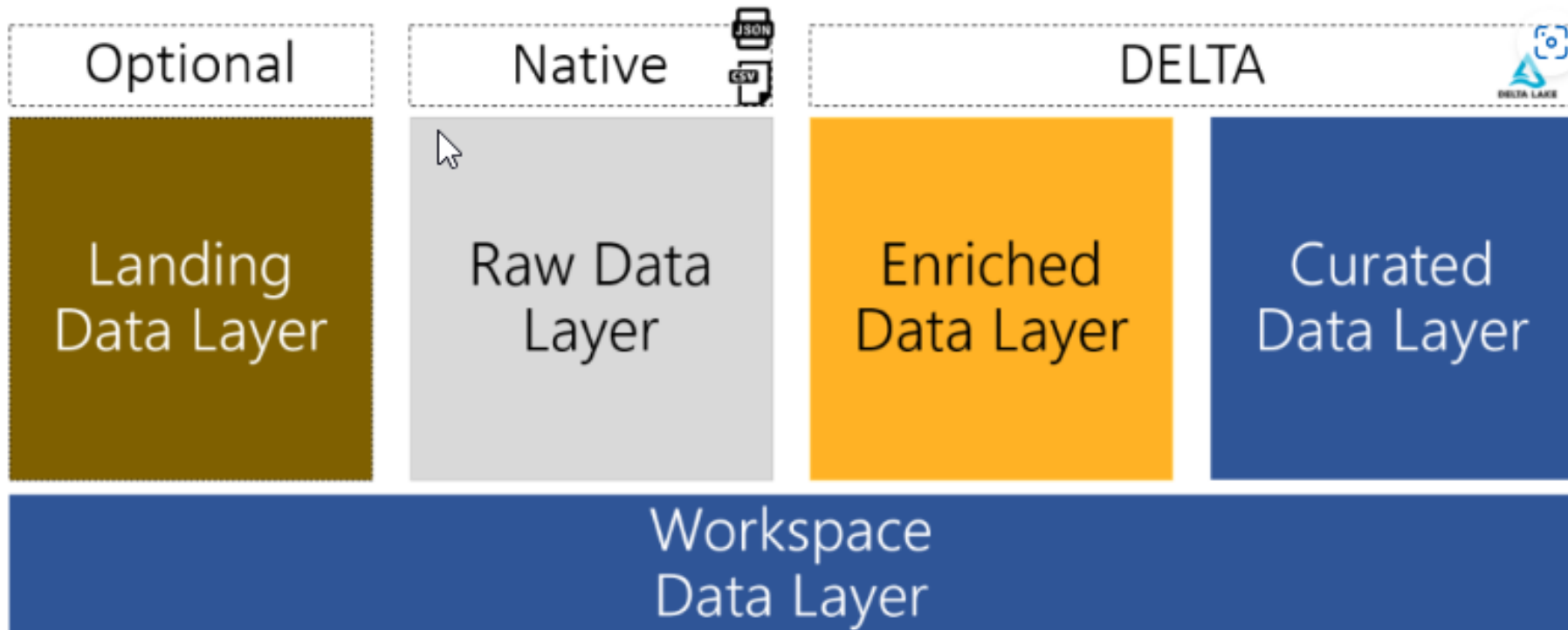
- Flexible consumption models
- Serverless pay-per-query ideal for logical data warehouse (lakehouse), ad-hoc data lake exploration and transformation
- TSQL code directly in data lake
- Support for Delta Lake file system via OPENROWSET
- Supports DELTA format



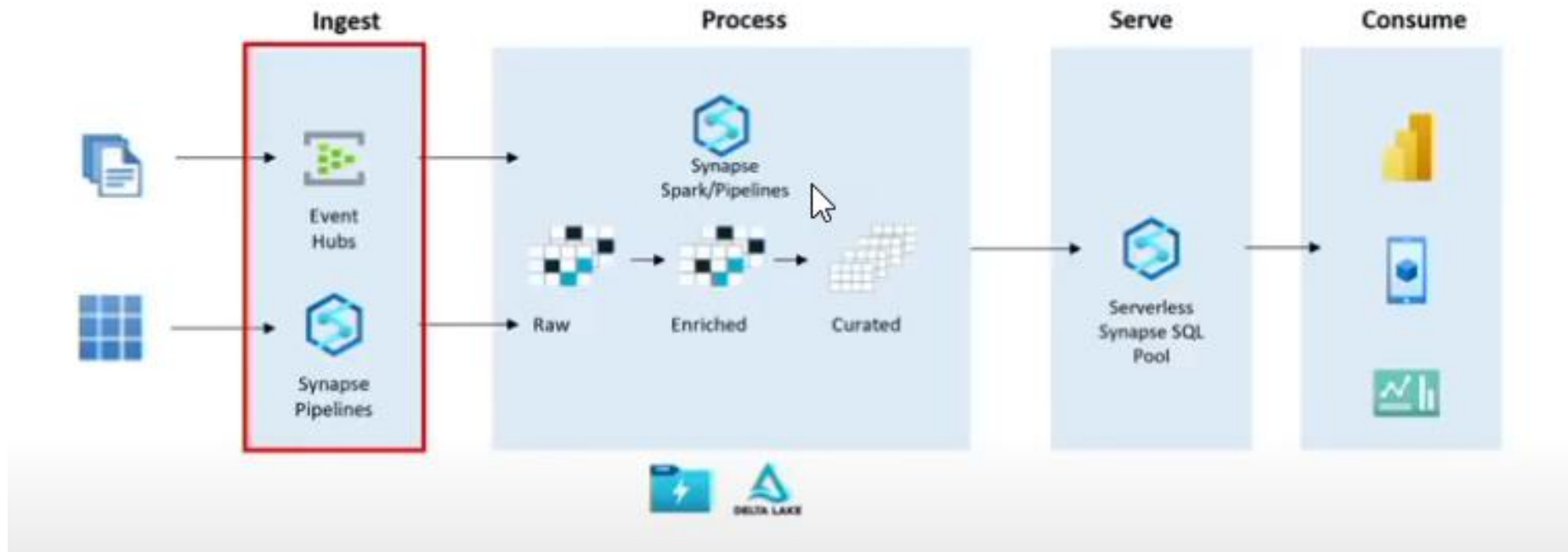
Lakehouse Architecture on Synapse



Data Lake Layers



ELT Pattern - Ingestion



Ingestion (Streaming)

- Ingest via message broker such as event hubs
- Either transform Stream Analytics or Spark
- Option to sink directly to Data Lake with event hub captures
- Sink options vary depending on use

ELT Streaming Ingestion Pattern – Stream Analytics

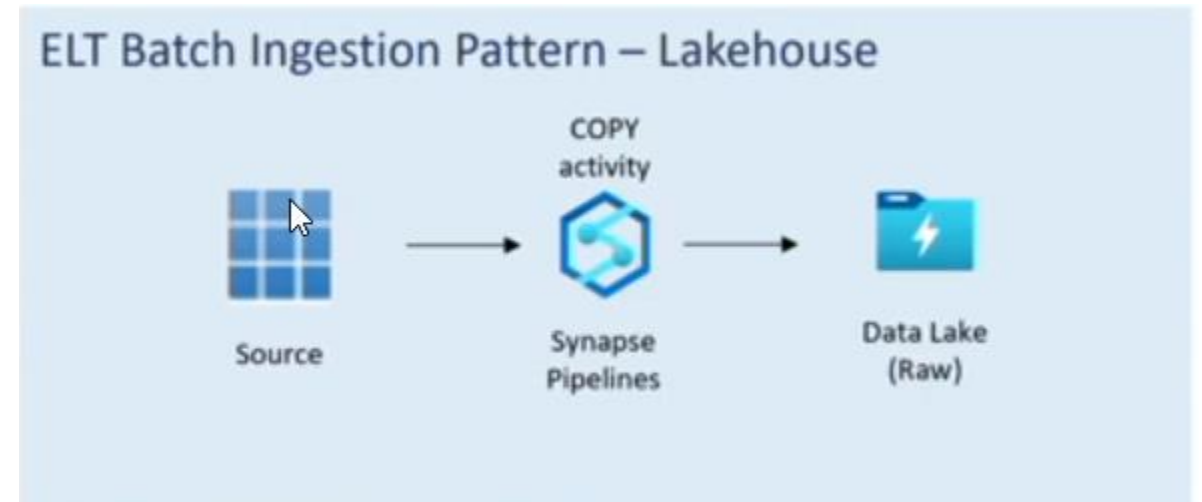


ELT Streaming Ingestion Pattern – Spark

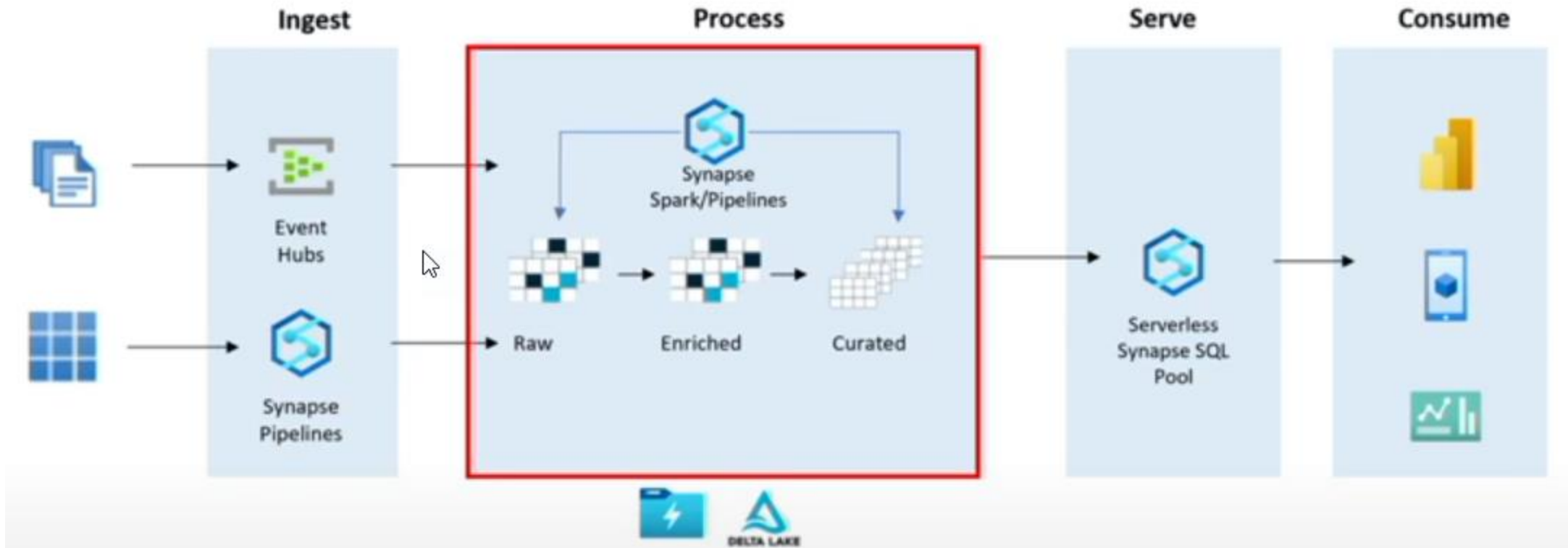


Ingestion (Batch)

- Load into raw zone in native format
- Copy Activity with Synapse Pipeline usual ingestion mechanism
- Can rerun without impact
- Folder path typically follows a raw/source/table [year/month/day]/file.csv pattern

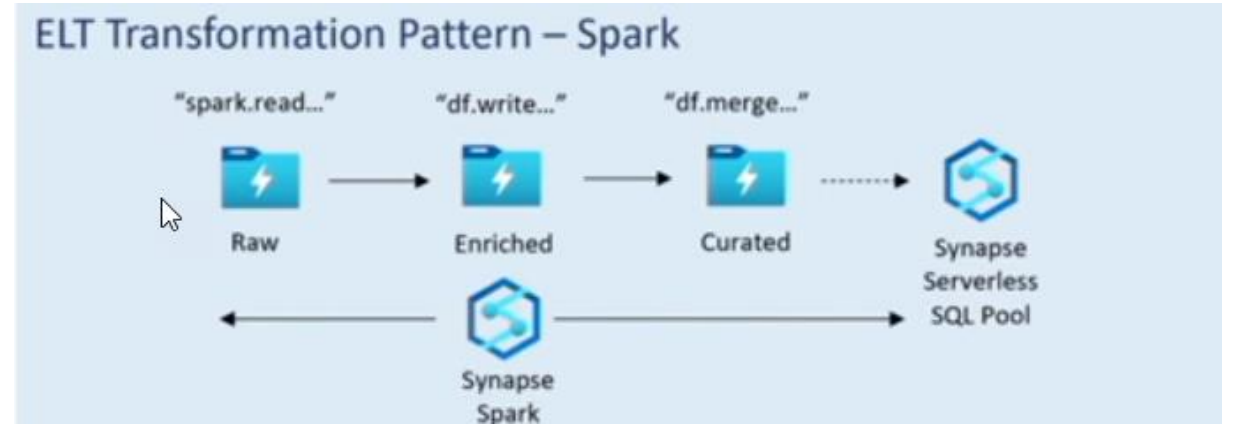


ELT Pattern - Ingestion



Process & Transform (Spark)

- Code centric way of doing ETL/ELT
- Can use SQL , Python, C#
- Read raw files into dataframe
- Apply transform/cleaning
- Apply Delta optimizations via code
- Use Delta format or enriched/curated zones



Process & Transform (Data Flows)

- GUI based
- Define source based on raw files
- Build transforms as tasks in at data flow
- Write to sink (curated zone)- use inline Dataset of type Delta
- Apply Delta optimizations via UI
- Parameterize and add to pipeline

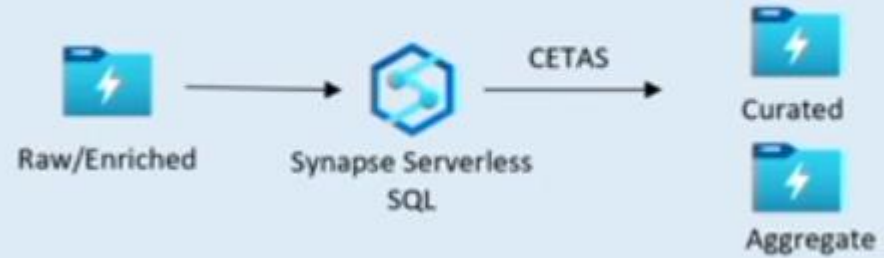
ELT Transformation Pattern – Data Flows



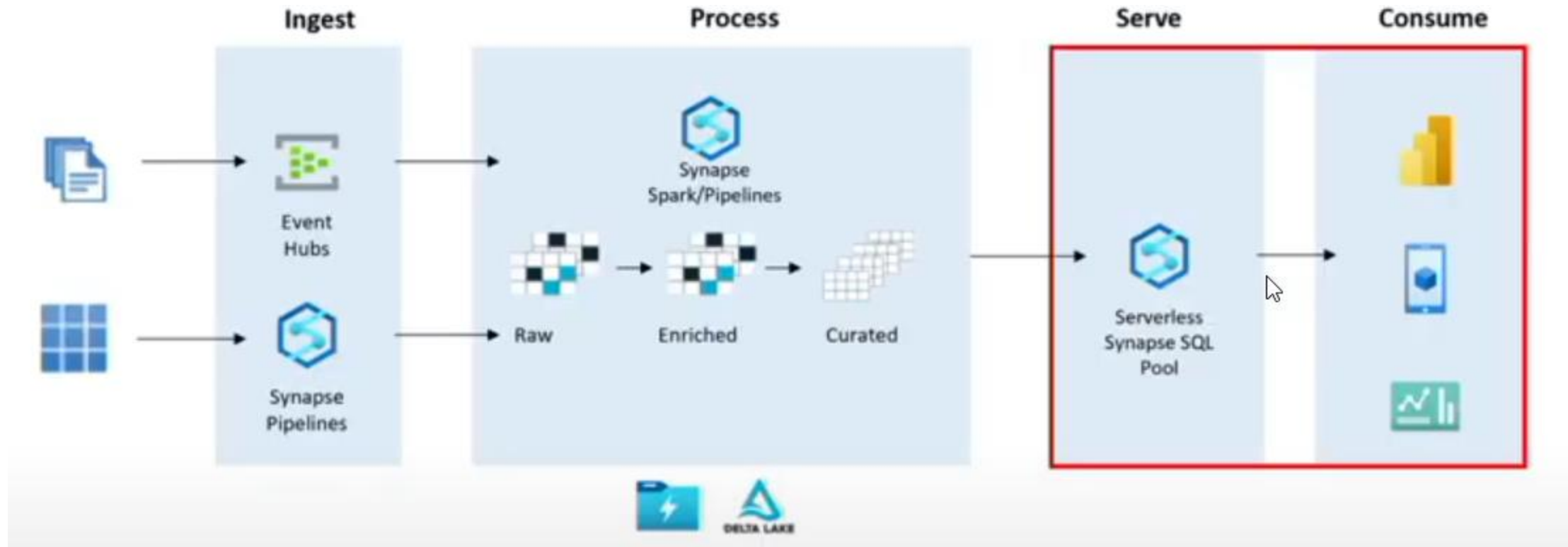
Process & Transform (Serverless SQL)

- Create external table as Select
- Creates new persisted table from select statement
- Good for commonly used aggregations & summary tables
- Outputted dataset is stored in data lake and referenced as external table
- Use parquet as storage format

ELT Transformation Pattern – Serverless SQL with CETAS



ELT Pattern – Serve & Consume



Process & Transform (Serverless SQL)

- Define external objects in Serverless pool pointing to Delta Lake
- Expose as reporting layer
- Connect Power BI using SQL connector or Workspace connector
- Power BI Delta Connector

ELT Transformation Pattern – Serverless SQL with CETAS




Bringing it all Together



| | |
|---|--|
| 1 | Ingest Raw Data in native format (csv, json) |
| 2 | Cleanse, transform and load dimension tables – MDF and/or Spark (delta) |
| 3 | Cleanse, transform and load fact tables – MDF and/or Spark (delta) |
| 4 | Process summary/aggregate tables – CETAS w/ Serverless SQL Pool (parquet) |
| 5 | Generate external SQL objects – views, tables, functions (Serverless SQL Pool) |





Q&A



POWERED BY  DUG

DYNAMICS CON LIVE

MAY 2024

