



Transforming Data into Gold: A Live Demonstration of Microsoft Fabric Lakehouse

Data Saturday, Copenhagen, 8/2-2025

twoday

Sponsors



GOLD  redgate



SILVER CatMan Solution



BI WISE
BUSINESS INTELLIGENCE



twoday

Agenda

Transforming Data into Gold

Fabric is Fantastic – but not as easy as Microsoft marketing says


Starting-point Fabric Data Lakehouse Architecture

AquaShack Data Lakehouse Accelerator

Layers in the Data Lakehouse

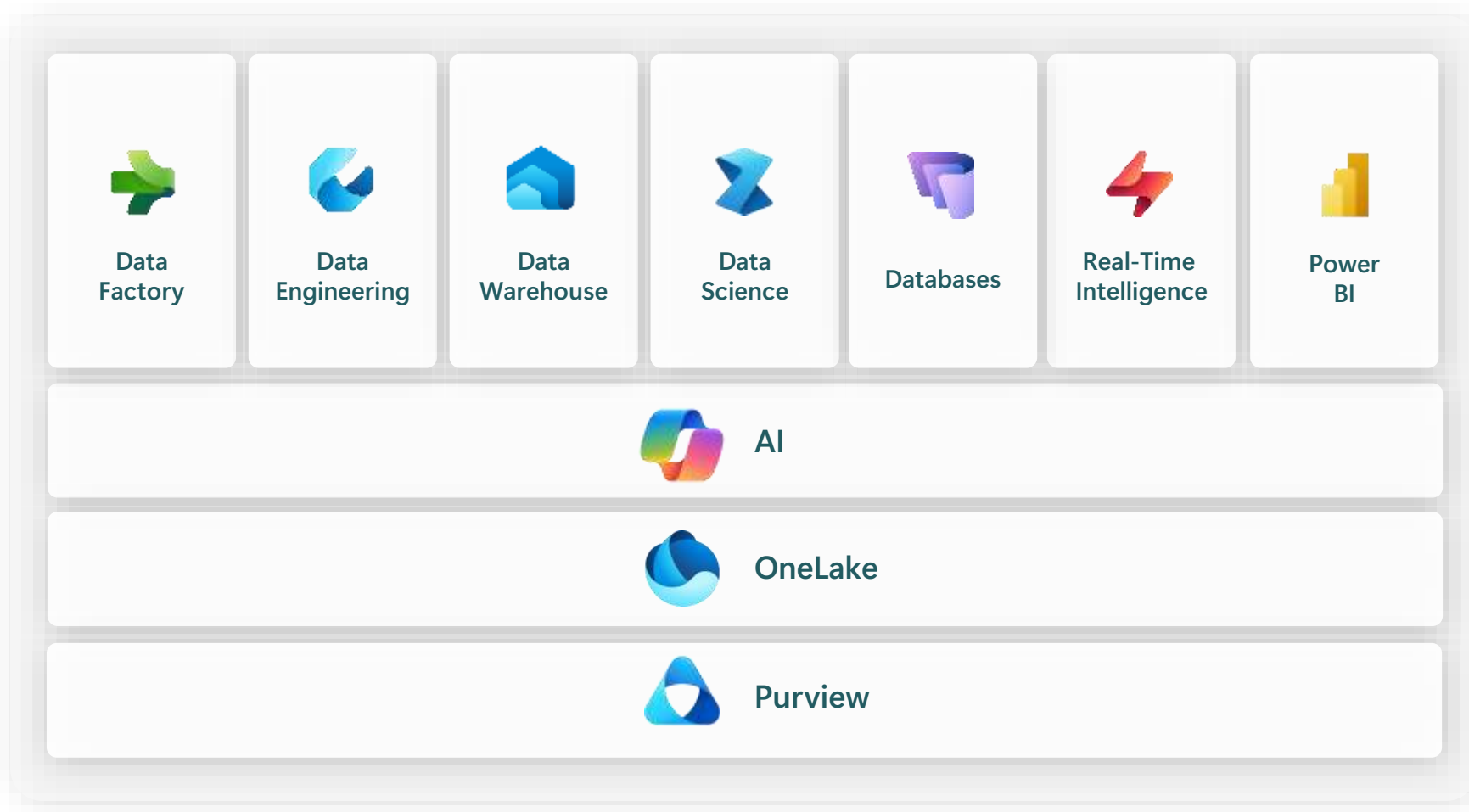
Just Blindbæk

- Principal Architect at twoday
 - Pre-sales and marketing
 - Internal practice development
 - Academy: External training
- Microsoft Data Platform MVP
- Found and organizer of
 - Danish Microsoft BI Community (MsBIP.dk)
 - Power BI UG Denmark (PowerBI.dk)
 - Power BI Next Step and Data Platform Next Step

 [linkedin.com/in/blindbaek/](https://www.linkedin.com/in/blindbaek/)

twoday

Fabric: End-to-end unified analytical platform



Software as a Service

Seven workloads

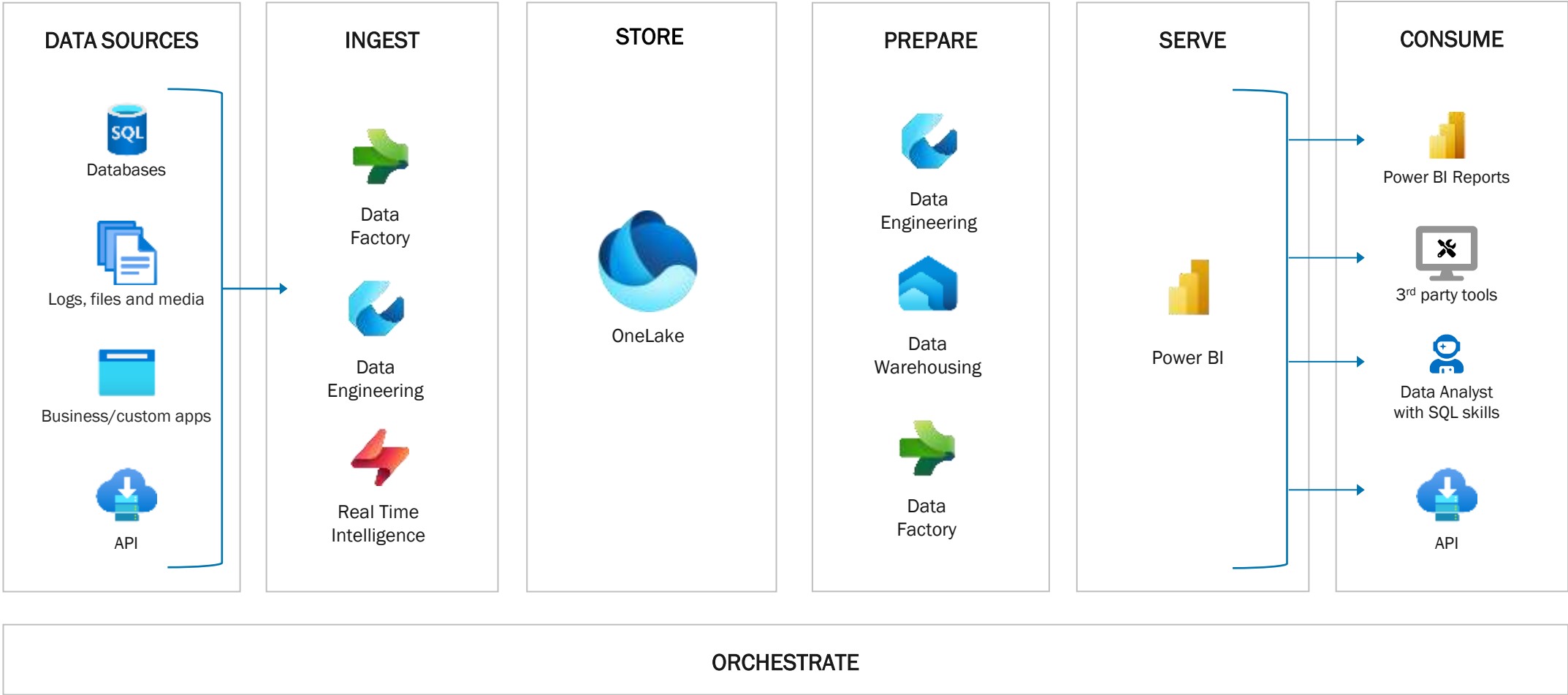
- Low-code/no-code
- Pro developer / engineers

- 30+ items

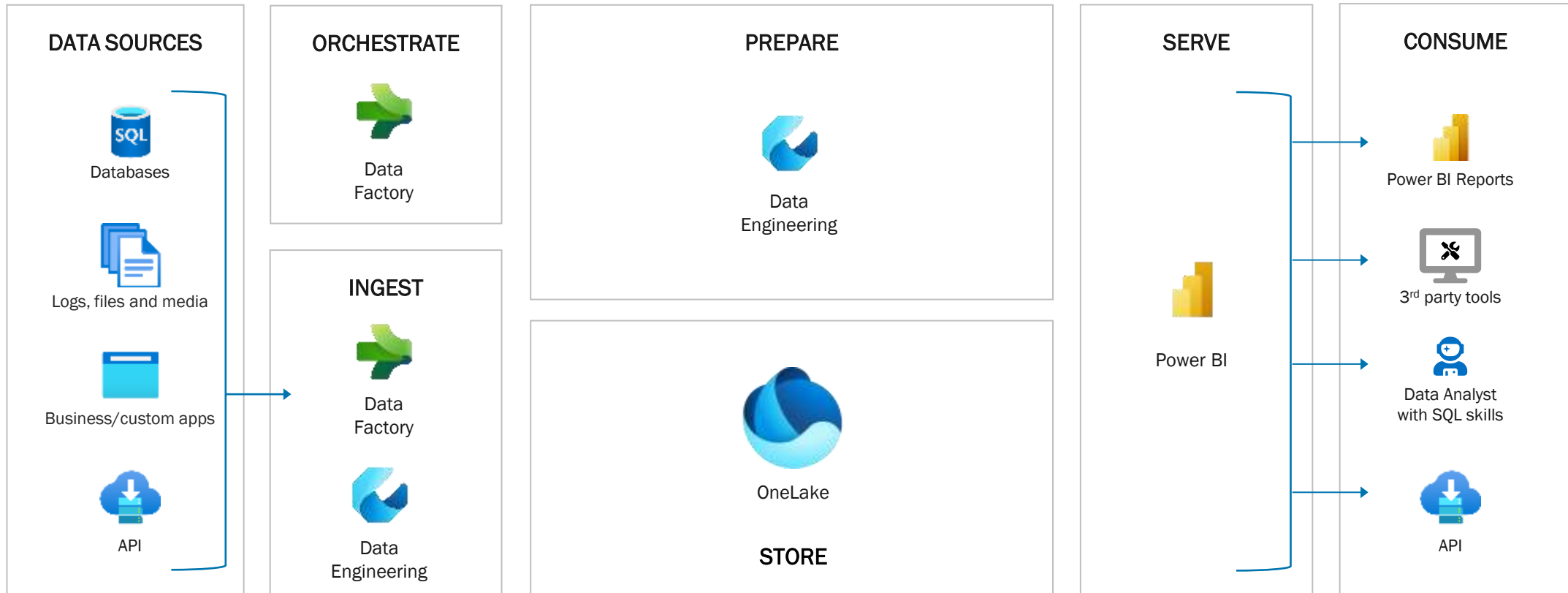
OneLake & Delta format

- Separation of storage and compute

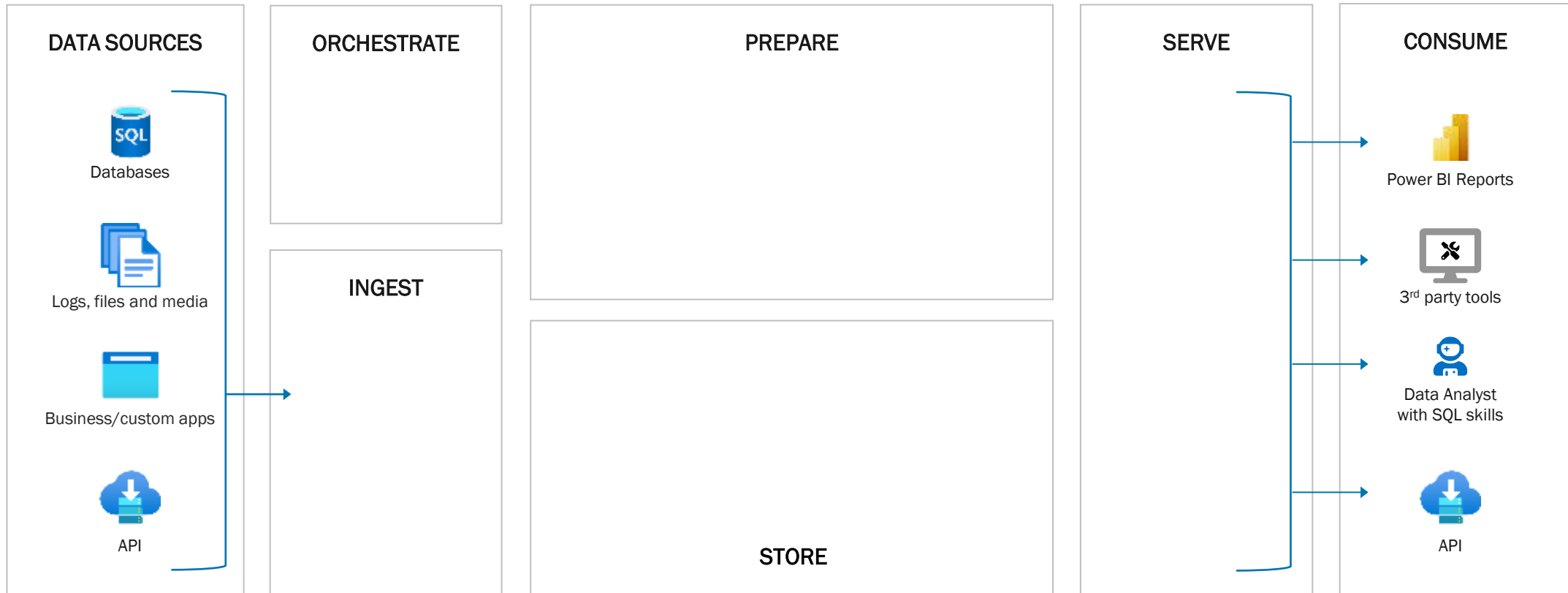
Architecture view of Fabric workloads



Lakehouse view of Fabric workloads



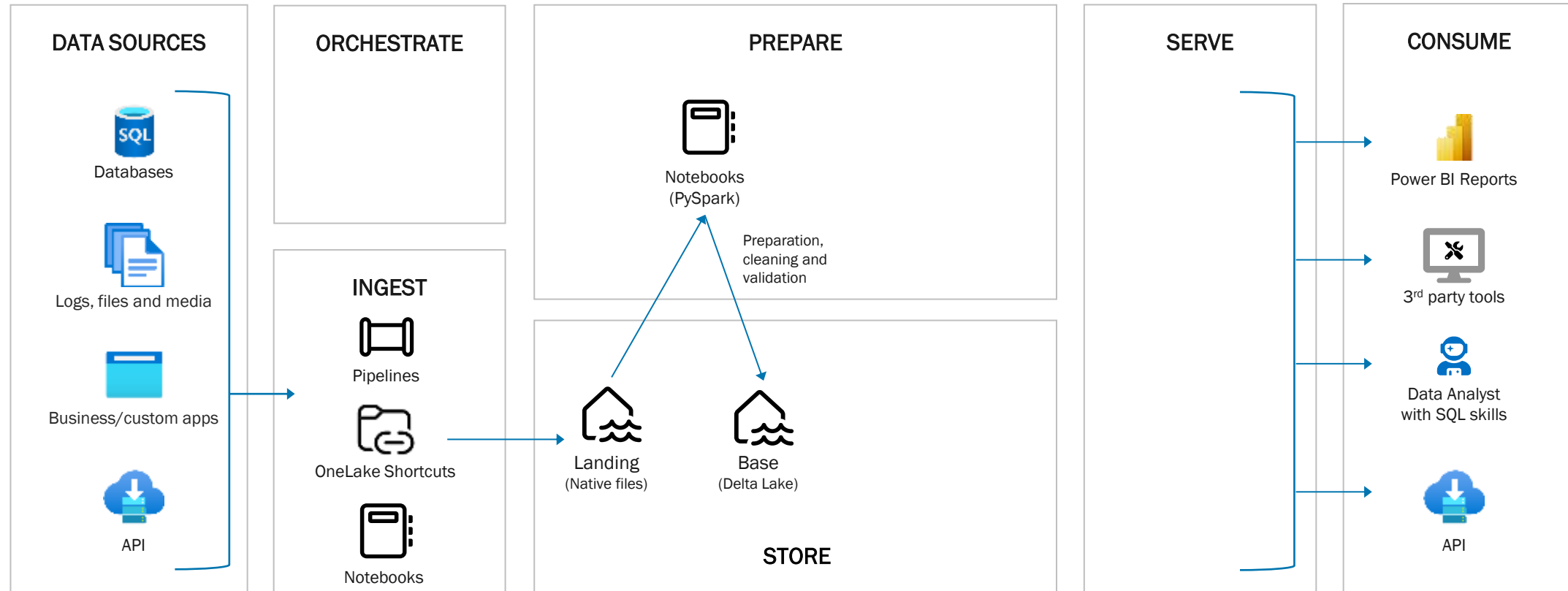
Dataflow in a Data Lakehouse in Fabric



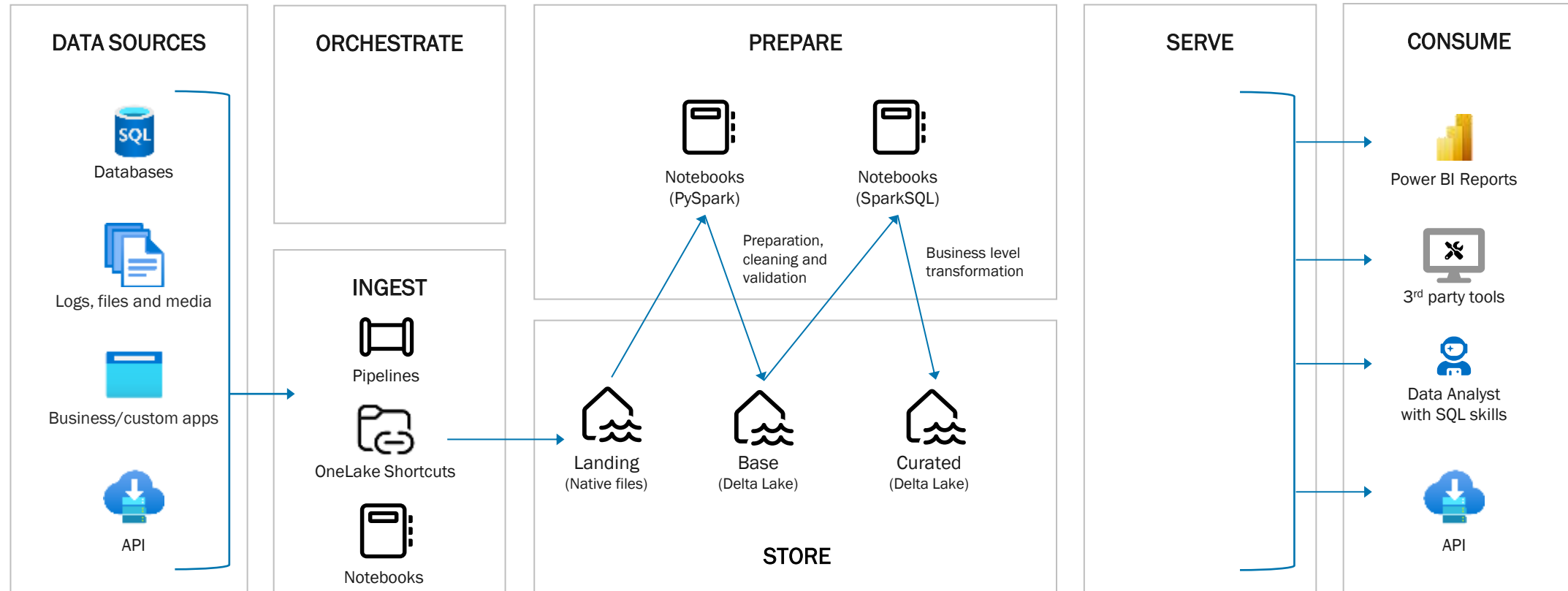
Dataflow in a Data Lakehouse in Fabric



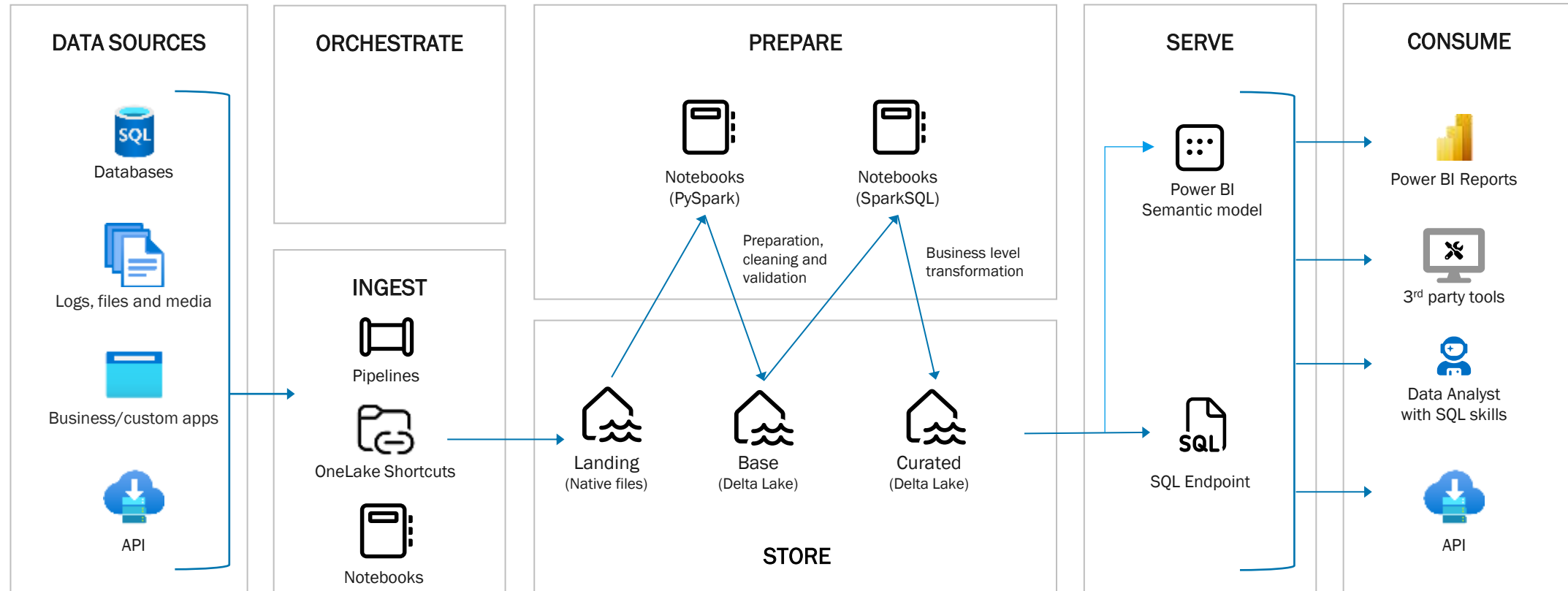
Dataflow in a Data Lakehouse in Fabric



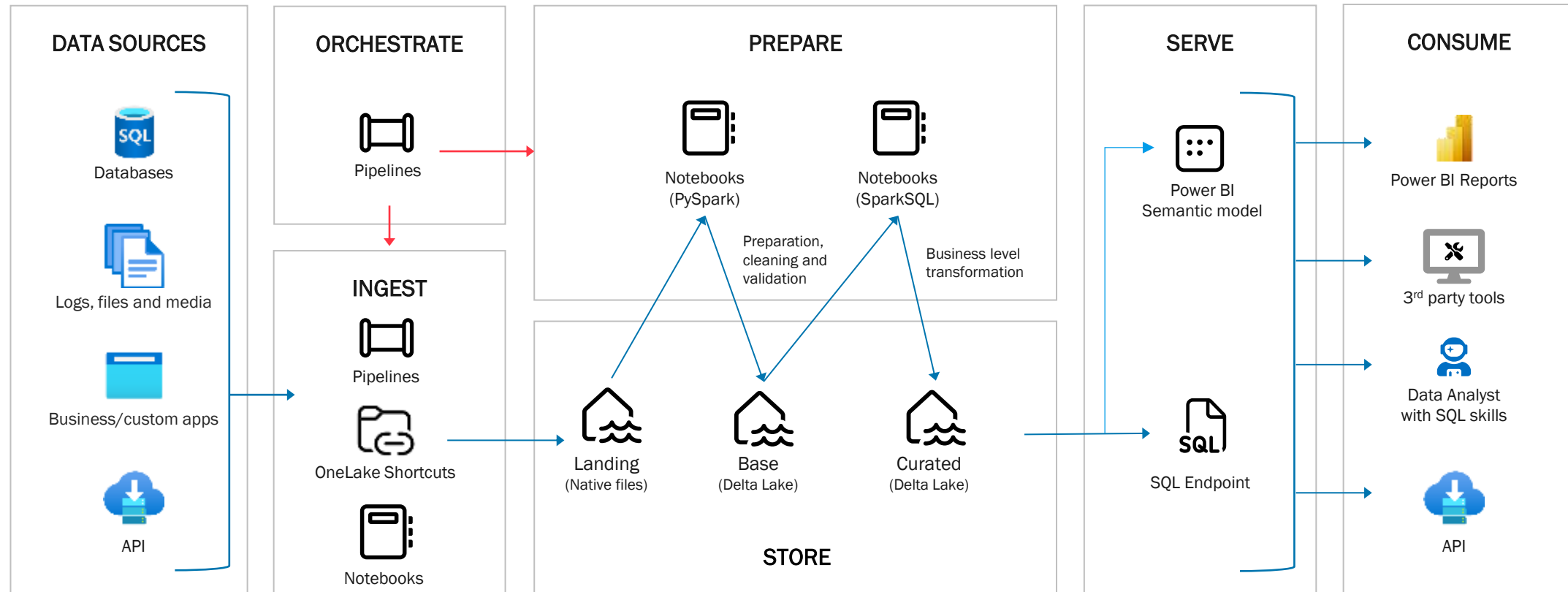
Dataflow in a Data Lakehouse in Fabric



Dataflow in a Data Lakehouse in Fabric



Dataflow in a Data Lakehouse in Fabric

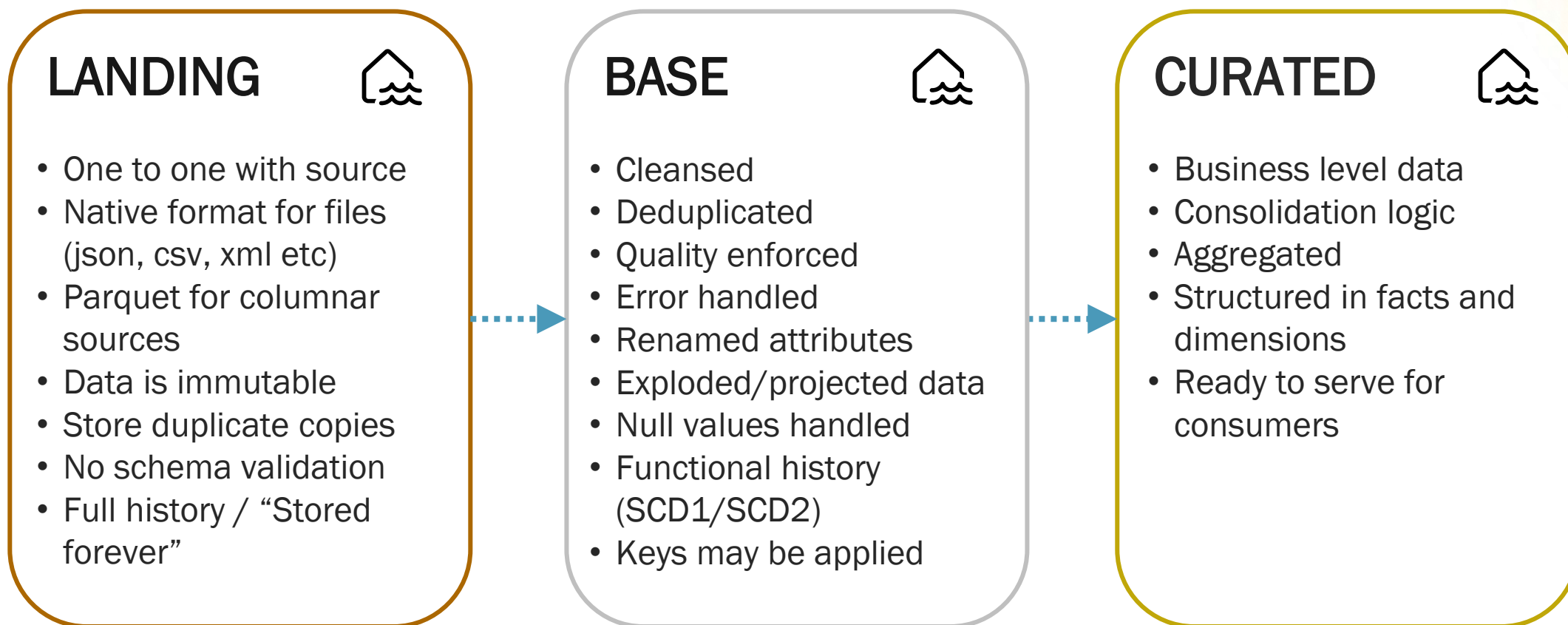


AquaShack: Data Lakehouse accelerator



- Pico-example of a meta-data driven lakehouse for Microsoft. Little brother to AquaVilla
- Data Lakehouse with three layers, where data is moved between.
 - Landing: Holds the data in original format if possible. Relation sources are stored in parquet.
 - Base: Aligned clean data, all stored in Delta tables.
 - Curated: Data for serving, business logic is applied, star schemas are defined etc.
- The notebooks builds upon functions defined in **AquaShack_functions**.
- We are reading metadata from JSON, it could have been from a database or YAML file as well. The important part is it ends in a common structure.

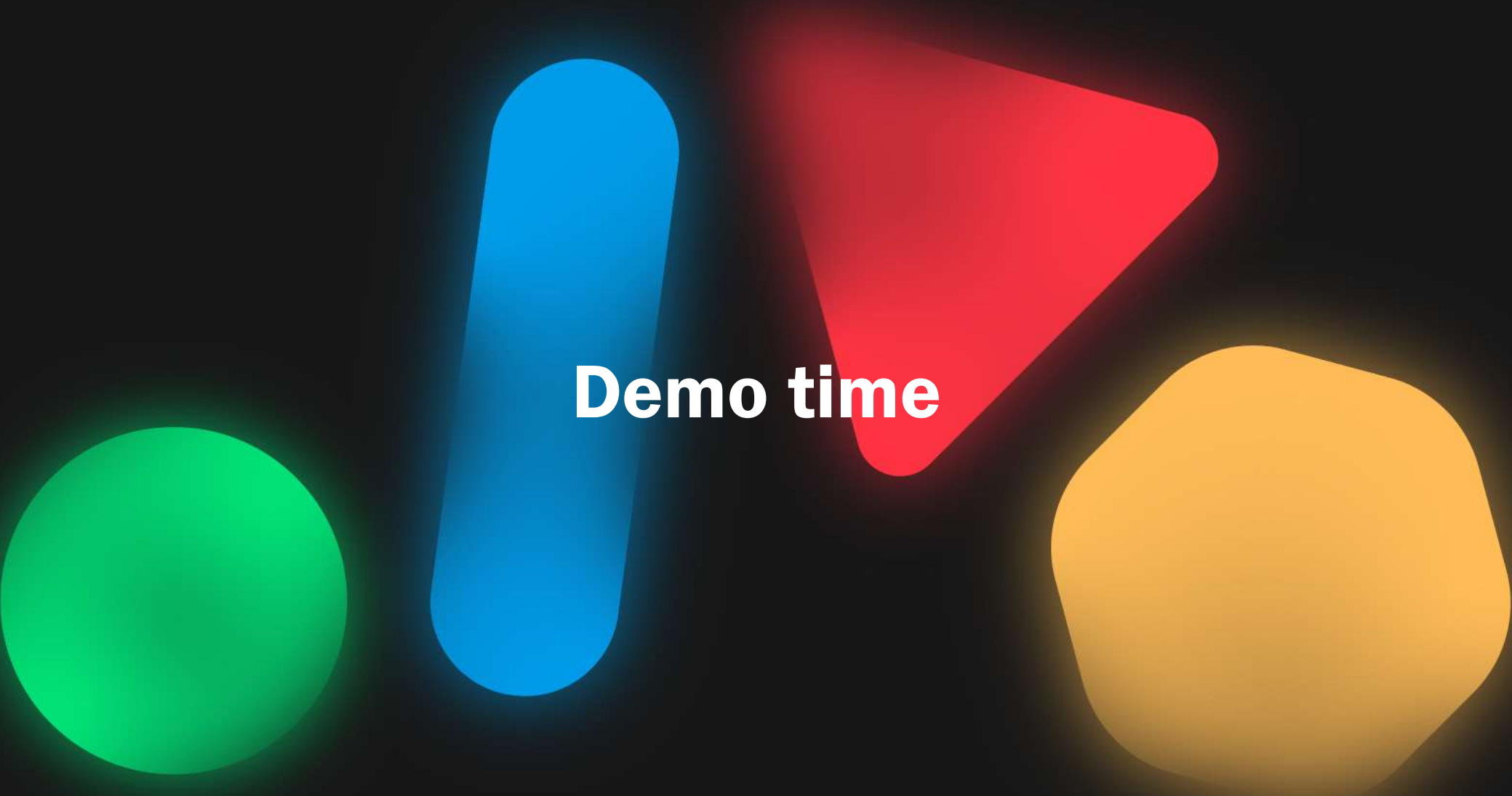
Layers in the Data Lakehouse



Metadata driven from Landing to Base

- The **AquaShack_Functions** notebook contains general functions used for operations like reading DataFrames, writing DataFrames, and utilities for manipulating DataFrames. To use the functions, include the notebook in your Spark session with the %run command. Once included, you can reference any function within the notebook, such as read_meta_from_json, write_to_delta_overwrite, and others.
- Data movement between the Landing and Base layers is managed by the LandingToBase notebook. This notebook reads metadata from a designated json-file and processes the source path (folder) specified within it.
- Metadata is maintained in the meta_data.json file. This metadata defines the rules for reading data from the Landing layer and writing it to the Base layer.

```
[
  {
    "source": "Files/External/Sample data/Sales/Customers/Customers.csv",
    "format": "csv",
    "destination": "sales_Customers",
    "projected_columns": ["CustomerID", "FirstName", "LastName"]
  },
  {
    "source": "Files/External/Sample data/Sales/Products/Products.parquet",
    "format": "parquet",
    "destination": "sales_Products",
    "projected_columns": []
  },
  {
    "source": "Files/External/Sample data/Sales/Transactions/",
    "format": "csv",
    "destination": "sales_Transactions",
    "projected_columns": []
  }
]
```

Demo time

twoday

Base to Curated

- The Curated layer stores data in a dimensional model with entities like dimensions, facts, and bridge tables.
- Entity-specific Spark notebooks handle data creation, named as:
 - Load_Dimension_[dimension_name]
 - Load_Fact_[fact_name]
- Each notebook follows a standard structure:
 - Parameters & Settings.
 - Business Logic: Written in SparkSQL.
 - Generic Load Execution: Functions like load_dimension or load_fact.
- All notebooks are tied to the Base Lakehouse item for centralized management.

Defining settings

Some settings run through the notebook and easy to set once before the job starts.

```
1 destination_lakehouse: str = "Curated"
2
3 dimension_name: str = "Product"
```

Defining business requirements

This could also be pure PySpark, it depends on the skill set of the BI Developer.

```
1 -- Step 1
2 CREATE OR REPLACE TEMPORARY VIEW dim_df AS
3
4 SELECT
5     -- Keys
6     Id as Dim_ProductId,
7
8     -- Subcategory
9     ProductId as ProductSubId,
10
11     -- Attributes
12     Manufacturer,
13     ProductName,
14     Price
15
16 FROM
17     Base.sales_products
```

Writing the dimension

Fetch the DataFrame and pass it to our function for writing dimensions.

```
1 dim_df = spark.table('dim_df')
2
3 load_dimension(df = dim_df, database = destination_lakehouse, table = dimension_name)
```

Overwriting Curated-dim_sales_products



Demo time

twoday

Getting started with AquaShack



- **GitHub:** <https://github.com/ChristianHenrikReich/AquaShack>
- The Setup.ipynb notebook holds all to get started and sets up the AquaShack lakehouse example. The only required action is to run this code in a notebook within the target Fabric Workspace.



AquaVilla: Fabric Lakehouse Accelerator

Infrastructure

Publisher app and scripts

- Azure infrastructure
- Fabric env. setup



Solution

Metadata-driven ingest, transform and orchestration

- Data Pipelines
- Onelake as storage
- Notebooks
- Metadata-driven development



CI/CD

Pipelines and scripts

- Build and release
- Handling features



Documentation, learning and guidelines

Selected features of AquaVilla

- **Automated setup:** Full setup of infrastructure, workspaces and items
- **Metadata:** Integration, processing and orchestration with metadata
- **Ingestion:** Batch ingest from multiple sources using delta and full.
- **Common transformations:** Deduplication, flattening of complex data structures, data conversion, translations and more.
- **Update strategies:** Append, overwrite, upsert type 1 & 2 etc.
- **Data warehousing:** Functions specifically for dimensional modelling
- **Samples:** Sample implementations for all functionality and layers
- **Orchestration:** Through Pipelines and Notebooks
- **Deployment:** Full enterprise CI/CD using Azure DevOps Pipelines



Learn more about AquaVilla: <https://www.youtube.com/watch?v=OfvLLYW5yW4>

twoday



twoday

Sponsors



 **Goldredgate**



SILVER
CatMan
Solution



BIWISSE
BUSINESS INTELLIGENCE



twoday