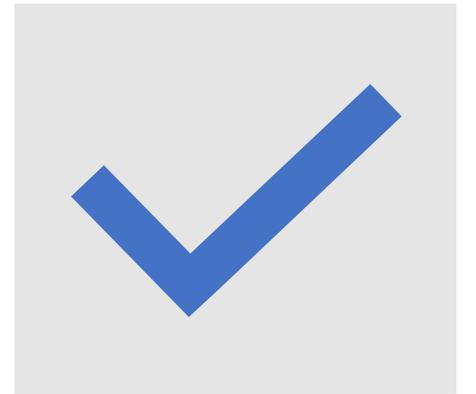


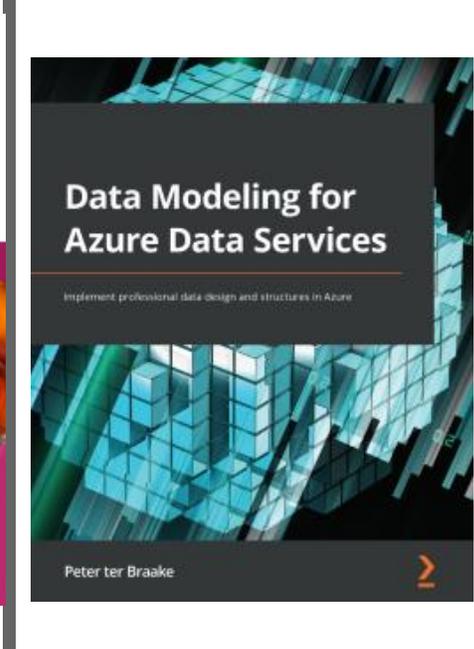
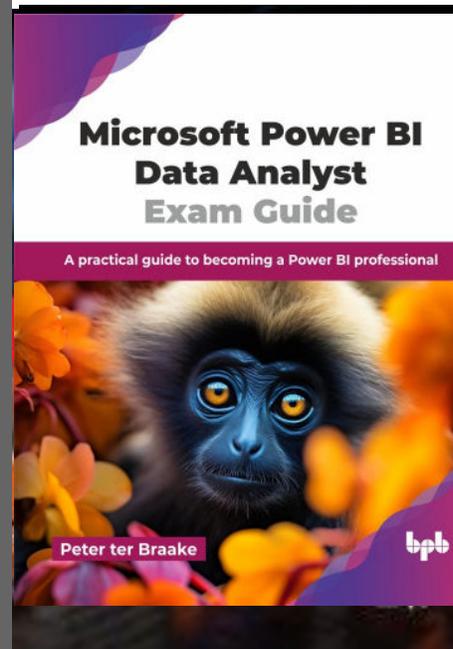
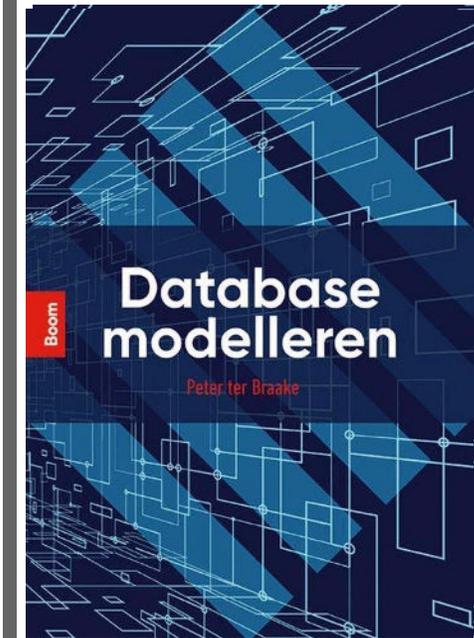
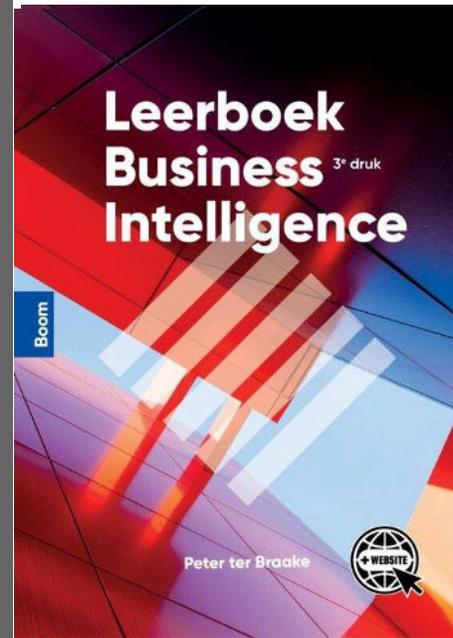
# Microsoft Fabric



Peter ter Braake

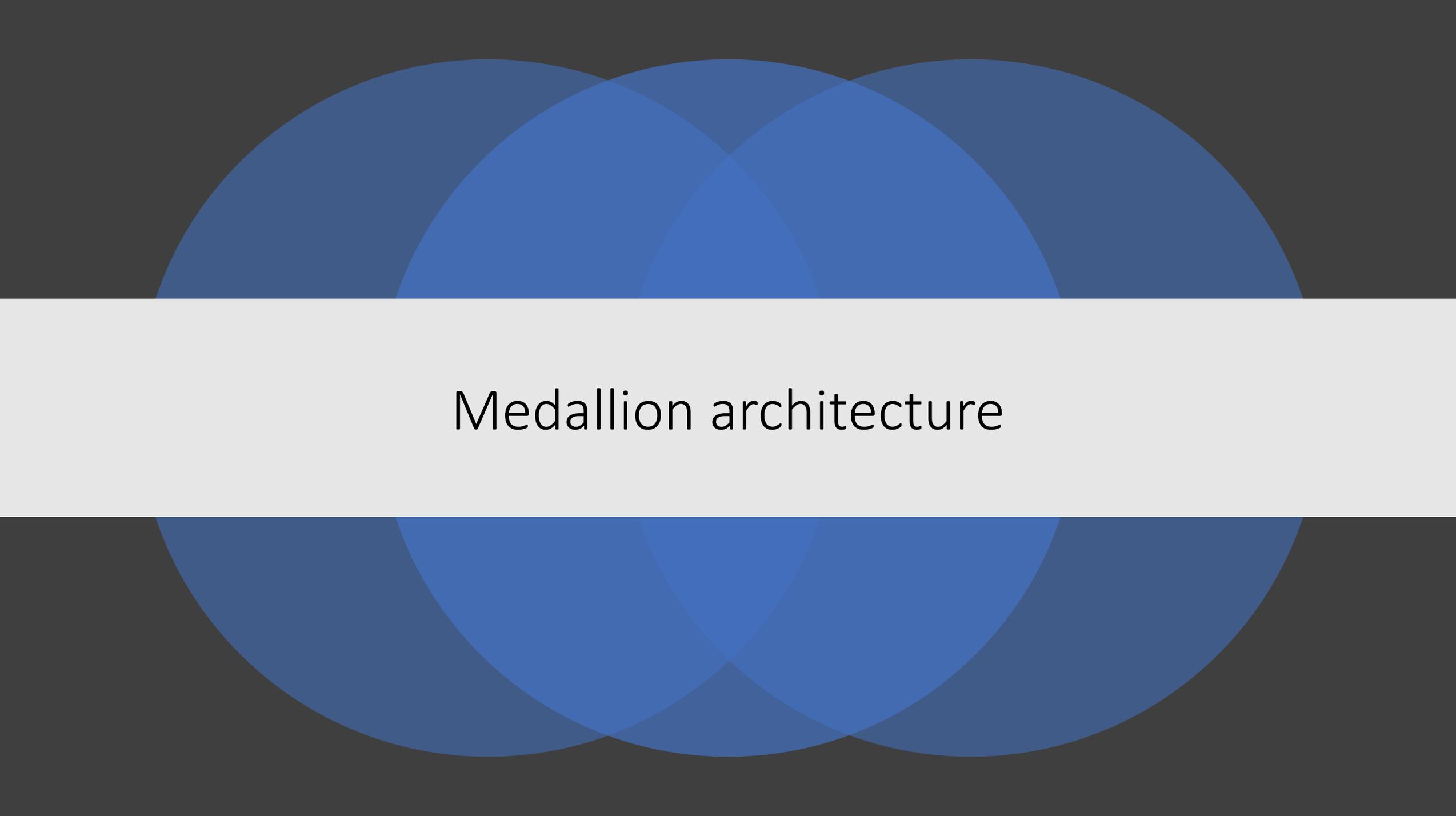
# Introducing myself

- Peter ter Braake
  - [trainsql@live.nl](mailto:trainsql@live.nl)
  - Independent contractor
  - MCT since 2002
- Microsoft Data Platform
  - SQL Server
    - Incl. SSRS
  - Business Intelligence
  - Power BI
- Author



# Agenda

- Medallion architecture
- Lakehouse
- Pipeline
- (py)Spark / Delta
- Dataflow Gen2
- Warehouse
- Semantic model
- Power BI
- Security
- Eventhouse
- Machine Learning

The image features a dark gray background with a central white horizontal band. Above and below this band are three overlapping circles in shades of blue, creating a decorative, layered effect. The text 'Medallion architecture' is centered within the white band.

# Medallion architecture

# Operational versus Analytical

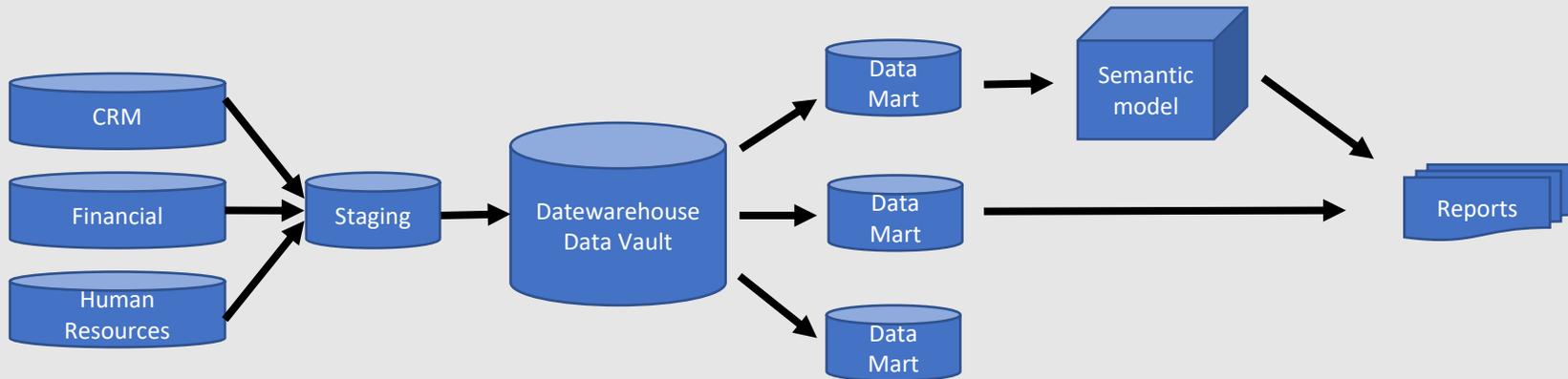


OLTP	ODS	BI	Analytics
Mostly transactional	Can be transactional	Nontransactional	Nontransactional
Subsecond response	Subsecond to seconds	Seconds to minutes	Minutes to hours
Customer experience	Customer experience or Business internal	Business internal	Business internal
Large update volume	Low update volume	No direct updates	No direct updates
Online updates	Batch to streaming feeds from OLTP	Batch to streaming feeds from OLTP/ODS	Batch/aggregates from BI
No historical data	Some historical data	Historical data	Historical and big data
High concurrency	Low concurrency if internal, high otherwise	Low to high concurrency	Low concurrency
Scales linearly	Near linear scale	Less linear in scale	Complex queries, nonlinear scale
Normalized data model	Normalized data model	Dimension data model	Columnar store
Custom applications or third-party solutions	Custom apps / third party	BI, OLAP, ROLAP tools—reporting and dashboards	Analytical tools
Keyed updates/queries	Keyed queries	Ad hoc and scheduled queries and large extracts	Ad hoc queries; Analytics in database
Mostly SMP; MPP for web-scale	Mostly MPP	Mostly MPP	Mostly MPP

Performing analytics on operational databases might suffer from:

- Bad performance
  - Of LOB application
  - Of analytics itself
- Complexity of operational databases
  - Error prone
  - Makes BI developers in-productive
- Lack of consistency / 360 degree view
- Bad Data Quality
- Lack of historical data

# Where we come from



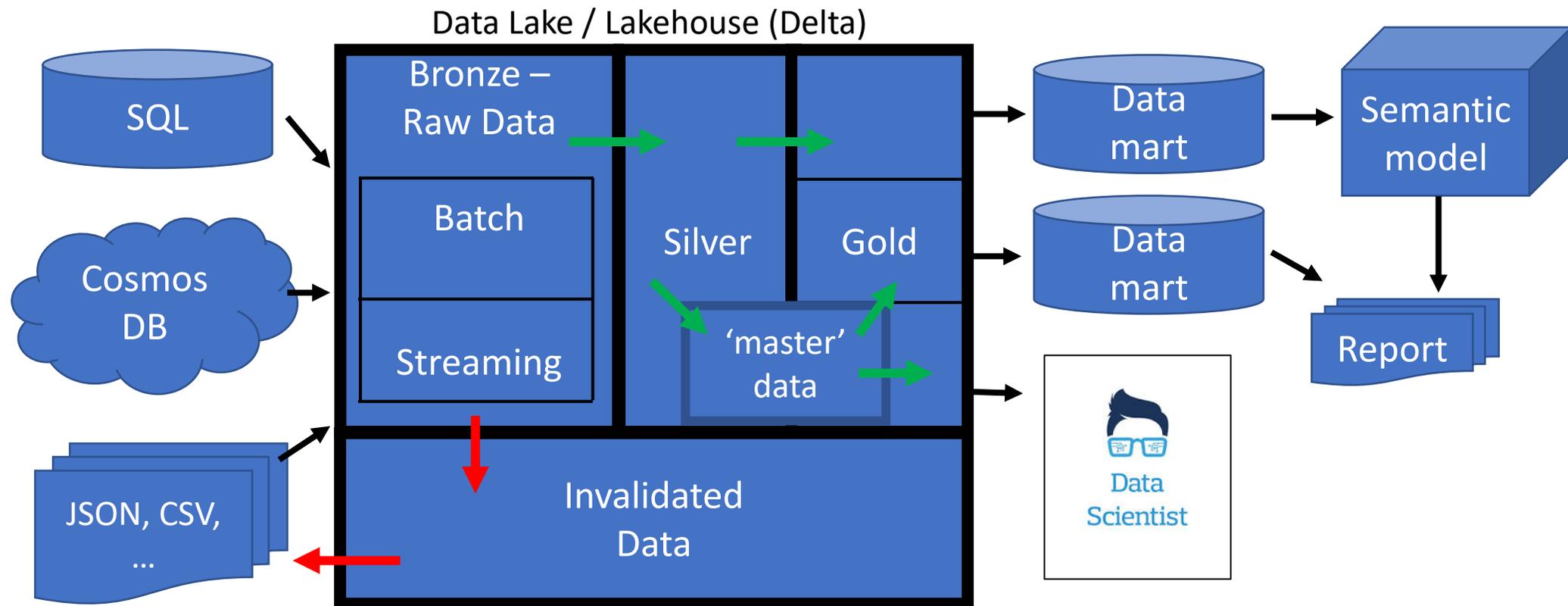
Source → ETL → BISM → Report

Power BI:

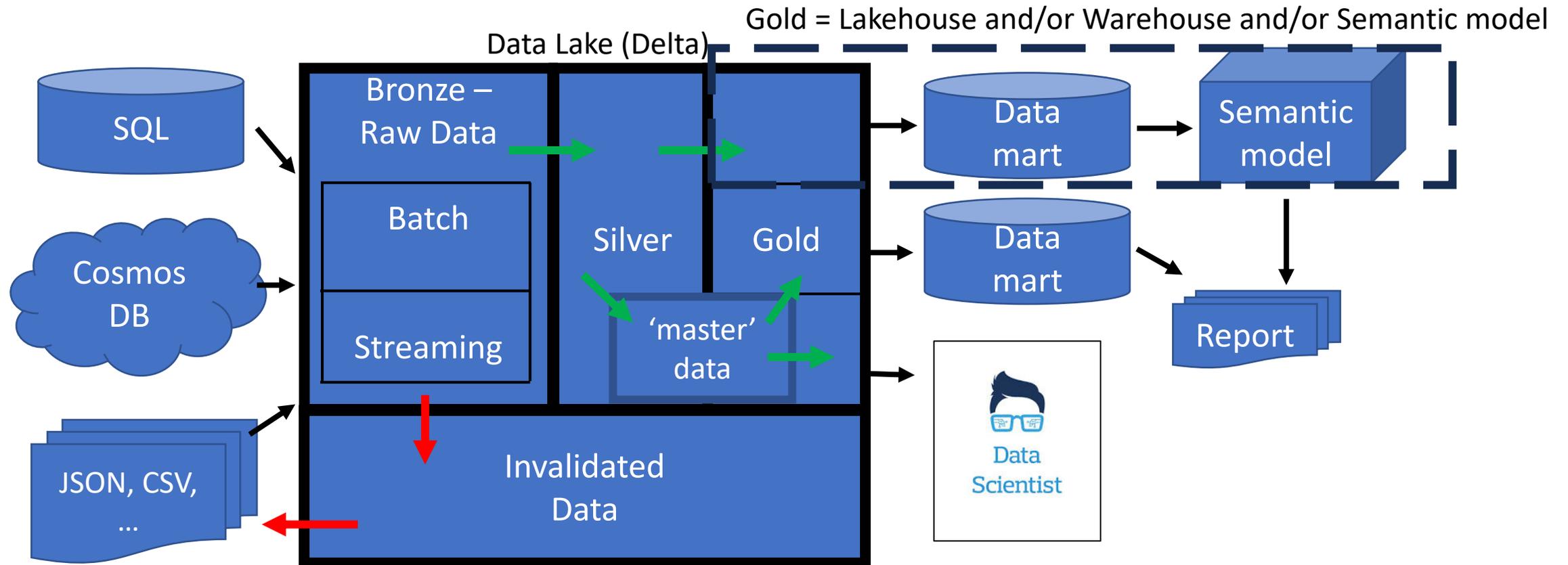
Source → Get Data → Power Query → DAX → Report

- DWH architecture:
  - Kimball
  - Inmon
  - Data Vault
- Same architecture, same best practices
- Implement solutions as far to the left as possible

# Architecture Overview



# Architecture Overview

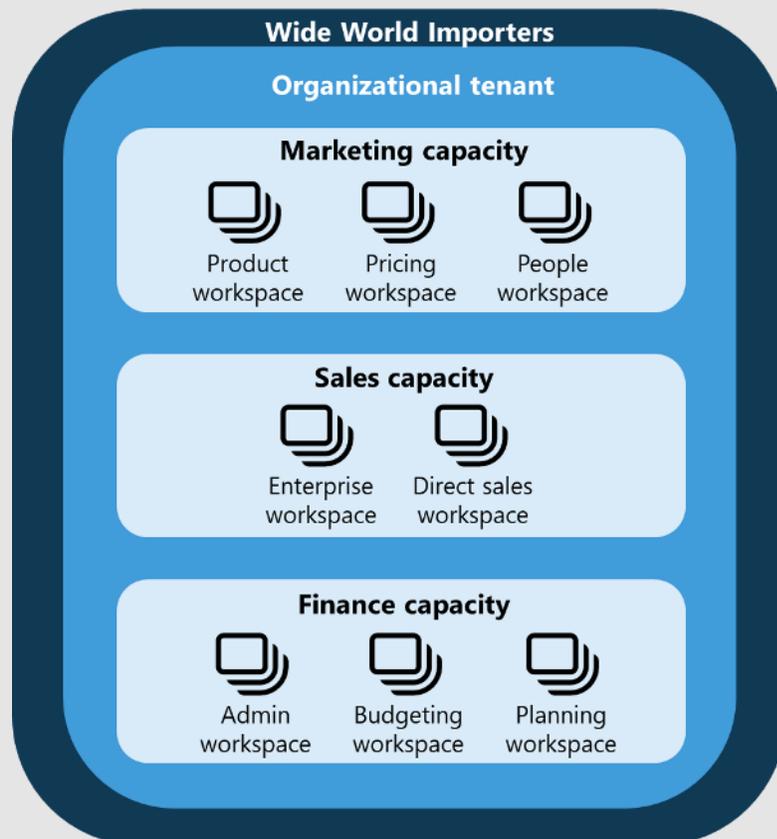


Consider virtual layers (data virtualization) whenever possible

The image features a dark gray background with a central white horizontal band. Above and below this band are three overlapping circles in shades of blue, creating a decorative, wave-like pattern. The text "Get started: Fabric Lakehouse" is centered within the white band.

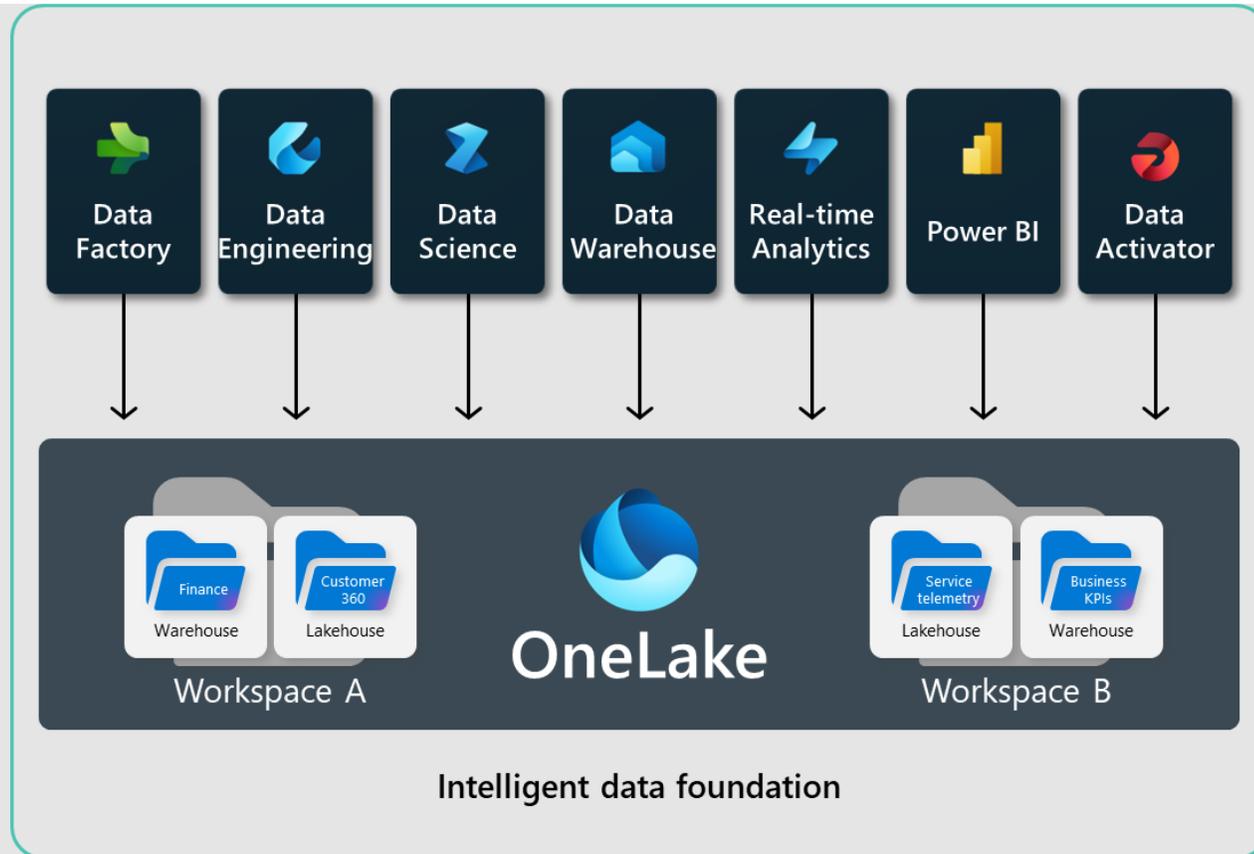
Get started: Fabric Lakehouse

# Fabric concepts



- **Tenant**
  - Dedicated space to create, store, and manage Fabric items
  - Aligned with Entra
  - Maps to root of OneLake
  - Top level of hierarchy
- **Capacity**
  - Dedicated set of resources available for use
  - One or more per tenant
  - Offered through Fabric SKU (and Trials)
- **Domain**
  - Logical grouping of workspaces
  - Used to organize items in a way that makes sense to organization
  - Can contain subdomains + contains one or more workspaces
  - Roles: Domain admin + Domain contributor
- **Workspace**
  - Collection of items
    - Access control
  - Can contain folders
- **Items**
  - Building blocks of Fabric platform
  - Notebooks, Semantic models, Reports, ...

# Fabric architecture



- Single SaaS lake
  - Provisioned automatically with tenant
  - Build on ADLS Gen2
- All data
  - Stored in OneLake
  - Organized in hierarchical namespace
- Data is automatically indexed for
  - Discovery
  - MIP labels
    - Microsoft Information Protection Sensitivity labels
  - Lineage
  - PII scans
    - personally identifiable information
  - Sharing
  - Governance and compliance

# Fabric workspace

Consider for example:

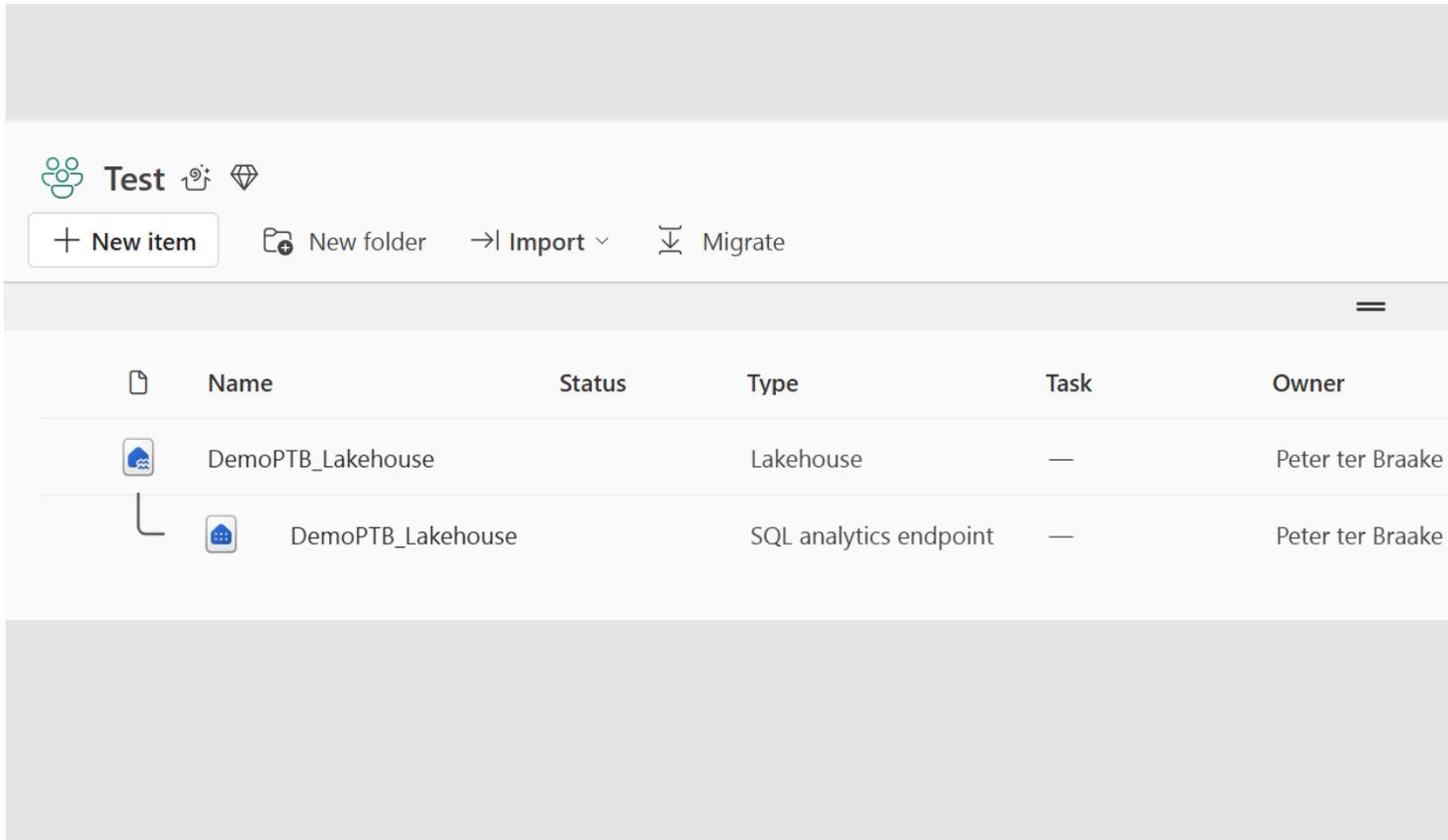
Workspace	Purpose	Artifacts
1	Storage	Bronze, Silver: Lakehouse
2	ETL	Pipelines, notebooks, variable library, dataflows
3	Storage	Gold: Lakehouse, Warehouse, Semantic model
4	Reporting	Reports, Dashboards
5	Monitor, logging	SQL Database

- Grouping of artifacts
- Can be monitored and shared as a whole
- Can be connected to Git repository
- Can be part of deployment pipeline
- Can be part of domain
- Has folders

# Demo

Create Lakehouse

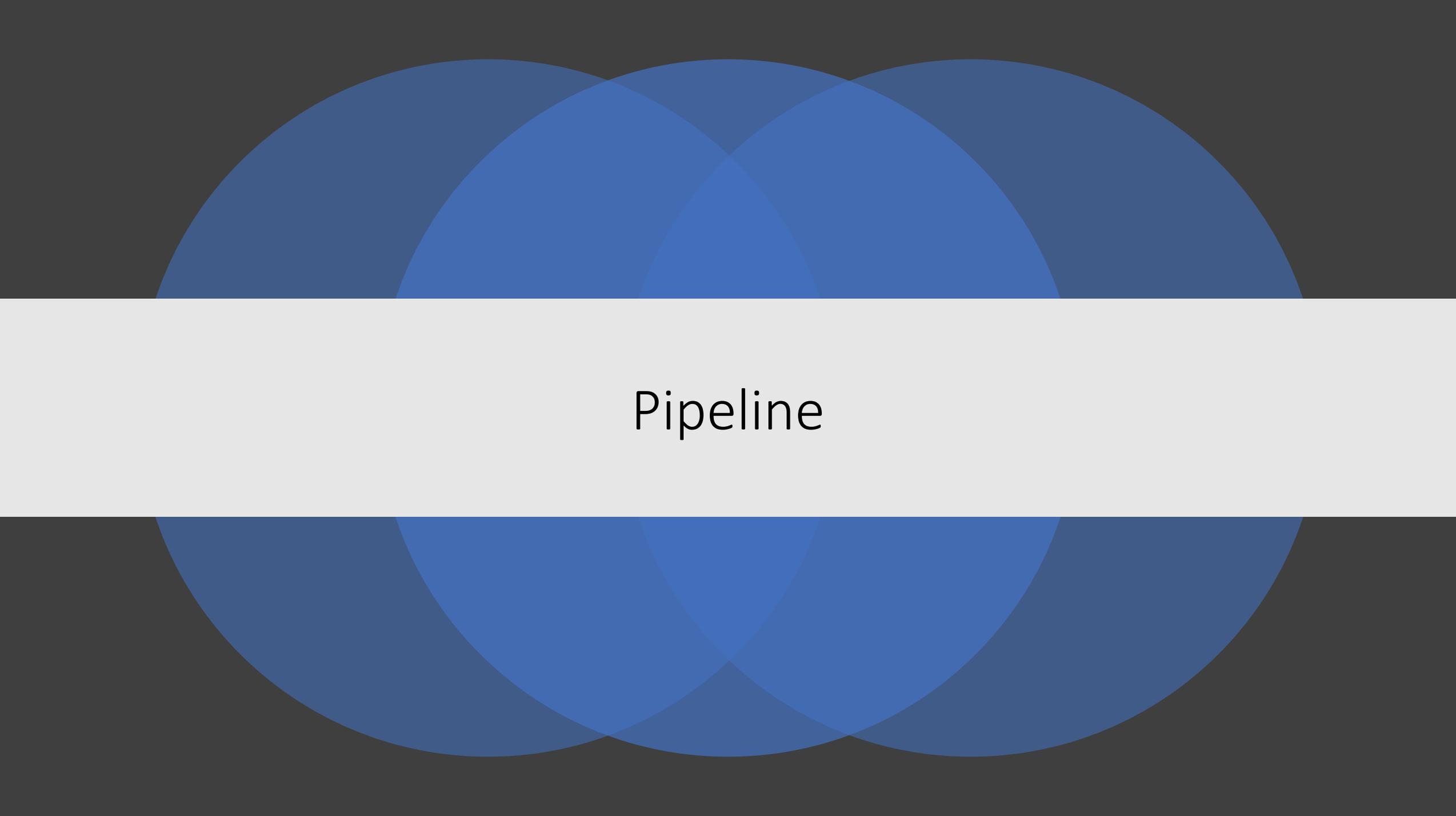
# Create Lakehouse



The screenshot shows the Azure Data Lake Storage Explorer interface. At the top, there's a header with the workspace name 'Test' and several icons. Below the header is a toolbar with buttons for '+ New item', 'New folder', 'Import', and 'Migrate'. The main area displays a table of artifacts:

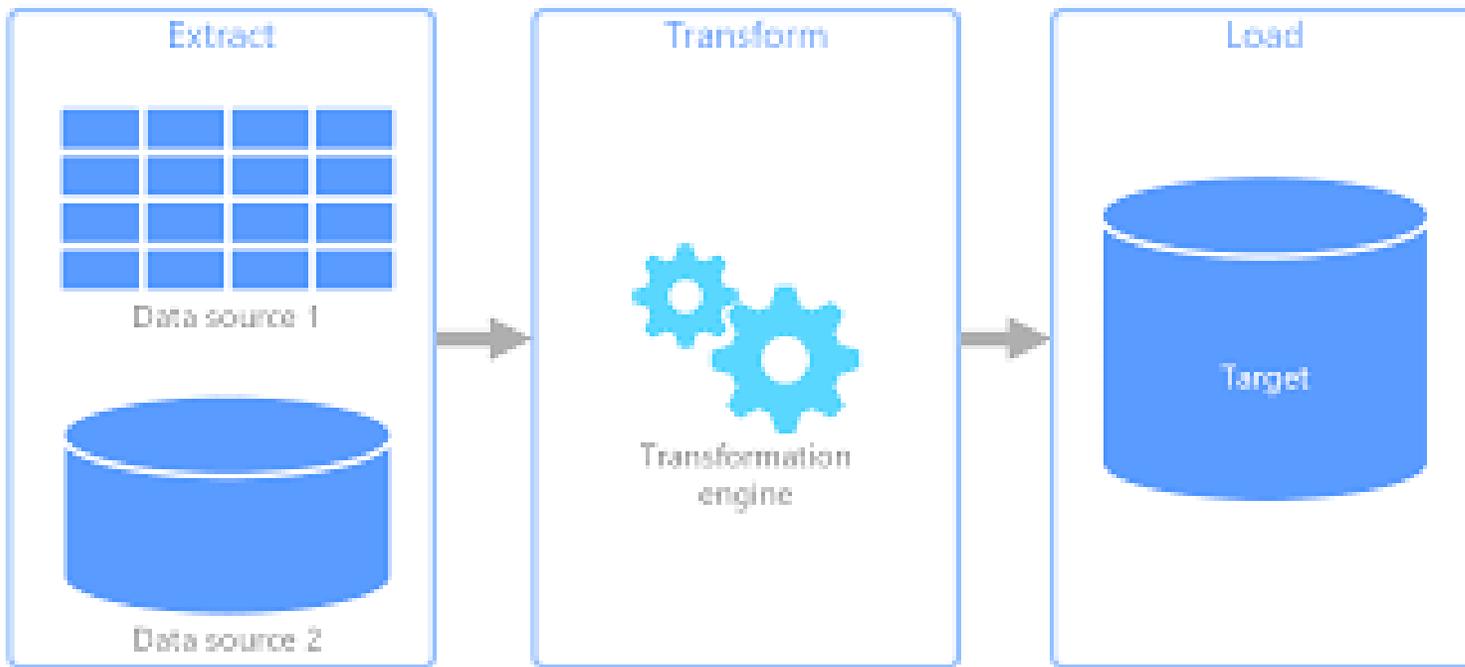
	Name	Status	Type	Task	Owner
	DemoPTB_Lakehouse		Lakehouse	—	Peter ter Braake
	DemoPTB_Lakehouse		SQL analytics endpoint	—	Peter ter Braake

- Create Lakehouse creates 3 artifacts
  - Lakehouse
    - Data in files
    - Data in tables
  - Semantic model
    - Formerly Power BI dataset
    - Based on Lakehouse tables
  - SQL Analytics endpoint
    - Use T-SQL to work with data
    - Use tools like
      - Azure Data Studio
      - SSMS

The image features a dark gray background with a central white horizontal band. Behind the band, there are three overlapping circles in shades of blue, arranged horizontally. The word "Pipeline" is centered within the white band.

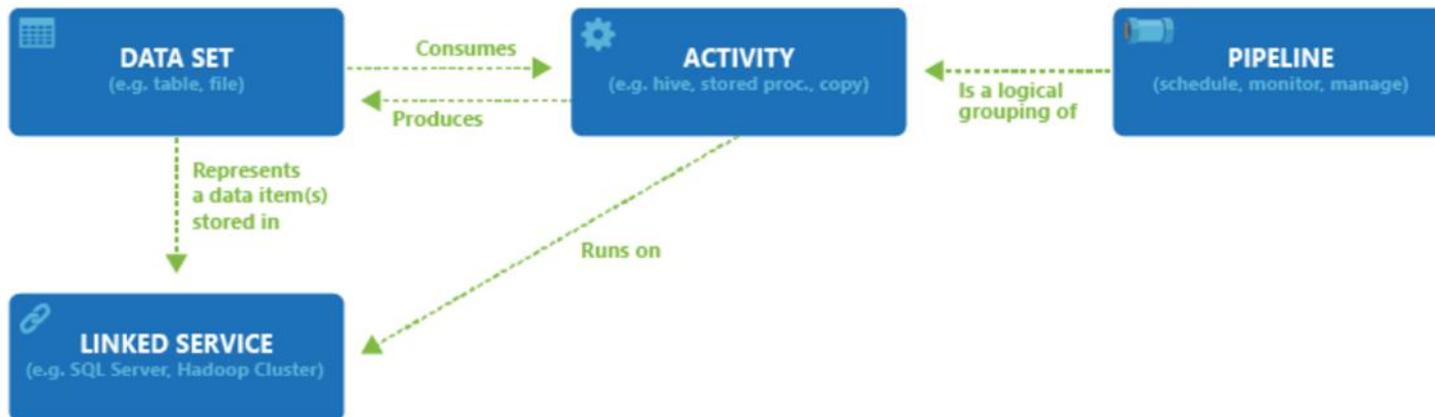
Pipeline

# Extract *Transform* Load



- Cleanse data
  - Change data type
  - Replace errors
  - Replace nulls
  - Add (calculated) columns
  - ...
- Transform
  - Combine tables
  - Combine sources
  - Split tables
- Apply business rules (when appropriate)

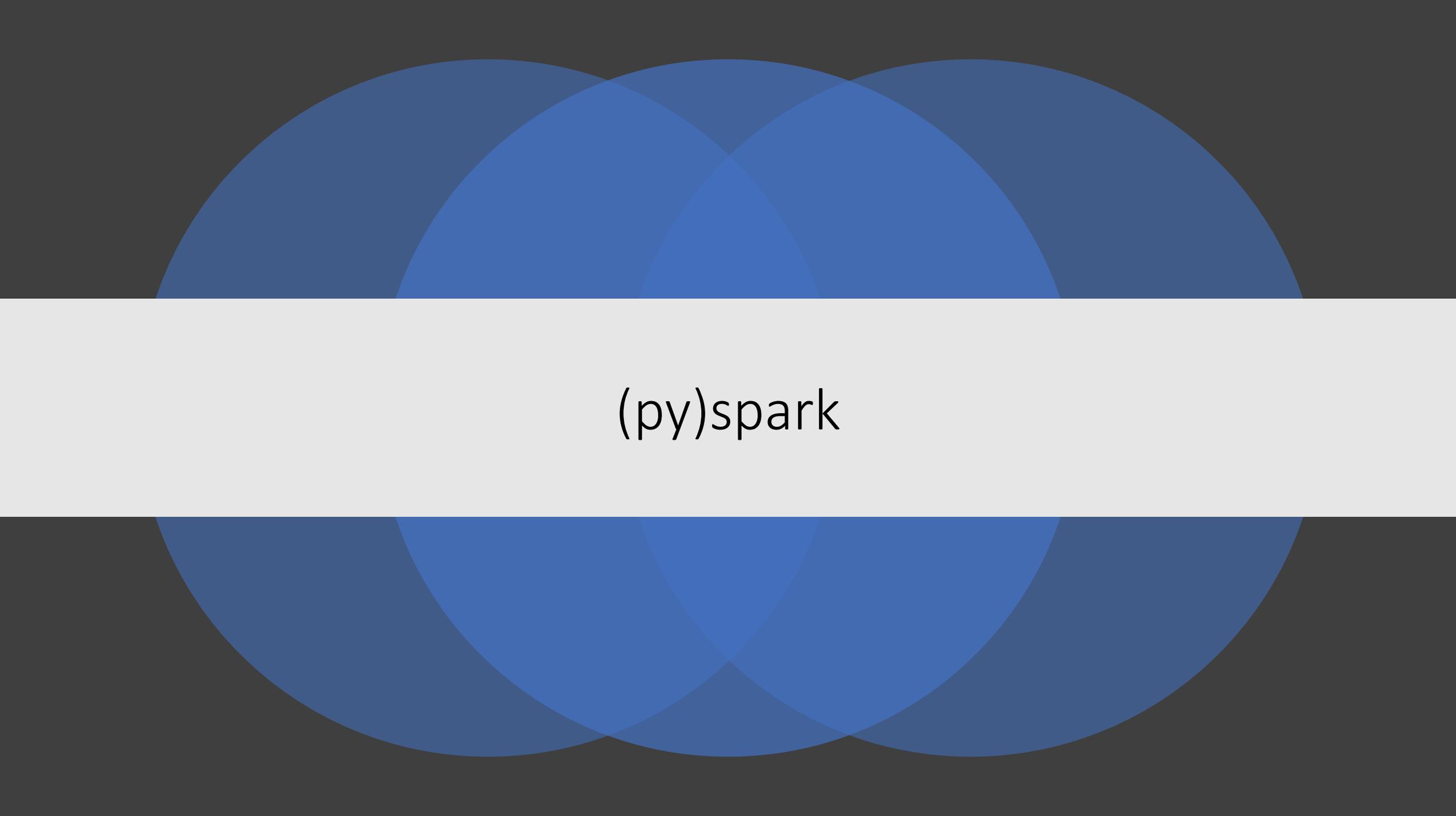
# Pipeline



- Encapsulate flow of activities orchestrated by integration runtime
- Activities can include:
  - Data movement /transformation
  - External processing activities
  - Control flow activities
    - Manage variables
    - Contains processing logic
- Linked service
  - Provides access to
    - Storage (Data stores)
    - Compute
- Dataset
  - Reference to data
    - Table, file, ...
  - Accessed through linked services
- Parameters

# Demo

Create pipeline

The image features a dark gray background with a central white horizontal band. Behind the band, three overlapping circles in shades of blue are visible, creating a decorative pattern. The text "(py)spark" is centered within the white band.

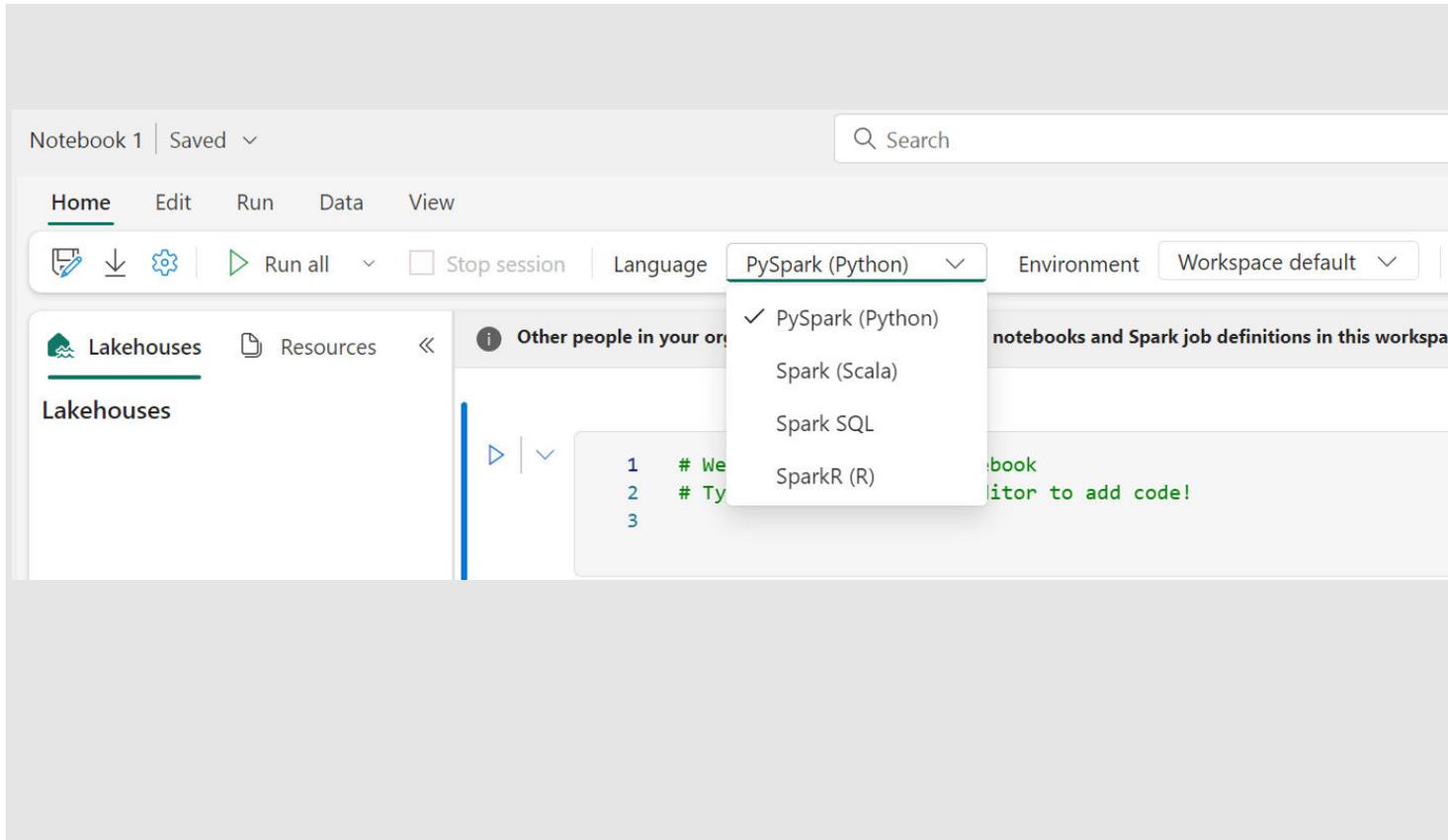
(py)spark

# Cluster



- Use more servers instead of bigger servers
- Head node / master node:
  - Where you connect to / what you see
  - Divides the work / assembles intermediate results
- Worker node:
  - Stores part of the data
  - Executes part of the query

# Spark Pools



- Apache open-source Spark technology
- Backed by (configurable) clusters
  - Highly scalable
- Notebook oriented
  - Multiple languages
    - Python
    - Scala
    - SQL
    - ...
  - Multiple libraries

# Demo

Create notebook

# spark.read

```
1 File = "dbfs:/FileStore/shared_uploads/trainsql@live.nl/Employee/Employee.csv"
2 Wildcard = "dbfs:/FileStore/shared_uploads/trainsql@live.nl/Employee/*.csv"
3 Folder = "dbfs:/FileStore/shared_uploads/trainsql@live.nl/Employee/"
4
5 # read single file
6 df1 = (spark.read
7     | .format("csv")
8     | .option("header", "true")
9     | .load(File)
10 )
11
12 # read multiple file2
13 df2 = (spark.read
14     | .format("csv")
15     | .option("header", "true")
16     | .load(Wildcard)
17 )
18
19 # read all files from folder
20 df3 = (spark.read
21     | .format("csv")
22     | .option("header", "true")
23     | .load(Folder)
24 )
```

- Read:
  - File
  - Multiple files using wildcards
  - All files from folder
- Use `.option("recursiveFileLookup", "true")`
- Infer schema or specify explicitly

# dataframe

```
1 df_employee_filtered = (df_employee
2 |   .filter(df_employee.Salary >= '3000')
3 |   .select("Name", "Salary")
4 | )
5
6 display(df_employee_filtered)
```

- Read data
  - `spark.read`
- Transform data
  - Select columns
  - Filter rows
  - Add calculated columns
  - Join data
  - Aggregate data
- Write data
  - `Dataframe.write`

# spark.write

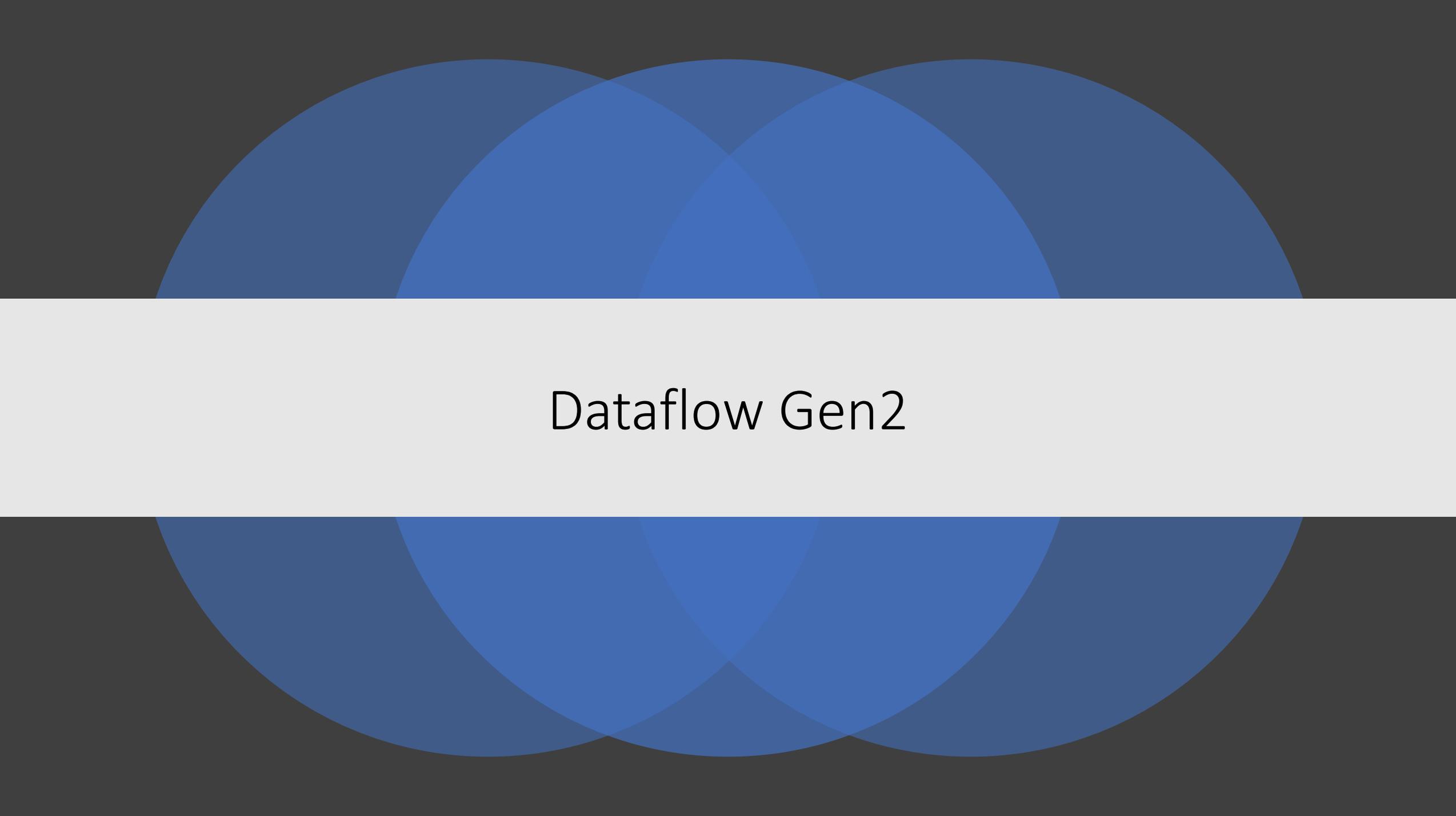
```
1  NewFile = 'Files/Silver/Northwind/Customers/'
2
3  (df_customer.write
4    .format('parquet')
5    .mode('overwrite')
6    .save(NewFile)
7
8  )
```

- Write:
  - Multiple file formats
    - Parquet
    - Delta
    - ...
- Write to
  - File section: save
  - Table section: saveAsTable
    - Allows use of SQL
- Creates multiple files
  - Depending on cluster
- 4 modes
  - Append
  - Overwrite
  - Ignore
  - Error (errorifexists)

# Delta Lake



- Open-source storage layer that adds relational database semantics to Spark
- Relational tables that support querying and data modification
- Support for ACID transactions
- Data versioning and time travel
- Support for batch and streaming data
- Standard formats and interoperability



# Dataflow Gen2

# Demo

Create Dataflow

# Dataflow Gen2

The screenshot displays the Microsoft Fabric Dataflow Gen2 interface. The top navigation bar includes tabs for Home, Transform, Add column, View, Help, and Schema tools. Below the navigation bar is a toolbar with various icons for actions like 'Get data', 'Enter data', 'Manage connections', 'Options', 'Manage parameters', 'Refresh', 'Advanced editor', 'Manage', 'Add data destination', 'Choose columns', 'Remove columns', 'Keep rows', 'Remove rows', 'Filter rows', 'Sort', 'Suggested transforms', 'Split column', 'Group by', 'Use first row as headers', 'Replace values', 'Combine', 'Map to entity', and 'Exp'. The main workspace shows a data pipeline with the following steps:

- Two source queries: 'Cash Transactions' (8 steps) and 'Digital Transactions' (6 steps).
- A transformation step: 'Append Transactions' (2 steps).
- A final transformation step: 'Join, Aggregate' (6 steps).

The 'Join, Aggregate' step is currently selected, and its properties are visible in the right-hand pane. The 'Applied steps' list includes 'Source', 'Expanded A...', and 'Grouped ro...'. The 'Data destination' is set to 'Warehouse'. Below the pipeline, a table displays the results of the query:

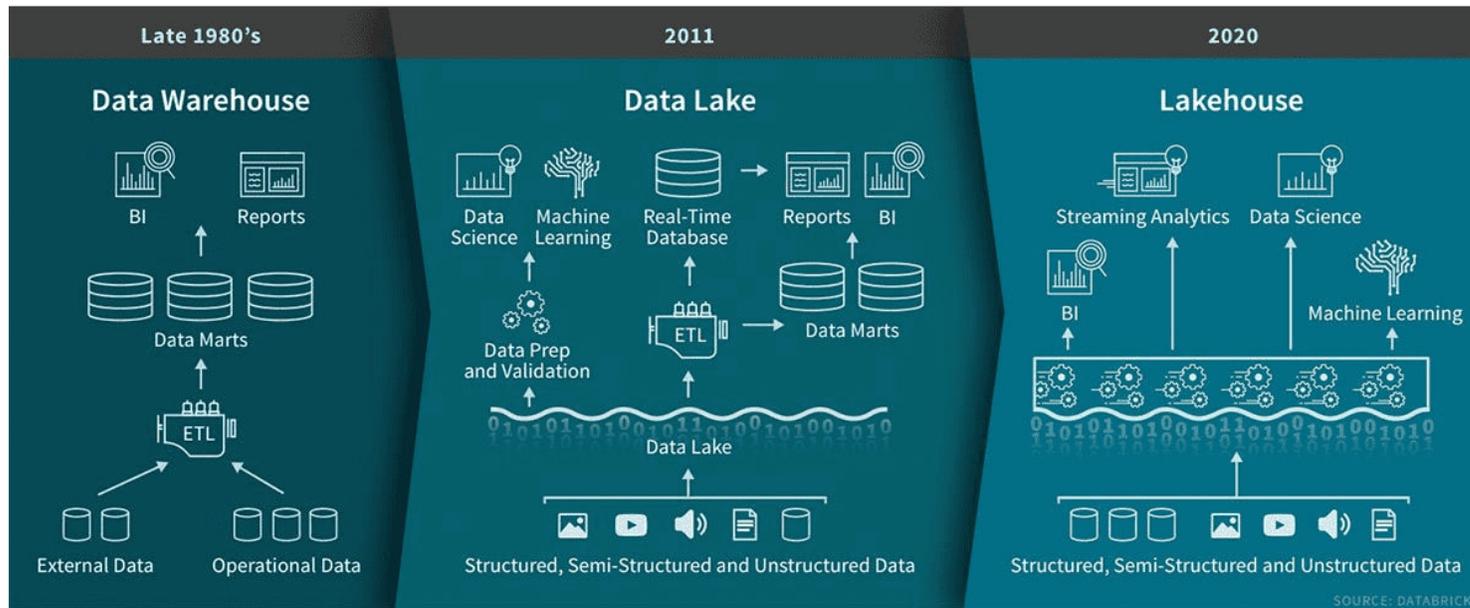
Name	Type	Key
FirstName	Text	
LastName	Text	
FullName	Text	
Total Transactions	Whole number	
Average Transaction a...	1,2 Decimal number	
Total Transaction amou...	1,2 Decimal number	

- Low-code graphical ETL environment
  - Extract data
    - Many source types
  - Transform data
    - 300+ transformations
  - Load data
    - Fabric destination
    - NO DESTINATION !
- Power Query online
  - Enhanced compute engine
- Run
  - Independently
    - Manual
    - Scheduled refresh
  - Pipeline activity

The image features a dark gray background with a central white horizontal band. Above and below this band are three overlapping circles in shades of blue, creating a decorative, abstract design. The text "Fabric Warehouse" is centered within the white band.

# Fabric Warehouse

# Warehouse versus Lakehouse



- Warehouse

- Structured data
- Multi-table transactions
- High performance
- Expansive security
  - Object/column/row-level, Dynamic Data Masking
- T-SQL
  - DDL, DML

- Lakehouse

- Semi-structured or unstructured data
- Scalable and cost-effective
- Supports Delta Lake features
- T-SQL security (row/table level)
- T-SQL, Spark (Scala, PySpark, Spark SQL, R)

# Lakehouse versus Warehouse

<https://learn.microsoft.com/en-us/fabric/fundamentals/decision-guide-lakehouse-warehouse>

No Code or Pro Code +

How do you want to develop?

Spark

Lakehouse

T-SQL

Warehouse

Warehousing Needs

Do you need multi-table transactions?

Yes

Warehouse

No

Lakehouse

Data Complexity

What type of data are you analyzing?

Don't know

Lakehouse

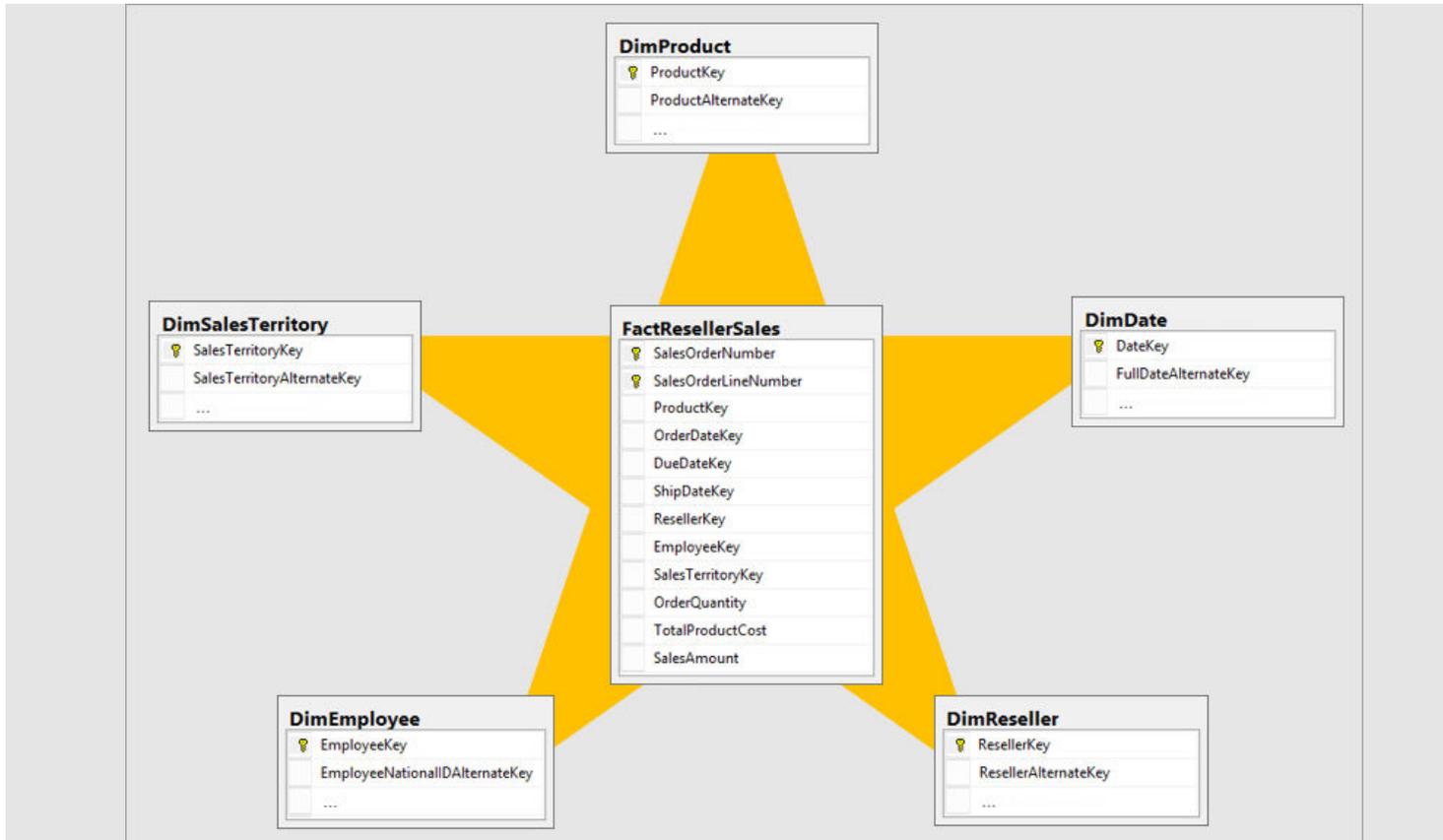
Unstructured and Structured

Lakehouse

Structured

Warehouse

# Why star schema?



- Warehouse used for data mart (NOT data warehouse)
- Intuitive / simple
  - Less error prone
  - Rapid report development
- Better performance
  - Less 'joins'
- Power BI
  - More consistent experience
  - Easier DAX
  - 'auto exist'
  - Effect cross filter both ways
  - Better performance

# Demo

Create warehouse

# Warehouse view versus table

VIEW	TABLE
A database object that allows generating a logical subset of data from one or more tables	A database object or an entity that stores the data of a database
A virtual table	An actual table
View depends on the table	Table is an independent data object

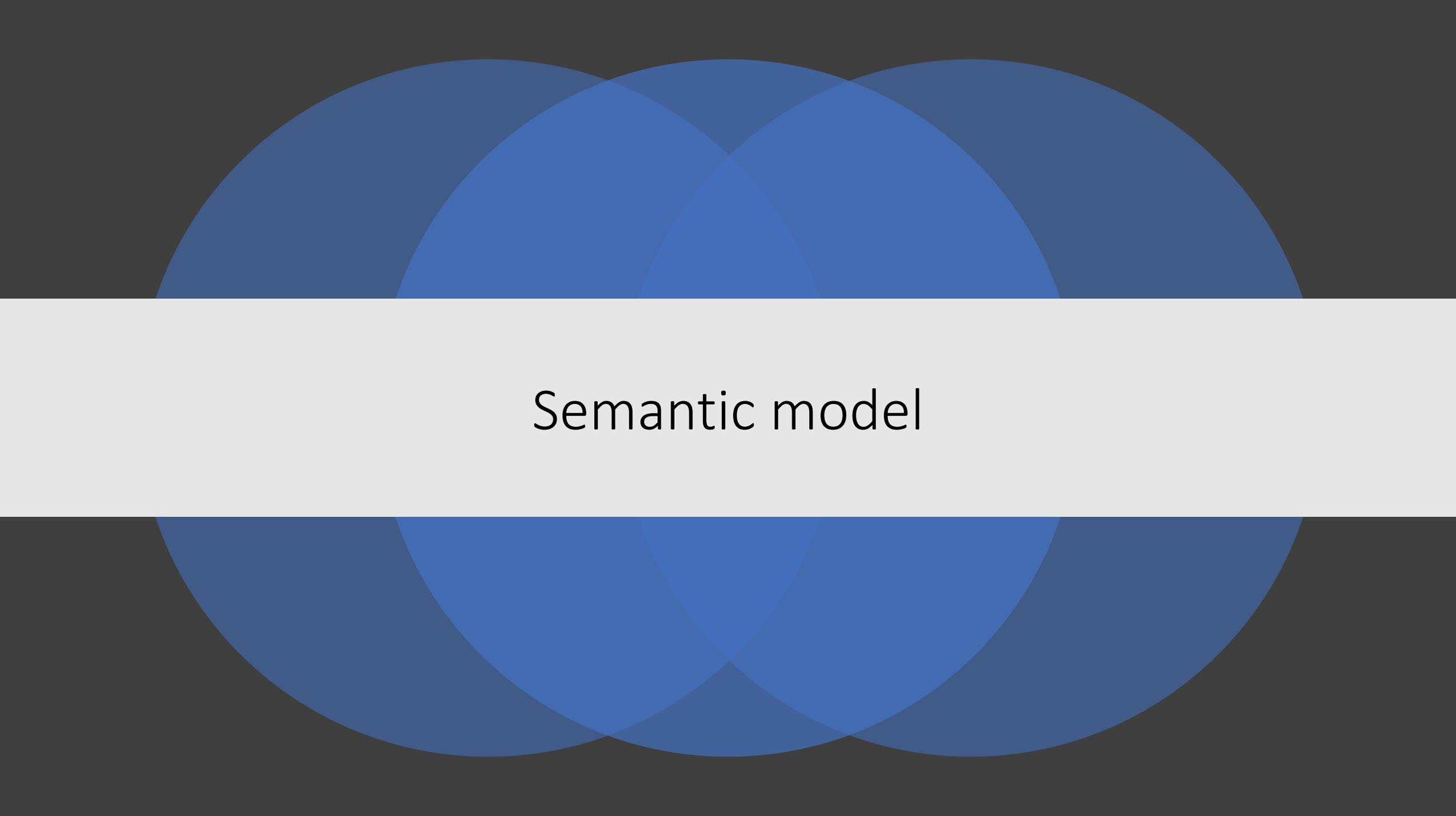
Visit [www.PEDIAA.com](http://www.PEDIAA.com)

- Table

- Stored physically:
  - storage cost overhead
  - ETL load time and cost
- Supports:
  - Row level security
  - Semantic model DirectLake access

- View

- Abstraction layer over tables
  - No storage
  - Compute cost when used
- Supports:
  - Querying using SQL Analytics endpoint
  - No RLS, no DirectLake

The image features a dark gray background with a central white horizontal band. Above and below this band are three overlapping circles in shades of blue, creating a decorative, wave-like pattern. The text "Semantic model" is centered within the white band.

Semantic model

# What is a semantic model?

 **Semantic:** The data has a meaning to a person or organization

**ProductCode**  
VS001393 ● → **Meaning:** Refers to a specific product with attributes; *i.e. Nitrogen engines*

**Quantity**  
2 ● → **Meaning:** Reflects business performance; *i.e. Total units sold*

 **Model:** The objects and calculations represent a real-world process

**Sales model:** Represents the sales process with data; *i.e. Tables, Relationships, DAX measures*



- Translates
  - Technical DB implementation
    - Table, PK, FK
    - Query, Join, Predicate
  - Into
    - Human terminology
- Abstraction over database
  - Allows for drag and drop reporting
  - Enables
    - Self service reporting
    - Faster reporting development
    - Central store for Business Rules

# Demo

Create semantic model

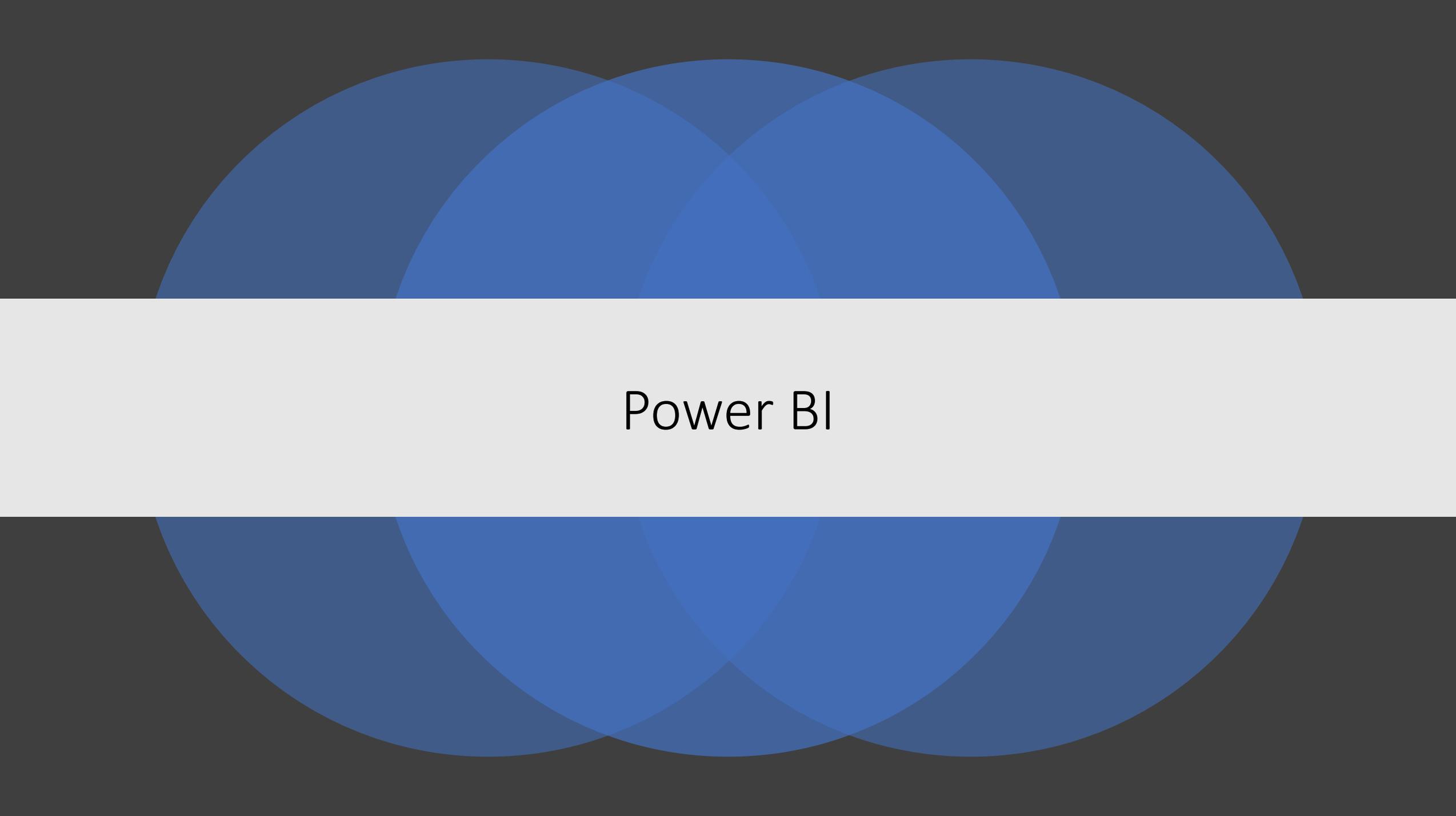
# Storage modes

## DAX objects in different semantic models

	<i>Table storage mode</i>	 <i>Measures</i>	 <i>Calc. columns</i>	 <i>Calc. tables</i>
1	 Import			
2	 DirectQuery			
3	 Hybrid			
4	 Direct Lake			

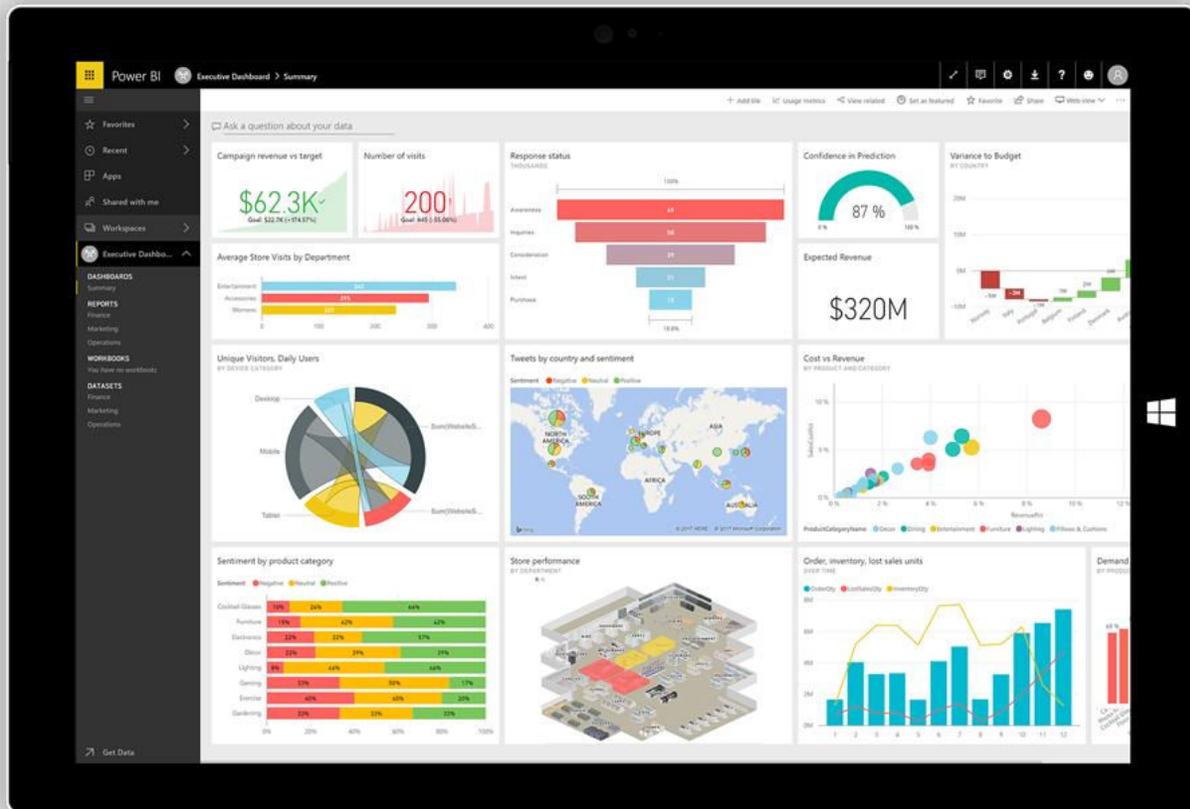
 *Calculated tables work only if they don't reference Direct Lake tables or columns*

- Import
- DirectQuery
- Direct Lake
- Hybrid / Composite models

The image features a dark gray background with a central white horizontal band. Behind the band, there are three overlapping circles in shades of blue, arranged horizontally. The text "Power BI" is centered within the white band.

Power BI

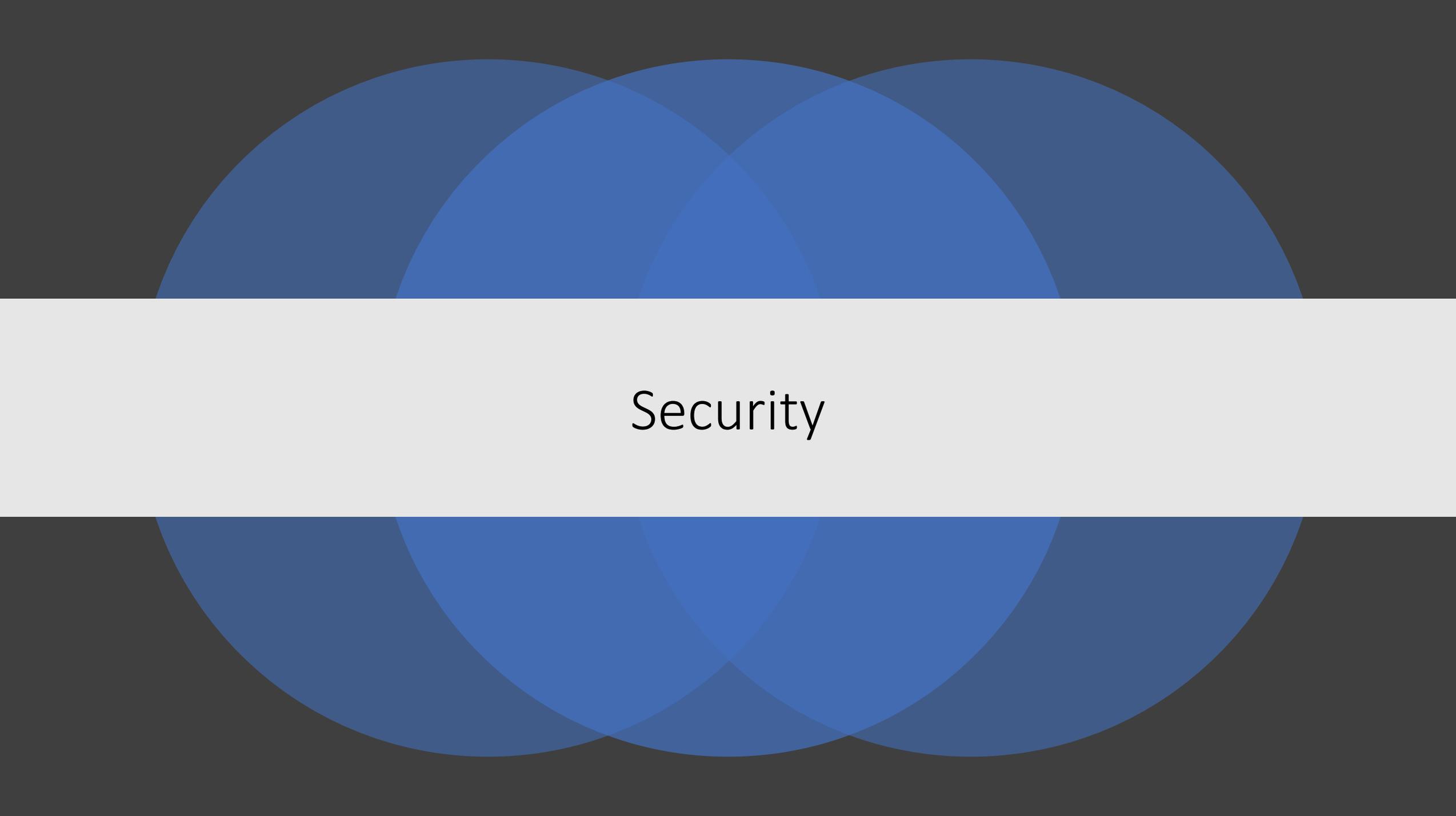
# Power BI



- Data visualization tool
- Use cases:
  - Corporate reporting
  - Exploratory reporting
  - Everything in between
- Parts:
  - Power BI Desktop
  - Power BI Service (Power BI Report Server)
  - Power BI Mobile app
- Power BI service
  - Collaboration
  - Workspaces
    - => Apps
  - Dashboards

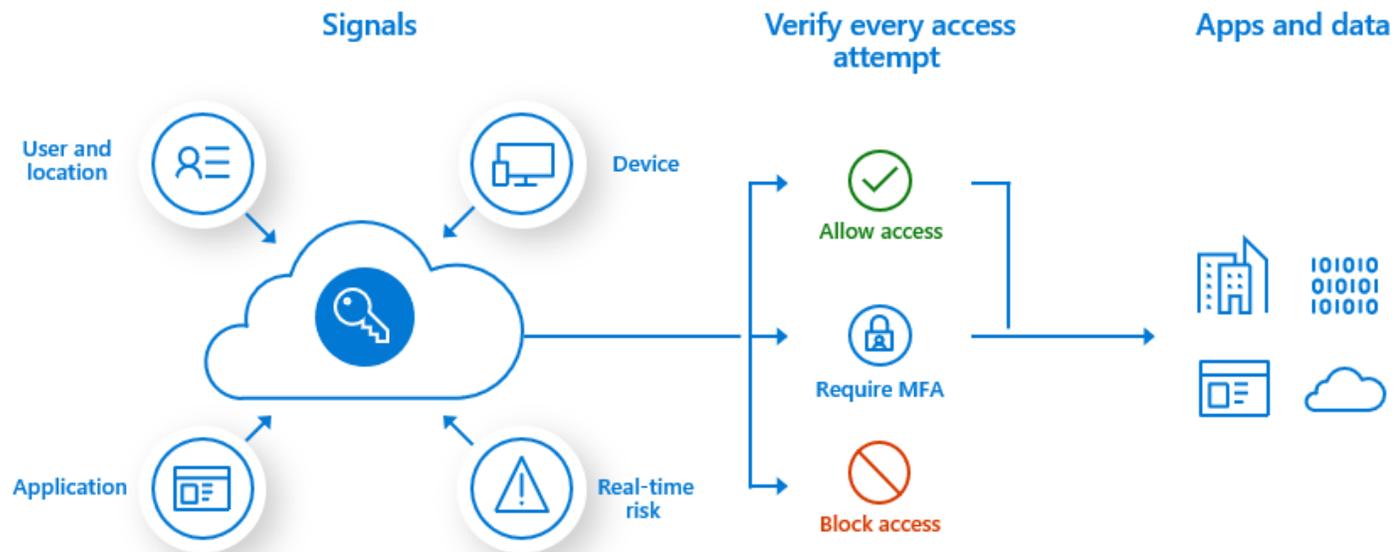
# Demo

Show report



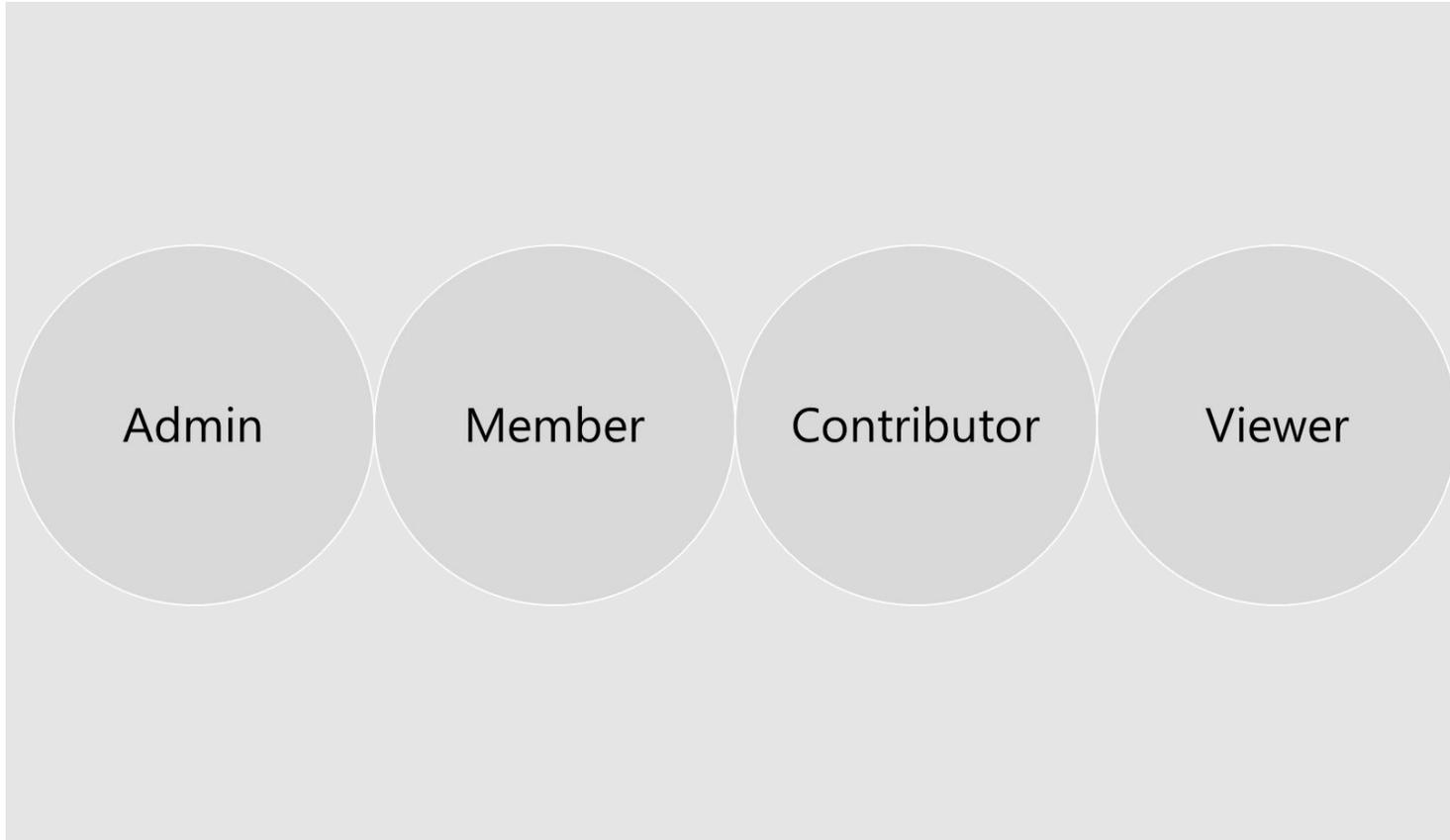
Security

# Fabric security model



- Principle of least privilege:
  - Restrict user permissions to only what's needed
- Three layers:
  - Entra ID authentication
    - Who are you
    - Single sign on
    - MFA
  - Fabric access
    - Checks if user can access Fabric
  - Data security
    - Permissions on actions and data

# Workspace roles



- **Workspace role:**
  - Preconfigured set of permissions
  - Hierarchical: share ALL artifacts
  - Assigned to
    - Individuals
    - Security groups
    - Microsoft 365 groups
    - Distribution lists
- **Workspace roles**
  - **Viewer**
    - View all content and data in workspace
    - Cannot modify anything
  - **Contributor**
    - Viewer +
    - Modify all content and data in the workspace
  - **Member**
    - Contributor +
    - Share all content and data in the workspace
  - **Admin**
    - Member +
    - Manage permissions

# Share individual items

## Lakehouse:

**Grant people access** LearnLakehouse

People you share this Lakehouse with can open it and its SQL endpoint and read the default dataset. To allow them to read directly in the Lakehouse, grant additional permissions.

**Additional permissions**

- Read all SQL endpoint data ⓘ
- Read all Apache Spark ⓘ
- Build reports on the default semantic model

**Notification Options**

- Notify recipients by email

ⓘ Depending on which additional permissions you select, recipients will have different access to the SQL endpoint, default dataset, and data in the lakehouse. For details, view lakehouse permissions documentation.

Grant Back

## Warehouse:

**Grant people access** DemoPTB\_Warehouse

People you share this warehouse with can connect to it. To give additional permissions, select them from the list.

**Additional permissions**

- Read all data using SQL (ReadData) ⓘ
- Read all OneLake data (ReadAll) and subscribe to events (SubscribeOneLakeEvents) ⓘ
- Build reports on the default semantic model (Build) ⓘ
- Monitor queries (Monitor) ⓘ
- Audit queries (Audit) - PREVIEW ⓘ
- Share granted permissions (Reshare) ⓘ

**Notification Options**

- Notify recipients by email

## Semantic model:

**Share semantic model** demo

- Allow recipients to modify this semantic model
- Allow recipients to share this semantic model
- Allow recipients to build content with the data associated with this semantic model
- Send an email notification

- Item sharing = connect permission
  - + read for Semantic model
- Extra permissions often too much (equal to db\_datareader)
  - Use fine grained security of SQL Server / Semantic model

# Granular permissions

```
1  -- Create a custom role
2  CREATE ROLE TestRole
3
4
5  -- Grant permission to role
6  GRANT SELECT ON Sales.Product
7  TO TestRole
8
9
10 -- Add user to role
11 ALTER ROLE TestRole
12 ADD MEMBER [peter@trainsql.onmicrosoft.com]
13
14
15 -- Remove user from role
16 ALTER ROLE TestRole
17 DROP MEMBER [peter@trainsql.onmicrosoft.com]
```

- ▼ DemoPTB\_DataWarehouse
  - > Schemas
  - ▼ Security
    - ▼ DB roles (Built-in)
      - db\_owner
      - db\_accessadmin
      - db\_securityadmin
      - db\_ddladmin
      - db\_backupoperator
      - db\_datareader
      - db\_datawriter
      - db\_denydatareader
      - db\_denydatawriter
    - ▼ DB roles (Custom)
      - public

- Assign only Read permission in Fabric
- Add T-SQL
  - GRANT
  - DENY
  - REVOKE
- No explicit CREATE USER
- Database roles
  - Collection of permissions
  - Built-in roles
  - Custom roles
- Row Level Security
  - Warehouse, Semantic model

# OneLake data access roles

The screenshot displays the 'Manage OneLake data access (preview)' interface. On the left, a sidebar contains a '+ New role' button, 'Assign' and 'Delete' buttons, and a search bar. Below this, a list of roles is shown, including 'Role name' and 'DefaultReader'. The main area is a 'New role (preview)' dialog box for 'LearnLakehouse'. It includes a description: 'Grant this role Read permissions to the selected data. [Learn more](#)'. An 'Assign role' button is present. The 'Role name' field contains 'SalesOnly'. Under 'Included folders', 'Selected folders' is selected. The folder tree shows '\Tables Folder' (expanded) with 'product-subcategory' and 'sales' (checked), and '\Files Folder' (expanded) with 'adw'.

- Lake view of lakehouse is used to read data in Tables and Files folder
- Workspace / item permissions
  - Coarse access to data in lakehouse
- OneLake data access roles (preview)
  - Fine grained access
  - Create custom roles within lakehouse
  - Grant read permissions only to specific folders in OneLake
  - Folder security is inheritable to all subfolders
- Create custom OneLake role:
  - Select Manage OneLake data access (preview)
    - Toolbar in lake view of lakehouse
  - In New Role window
    - Provide role name
    - Select folders
  - Assign user / group to role
  - Select permissions:
    - Read
    - Write
    - Reshare
    - Execute
    - ReadAll

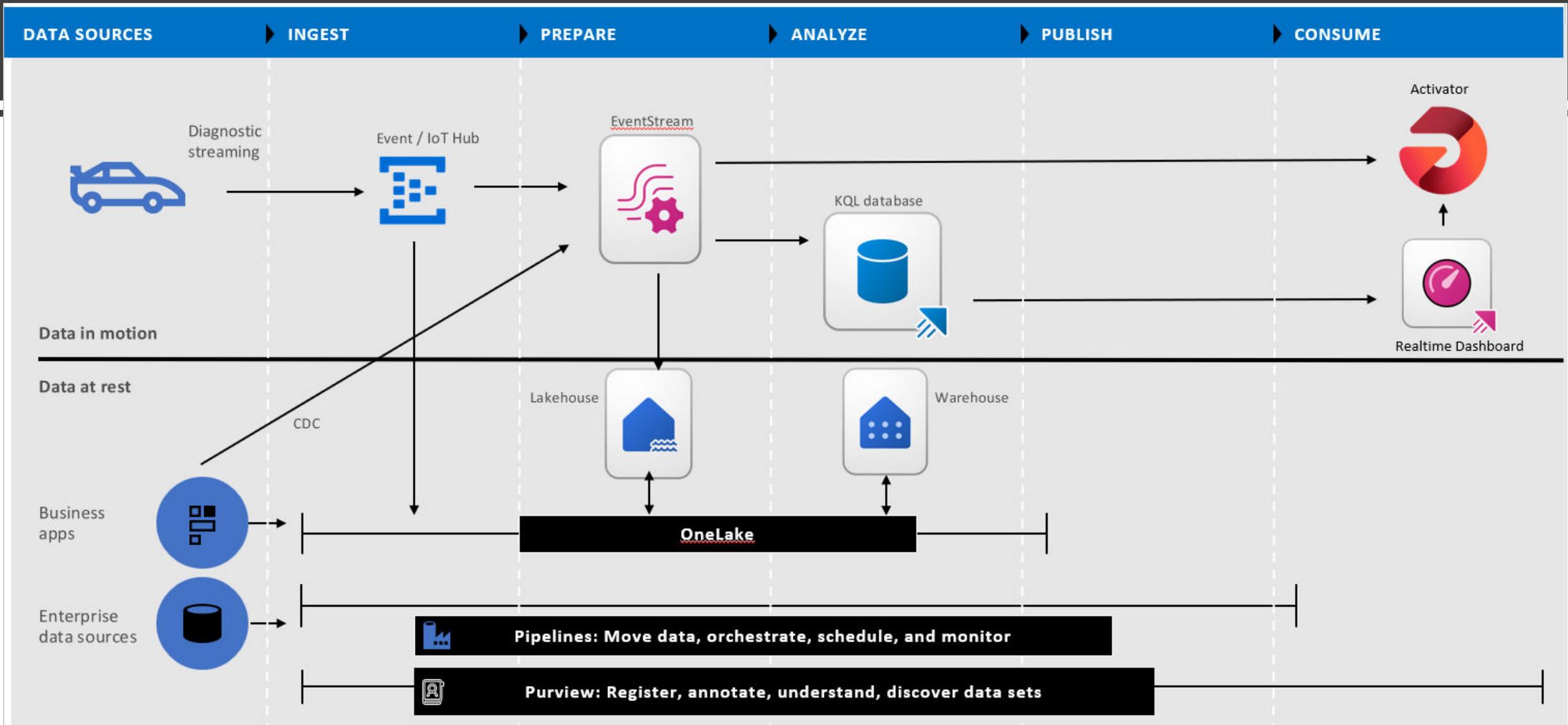
# Demo – If Time Permits

Row Level Security

The logo graphic consists of three overlapping circles in shades of blue, arranged horizontally. The circles overlap in the center, creating a darker blue area. A white horizontal band cuts through the middle of the circles.

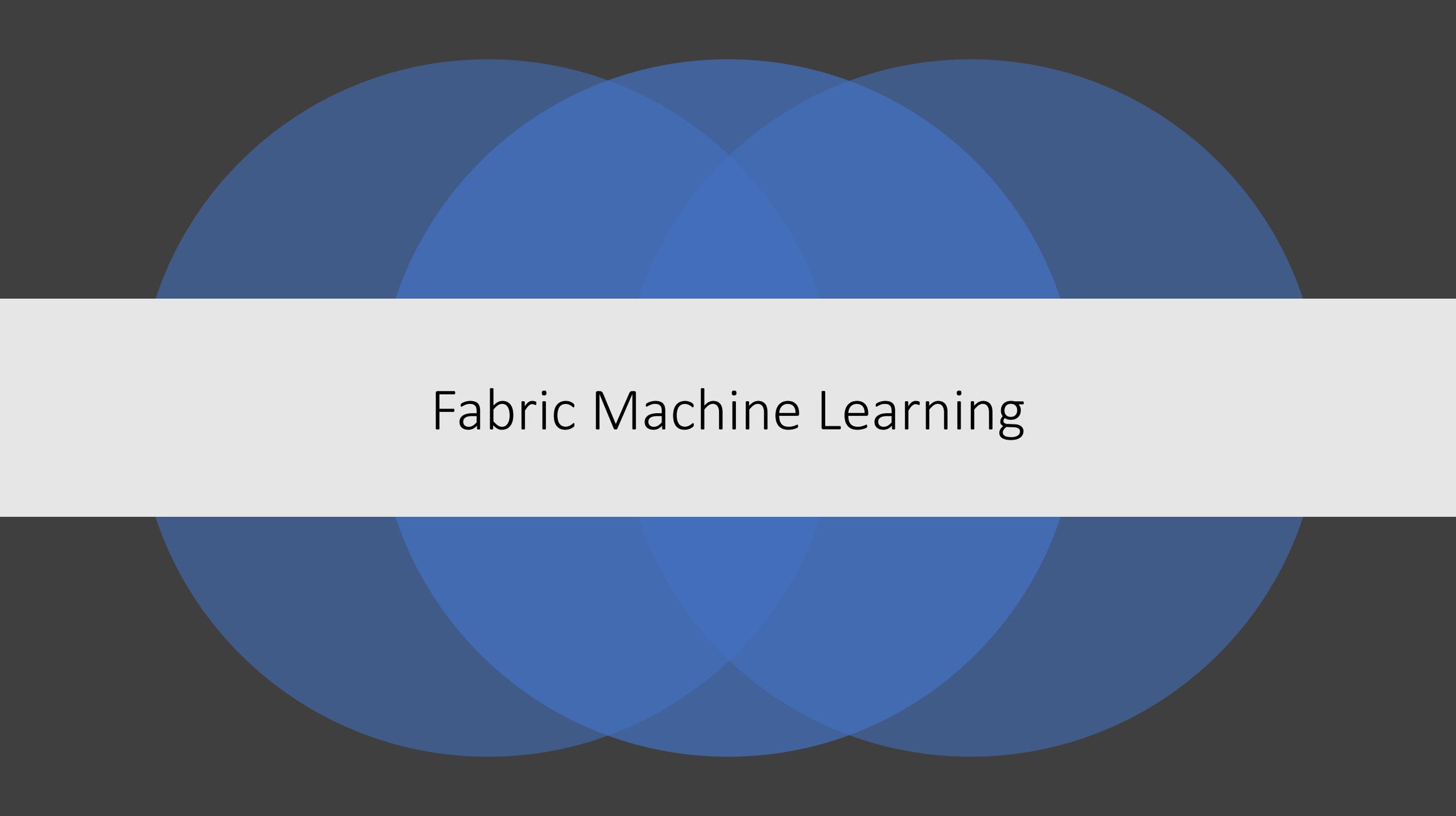
Eventhouse

# Streaming Data Architecture Overview



# Demo – If Time Permits

Stock eventstream

The image features a dark gray background with a central white horizontal band. Behind the band, three overlapping circles in shades of blue are visible, creating a decorative pattern. The text "Fabric Machine Learning" is centered within the white band.

# Fabric Machine Learning

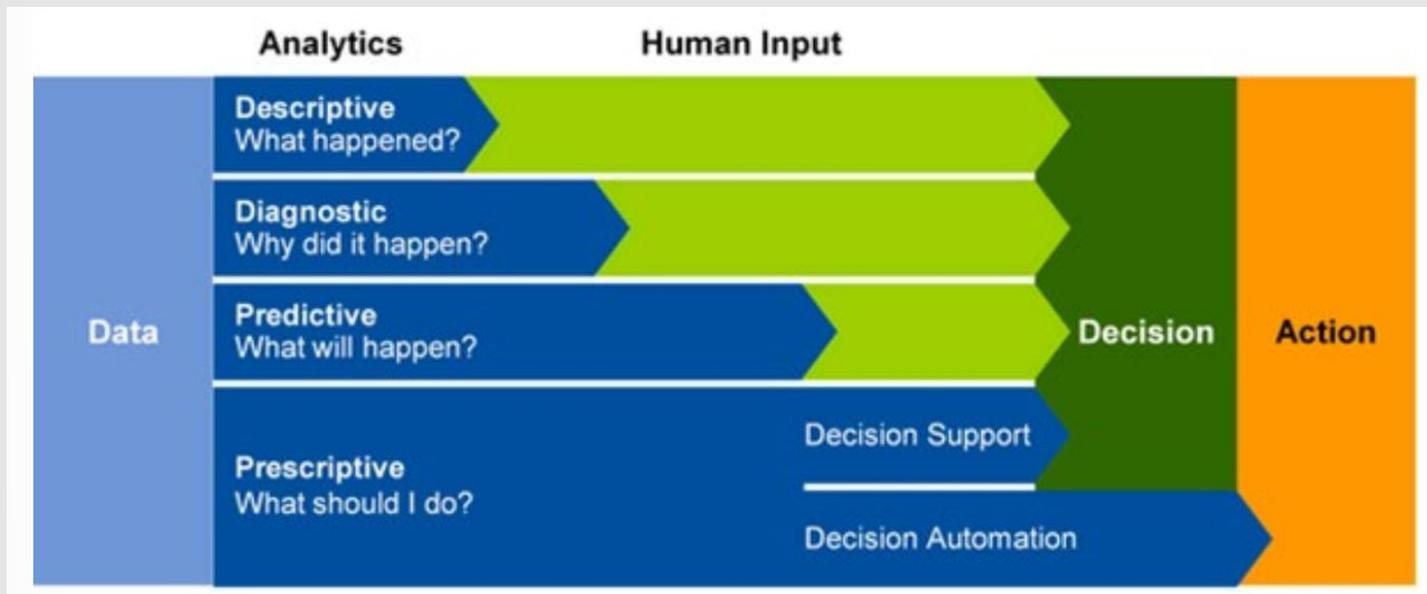
# Artificial Intelligence?



Software that imitates human capabilities

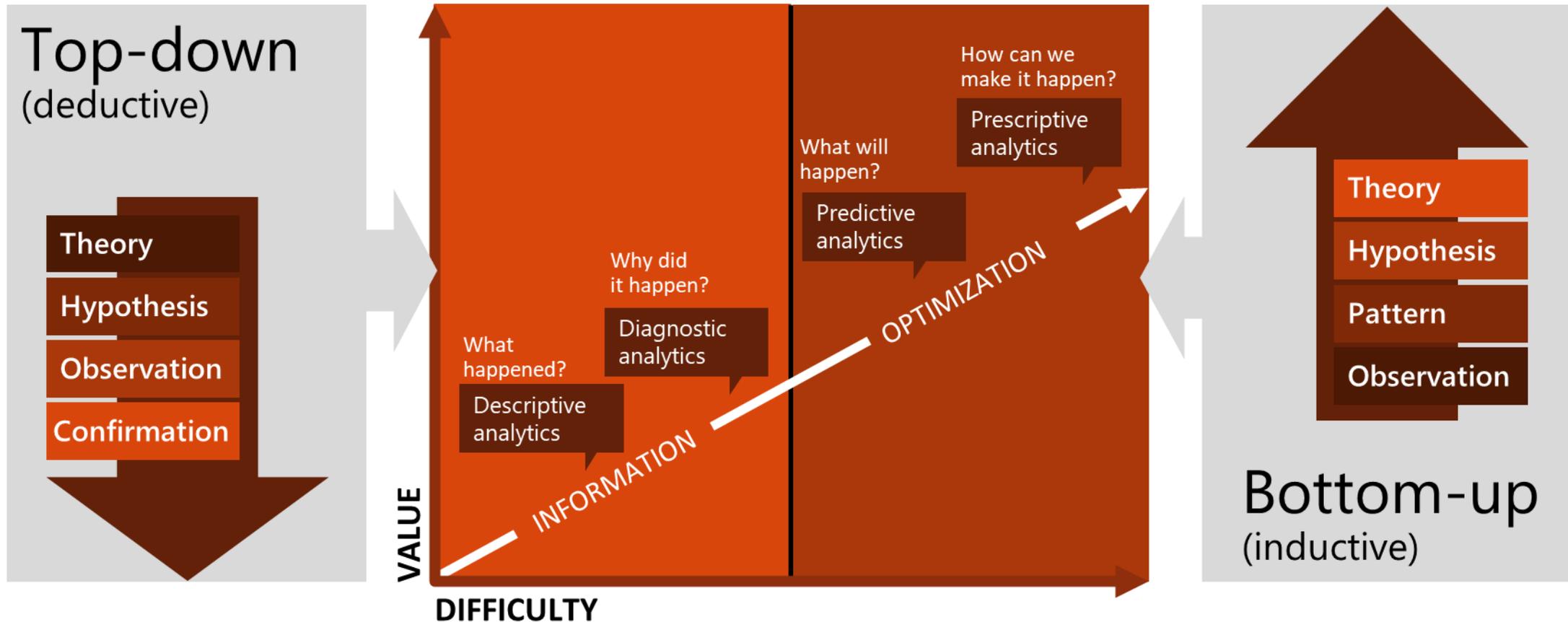
- Predicting outcomes and recognizing **patterns** based on historic data
- Recognizing abnormal events and making decisions
- Interpreting visual input
- Understanding language, engaging in conversations
- Extracting information from sources to gain knowledge

# Advanced Analytics

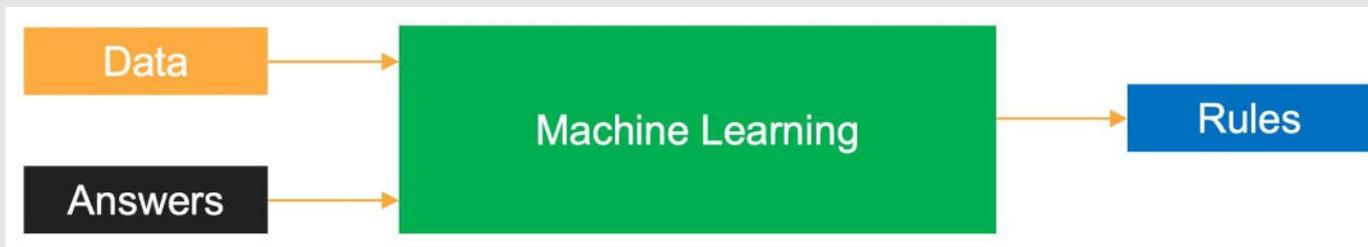
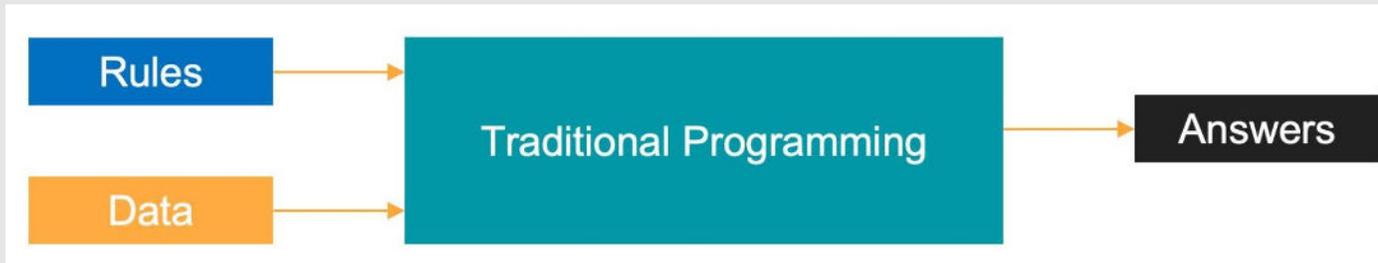


- Machine learning is a type of artificial intelligence that provides computers with the ability to learn without being explicitly programmed
  - They do not follow strictly static program instructions
- Formal definition: “A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ ” - *Tom M. Mitchell*
- “The goal of machine learning is to program computers to use example data or past experience to solve a given problem.” — *Introduction to Machine Learning, 2<sup>nd</sup> Edition, MIT Press*

# Top-down versus bottom-up



# Machine Learning



- Supervised learning
- Unsupervised learning
- Semi-supervised learning
  
- With tabular data
  - 1 row = 1 observation
  - 1 column =
    - 1 feature
    - Label to predict
  
- Columns/features can be
  - Continuous (numerical)
  - Discrete (categorical)
  - Datetime
  - Text
  
- Labels:
  - Continuous: regression
  - Discrete: classification

# Analyze and train data

## Analyze and train data

Propose hypotheses, train models, and explore your data to make decisions and predictions.

### Anomaly detector (preview)



Detect anomalies in eventhouse tables and set alerts.



### Data agent (preview)



Build a generative AI agent that understands your data and can answer complex questions in a variety of conversational interfaces.



### Environment



Set up shared libraries, Spark compute settings, and resources for notebooks and Spark job definitions.



### Experiment



Create, run, and track development of multiple models for validating hypotheses.



### Graph model (preview)



Explore data and build algorithms with Graph



### ML model



Use machine learning models to predict outcomes and detect anomalies in data.



- Lake view of lakehouse is used to read data in Tables and Files folder
- Workspace / item permissions
  - Coarse access to data in lakehouse
- OneLake data access roles (preview)
  - Fine grained access
  - Create custom roles within lakehouse
  - Grant read permissions only to specific folders in OneLake
  - Folder security is inheritable to all subfolders
- Create custom OneLake role:
  - Select Manage OneLake data access (preview)
    - Toolbar in lake view of lakehouse
  - In New Role window
    - Provide role name
    - Select folders
  - Assign user / group to role
  - Select permissions:
    - Read
    - Write
    - Reshare
    - Execute
    - ReadAll

# Demo – If Time Permits

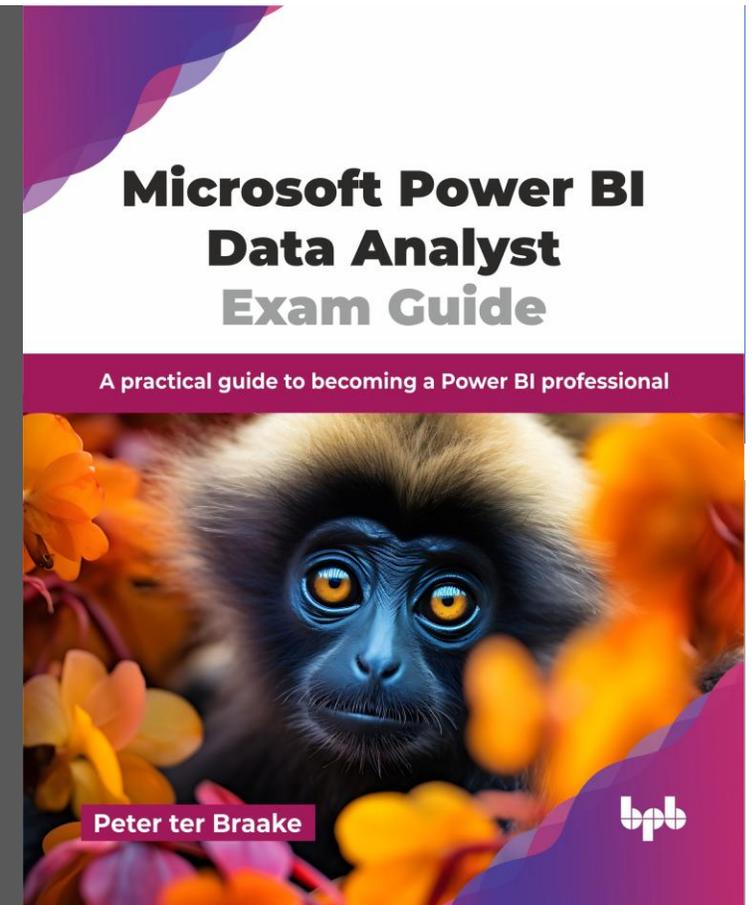
Predict Loan

Questions ?



# Thank you!

Download slides:  
<add tiny url here>



Peter ter Braake  
trainsql@live.nl

15% Discount on PowerBI exam guide:  
<https://tinyurl.com/39cs9ync> + code POWERBI15