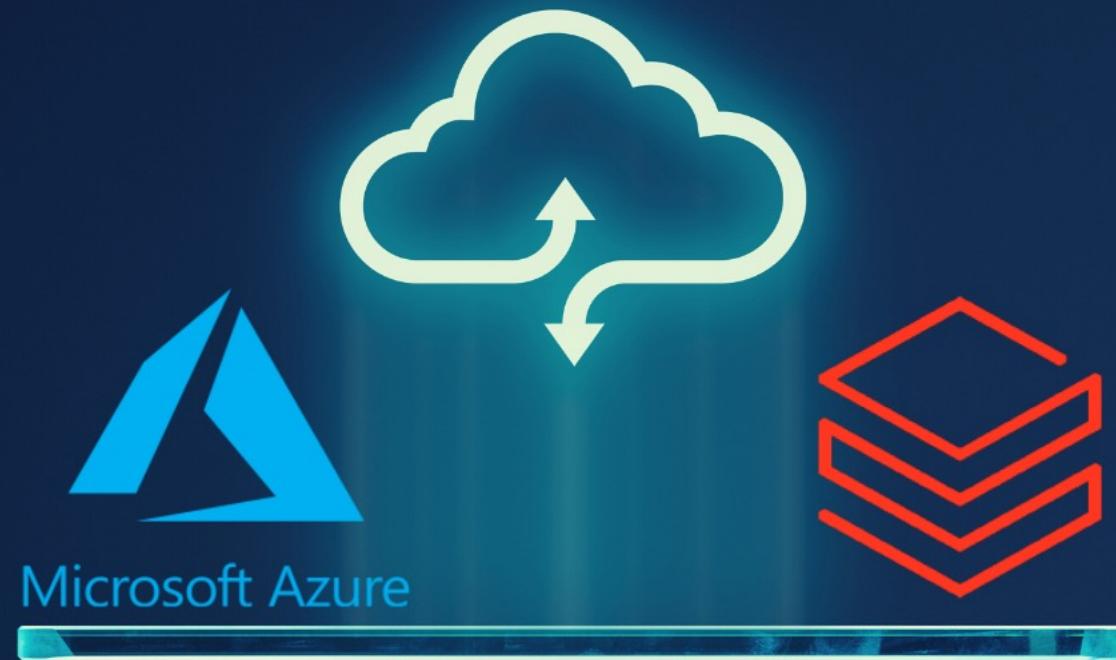


Azure Databricks For Data Engineers

Project on Formula1 Racing



About Me



Ramesh Retnasamy
Data Engineer/ Machine Learning Engineer



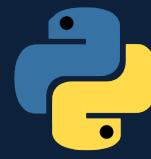
LinkedIn

<https://www.linkedin.com/in/ramesh-retnasamy/>

About this course



Azure Databricks



Python

PySpark



Spark SQL



Delta Lake

About this course



Azure Databricks



Azure Data Lake Storage Gen2



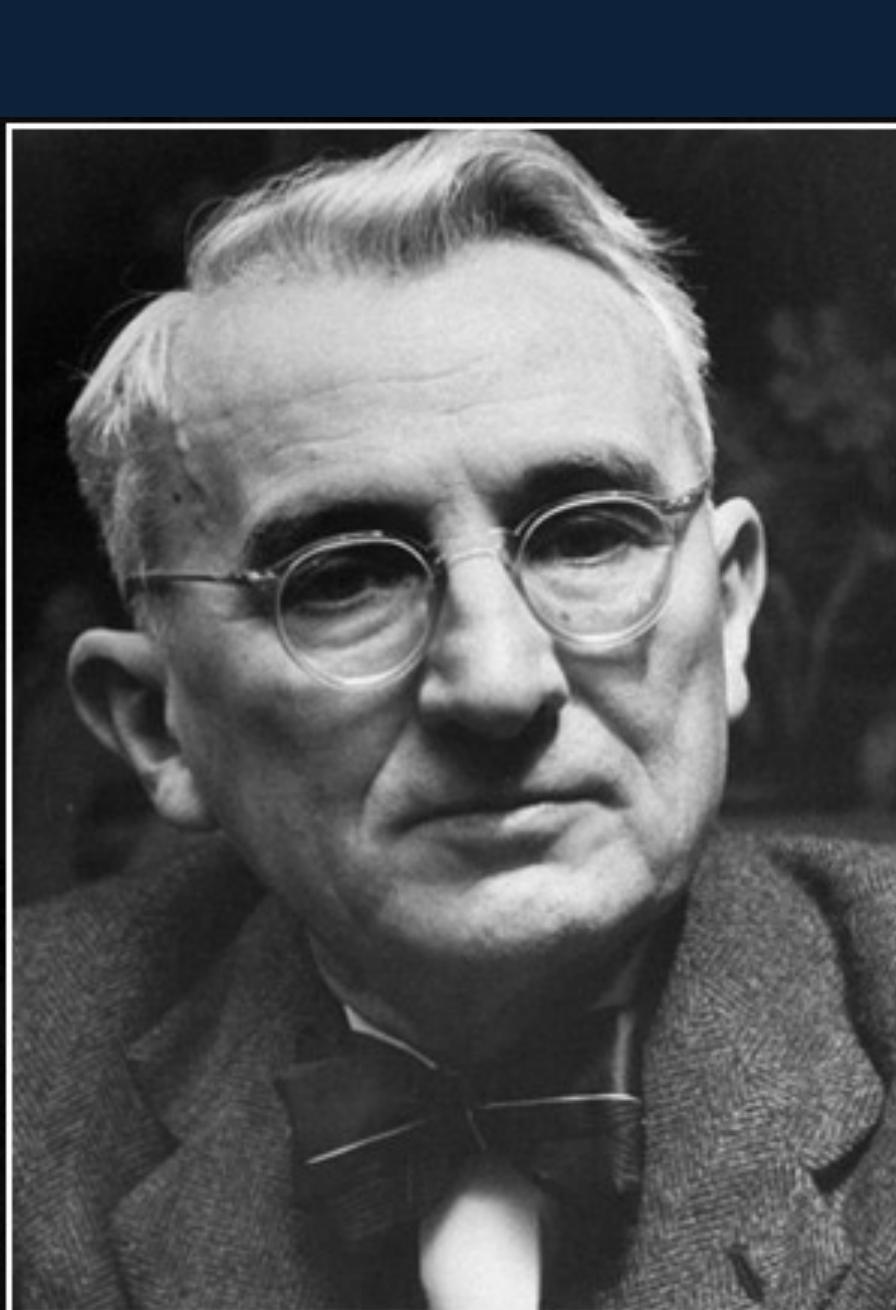
Azure Data Factory



Azure Key Vault



PowerBI

A black and white portrait of Dale Carnegie. He is shown from the chest up, wearing round-rimmed glasses and a dark, textured suit jacket over a light-colored shirt. He has a thoughtful expression, with his right hand resting against his chin. The background is dark and out of focus.

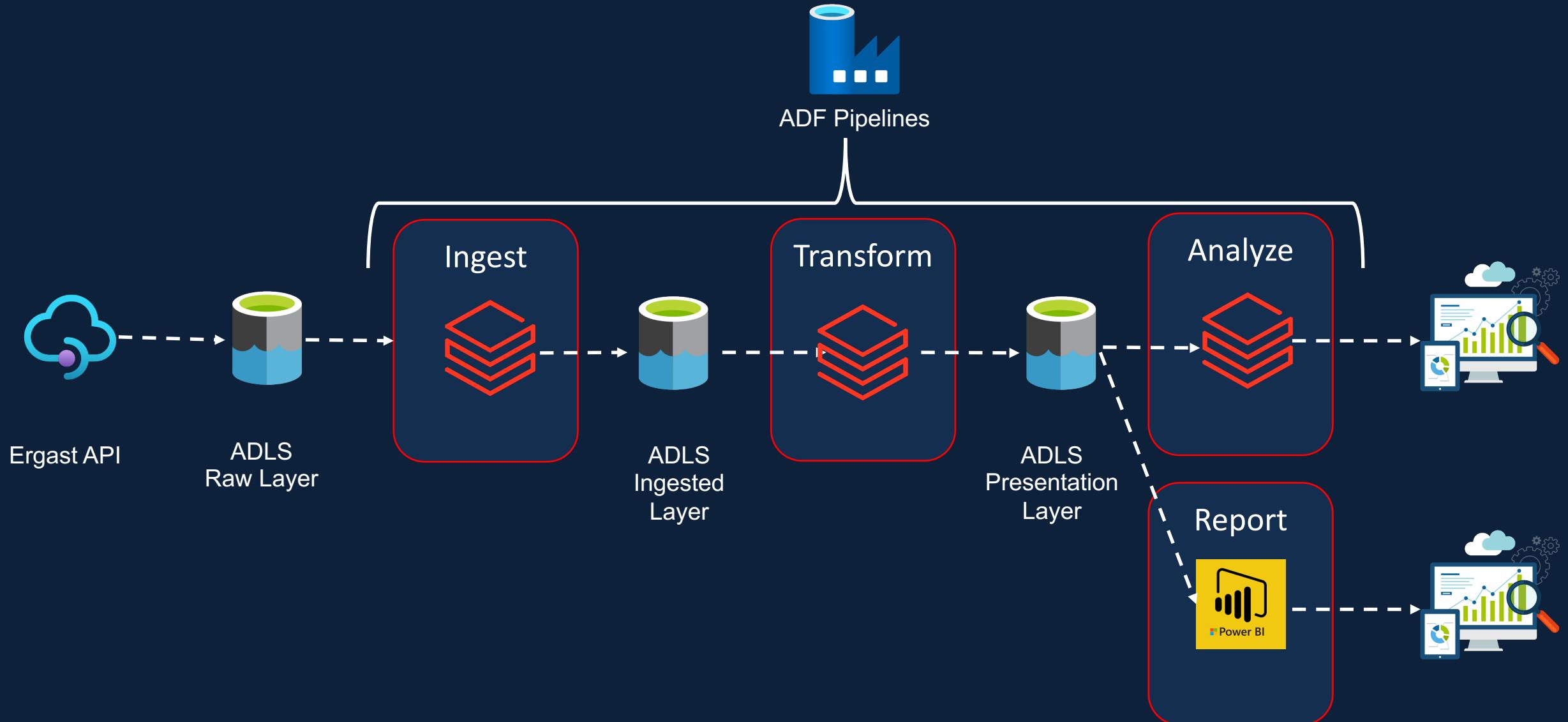
Learning is an active process. We
learn by doing.. Only knowledge that
is used sticks in your mind.

— *Dale Carnegie* —

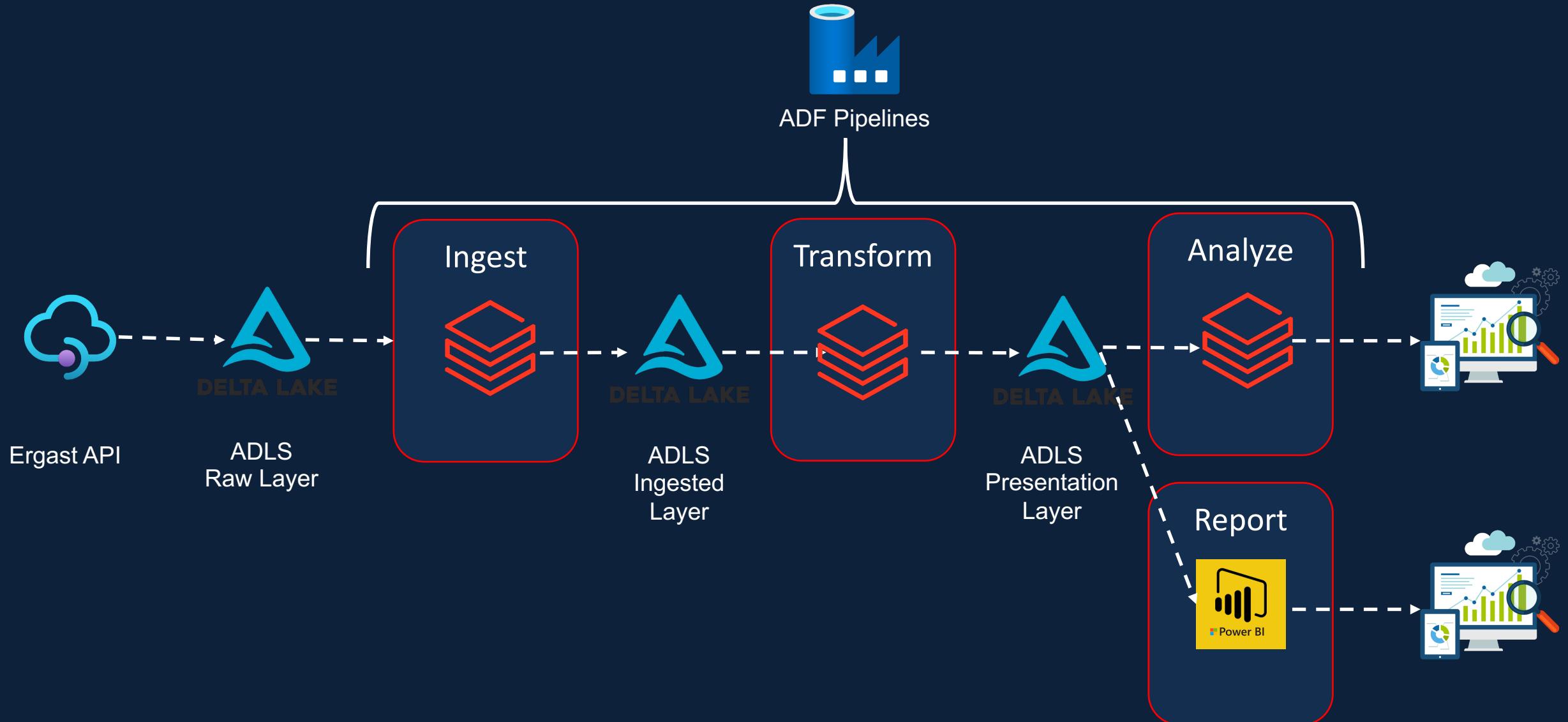


Formula1 Cloud Data Platform

Formula1 Cloud Data Platform

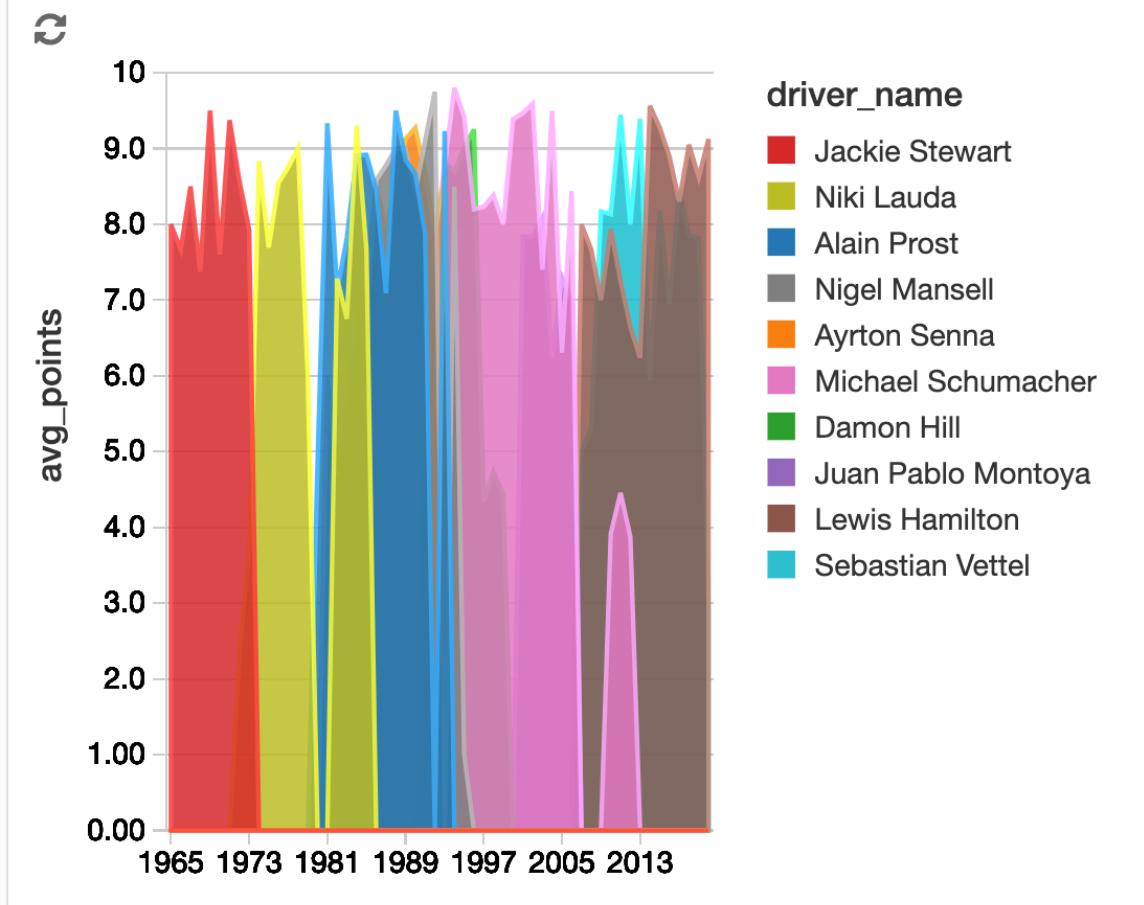
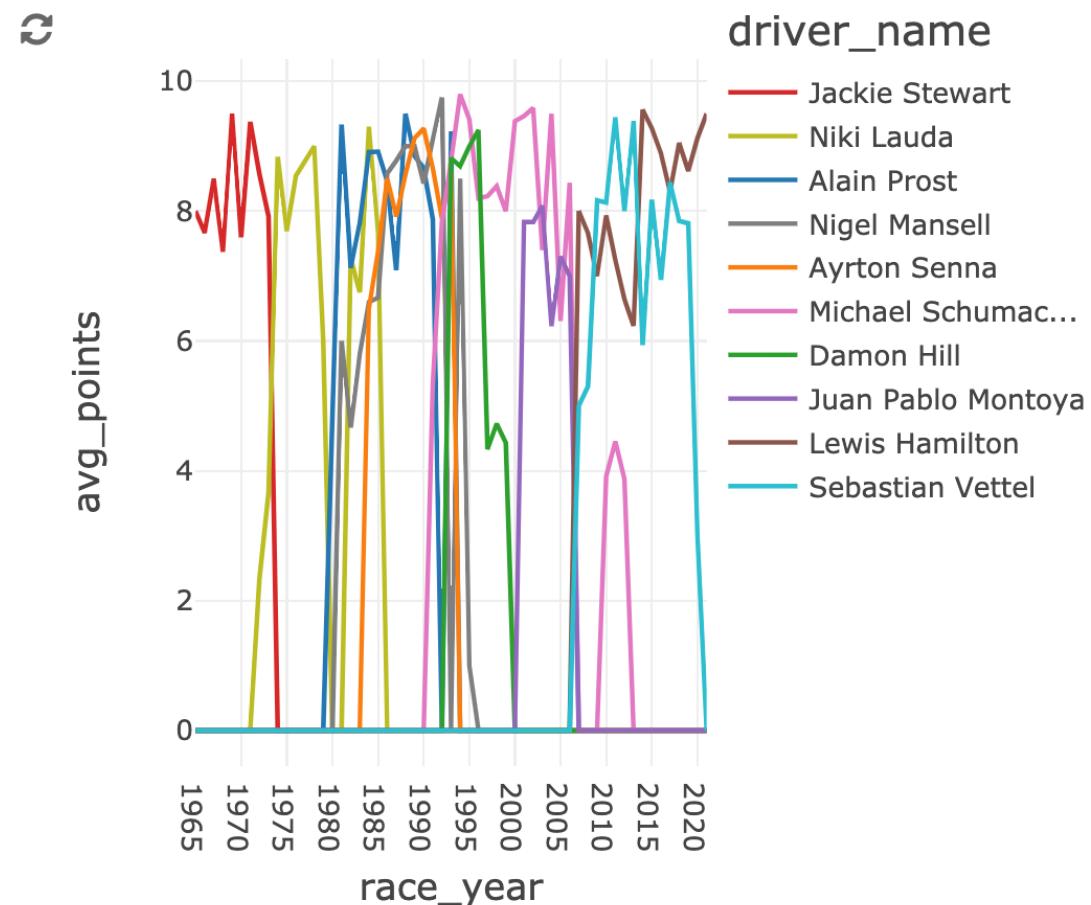


Formula1 Cloud Data Platform



Formula1 Cloud Data Platform

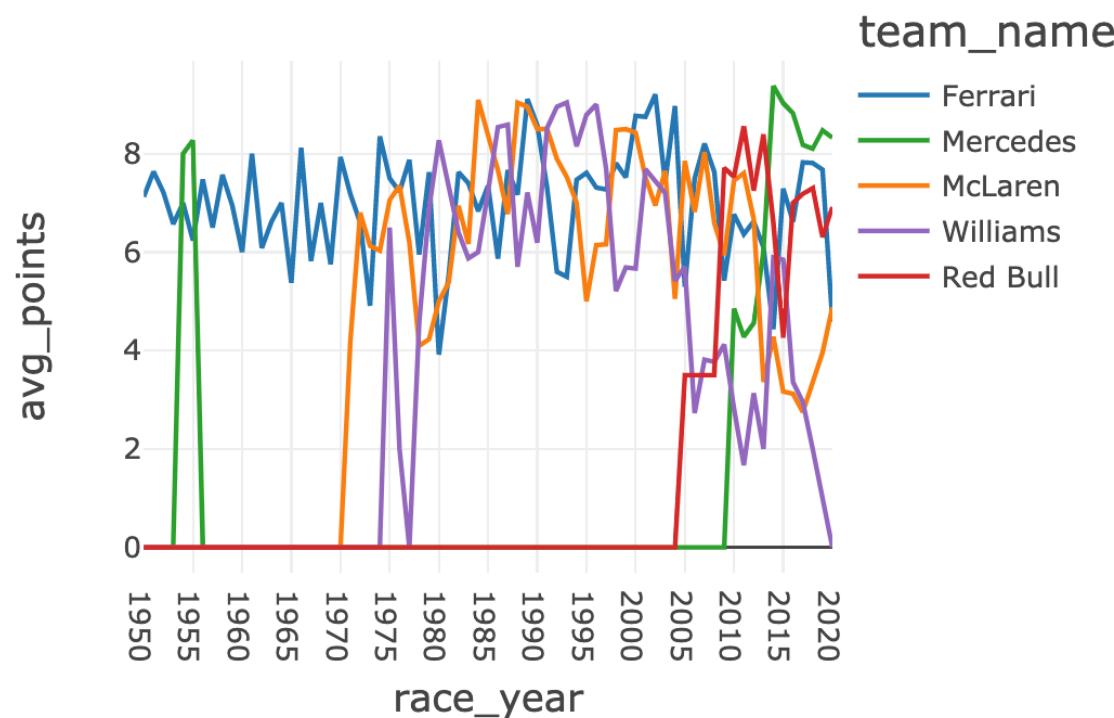
Dominant Formula 1 Drivers of All Time



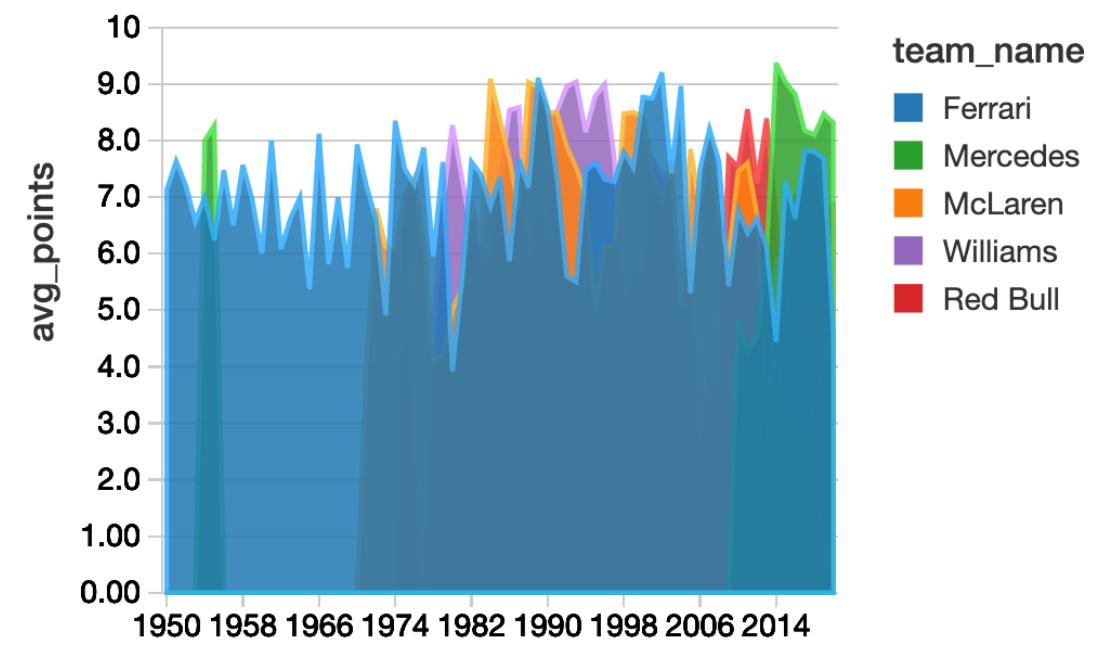
Formula1 Cloud Data Platform

Dominant Formula 1 Teams of All Time

Dominant Teams



Dominant Teams



Who is this course for

University students

IT Developers from other disciplines

AWS/ GCP/ On-prem Data Engineers

Data Architects

Who is this course not for

You are not interested in hands-on learning approach

Your only focus is Azure Data Engineering Certification

You want to Learn Spark ML or Streaming

You want to Learn Scala or Java

Pre-requisites

All code and step-by-step instructions provided

Basic Python programming knowledge required

Basic SQL knowledge required

Cloud fundamentals will be beneficial, but not mandatory

Azure Account

Our Commitments

Ask Questions, I will answer 😊

Keeping the course up to date

Udemy life time access

Udemy 30 day money back guarantee

Course Structure

Overviews

2.Azure Portal

3.Azure Databricks

9.Project Overview

10.Spark Overview

Databricks

4. Clusters

5. Notebooks

6. Data Lake Access

7. Securing Access

8. Databricks Mounts

14. Jobs

Spark (Python)

11. Data Ingestion 1

12. Data Ingestion 2

13. Data Ingestion 3

15. Transformation

16. Aggregations

19.Incremental Load

Spark (SQL)

17. Temp Views

18. DDL

19. DML

20. Analysis

21.Incremental Load

Delta Lake

22.Delta Lake

Orchestration

23.Azure Data Factory

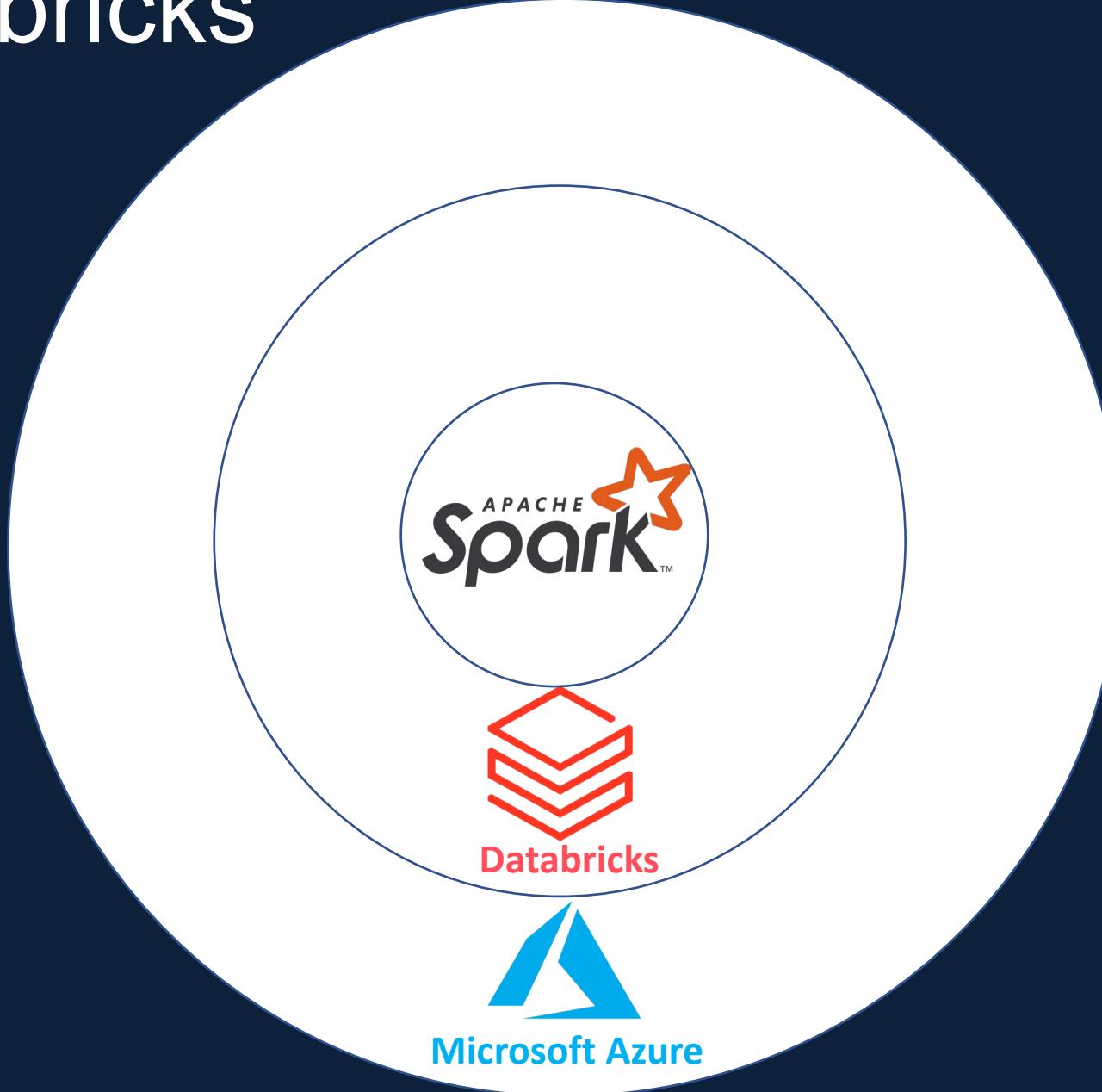
24.Connecting Other Tools



Introduction to Azure Databricks



Azure Databricks



Apache Spark

Apache Spark is a lightning-fast unified analytics engine for big data processing and machine learning



100% Open source under Apache License

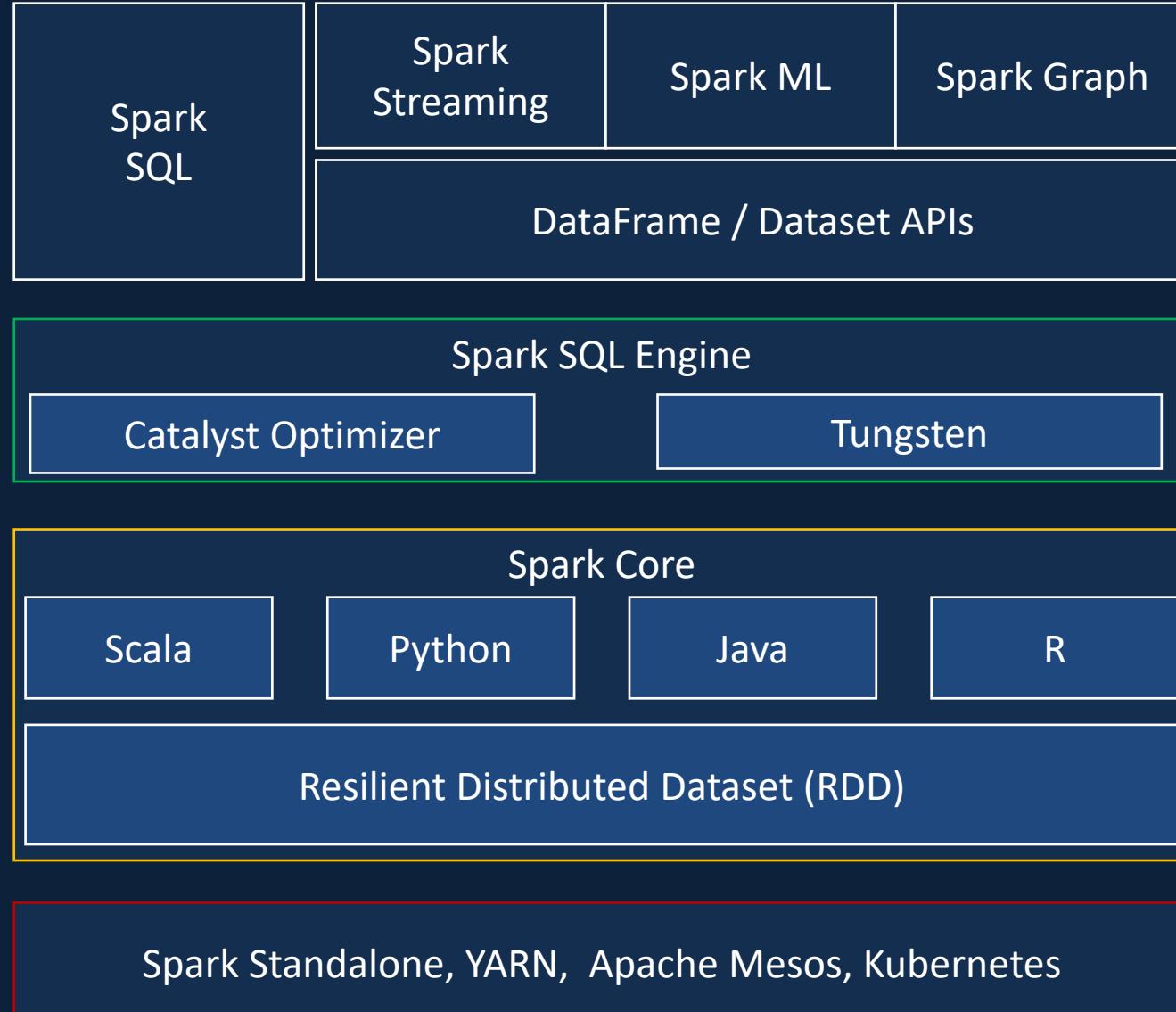
Simple and easy to use APIs

In-memory processing engine

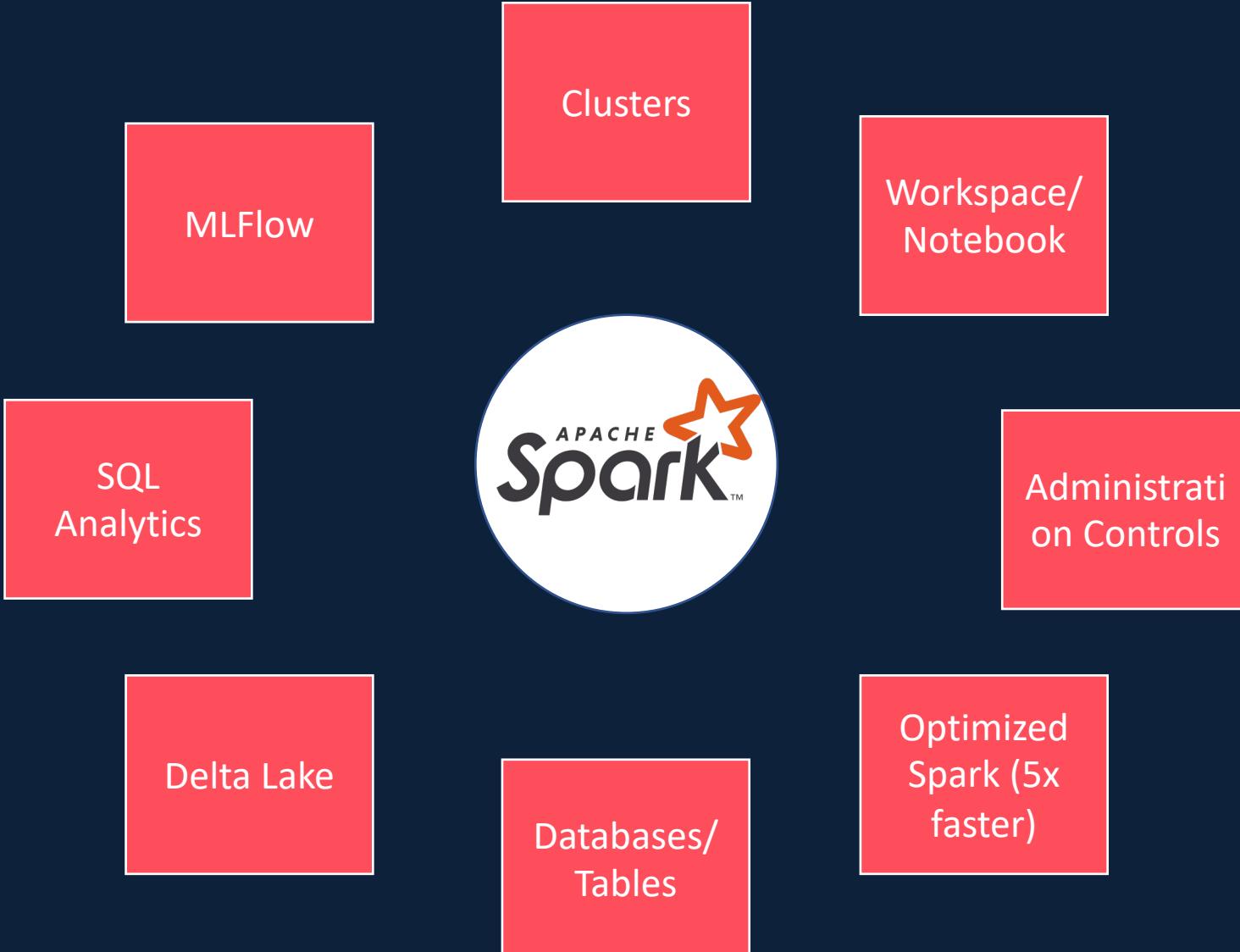
Distributed computing Platform

Unified engine which supports SQL, streaming, ML and graph processing

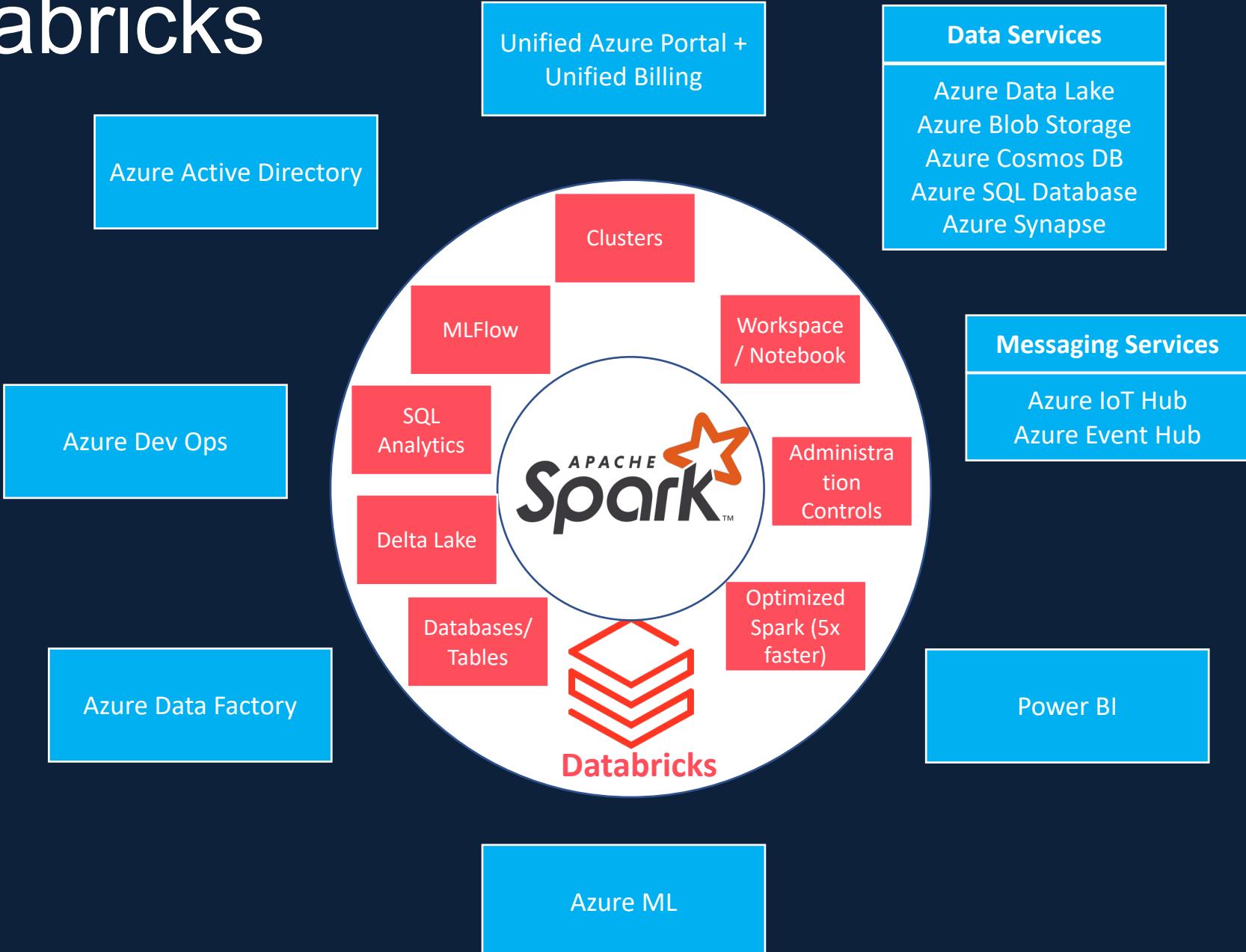
Apache Spark Architecture



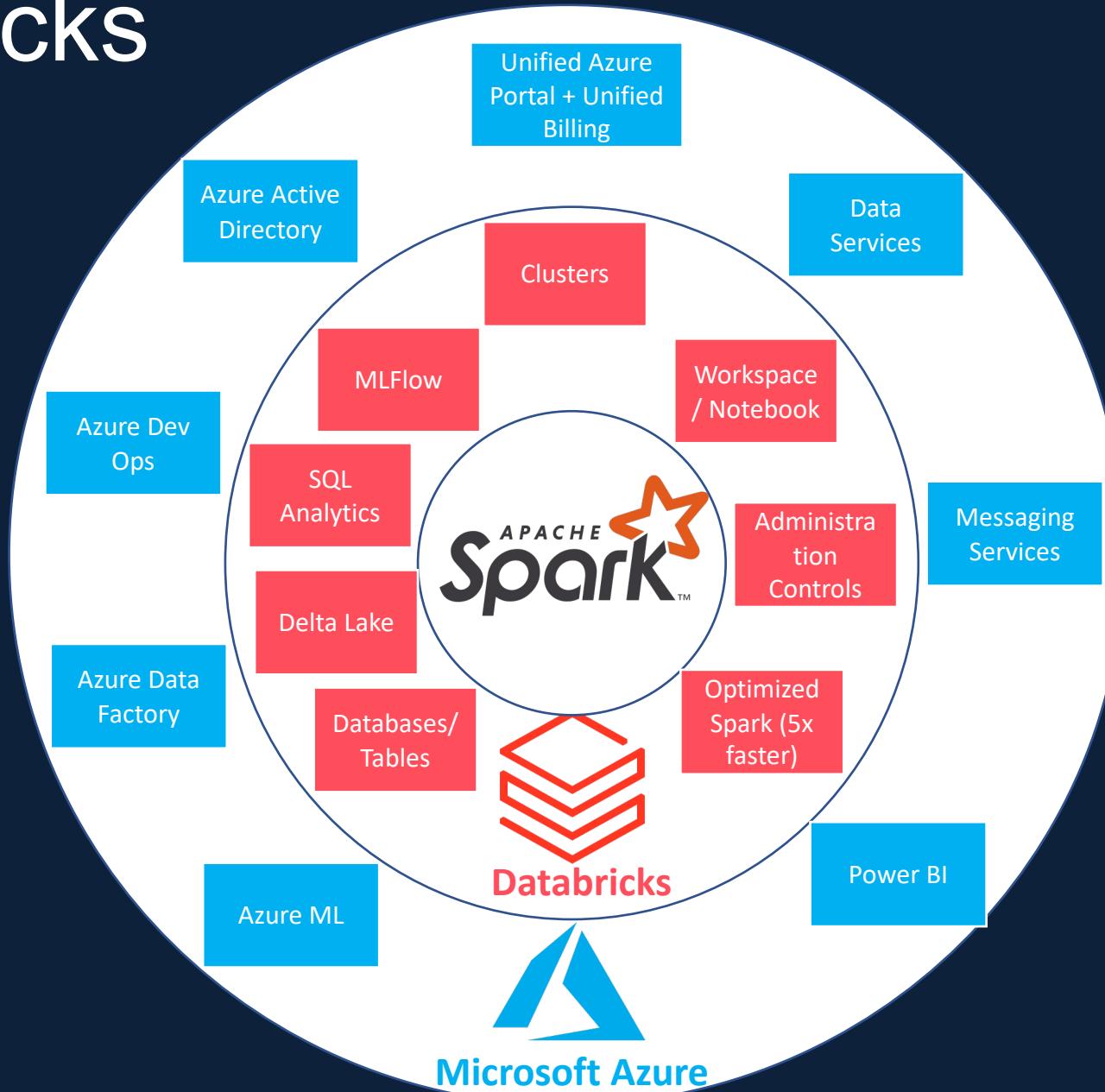
Databricks



Azure Databricks



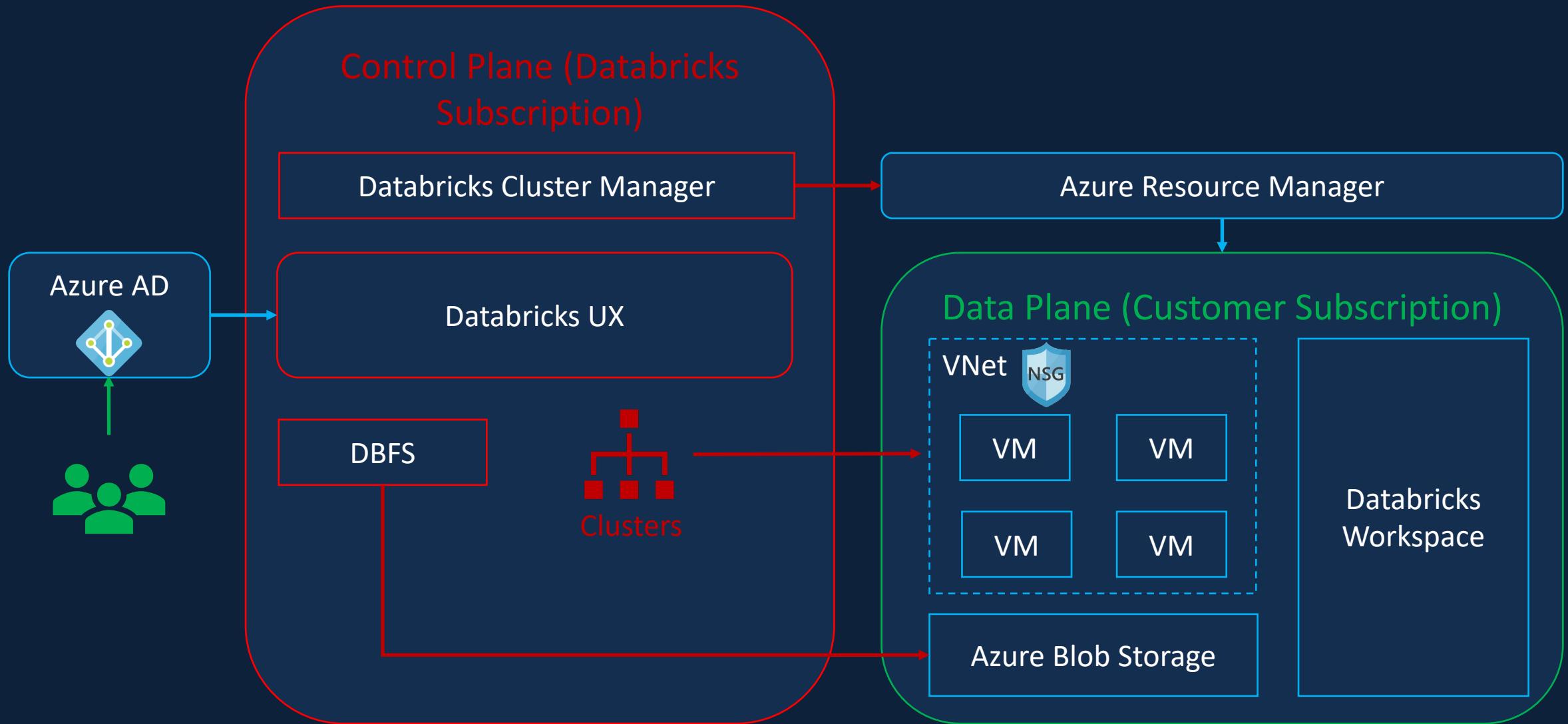
Azure Databricks



Creating Azure Databricks Service



Azure Databricks Architecture



Databricks Workspace Components

Notebooks

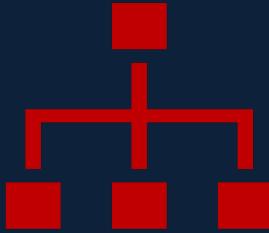
Data

Clusters

Jobs

Models

Databricks Clusters



What is Databricks Cluster

Cluster Types

Cluster Configuration

Creating a cluster

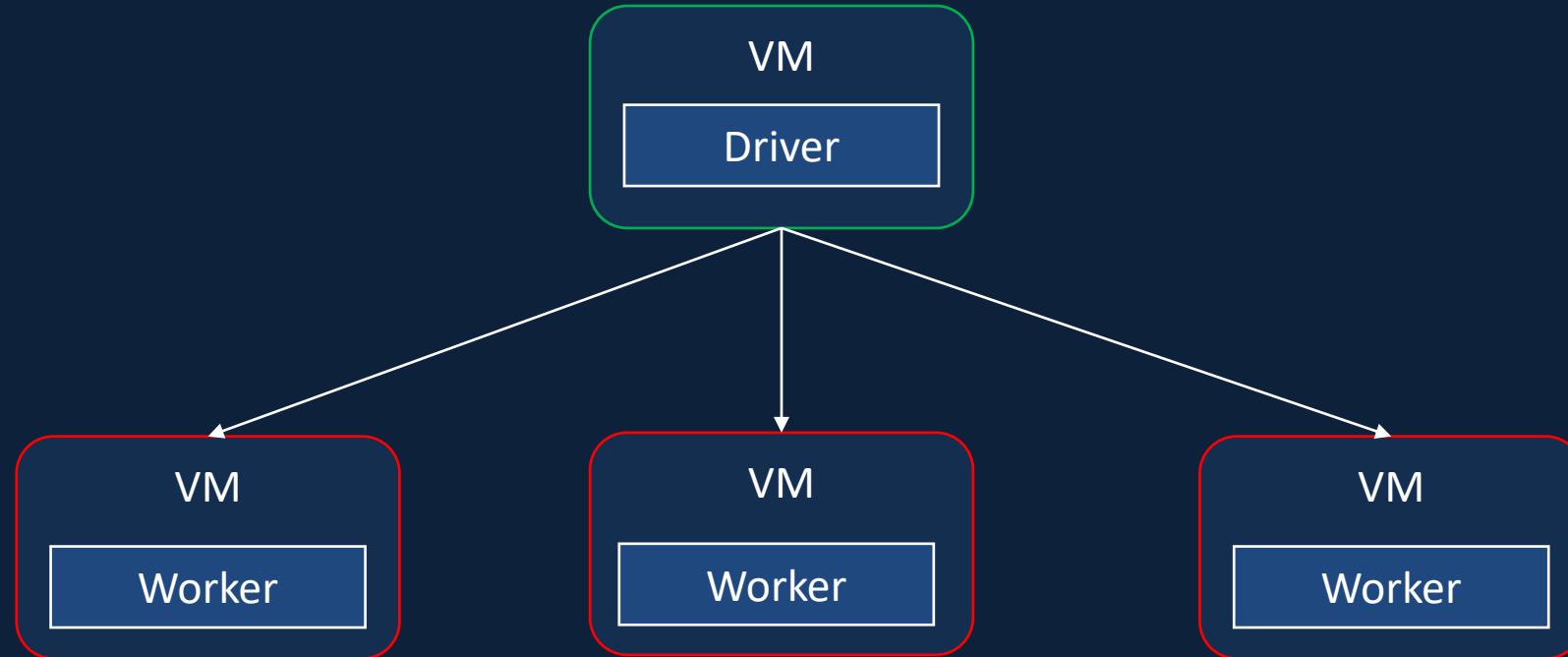
Pricing

Cost Control

Cluster Pools

Cluster Policy

Databricks Cluster



Cluster Types

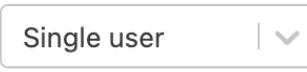
All Purpose	Job Cluster
Created manually	Created by Jobs
Persistent	Terminated at the end of the job
Suitable for interactive workloads	Suitable for automated workloads
Shared among many users	Isolated just for the job
Expensive to run	Cheaper to run

Cluster Configuration

Cluster Configuration 

Policy  

Multi node Single node

Access mode  **Single user access**   

Performance

Databricks runtime version  

Use Photon Acceleration 

Worker type	Min workers	Max workers
Standard_DS3_v2 	2	8

 Spot instances 

Driver type 

Enable autoscaling 

Terminate after  minutes of inactivity 

Cluster Configuration

Single/ Multi Node

Multi Node

Single Node

Cluster Configuration

Single/ Multi Node

Access Mode

Single User

Only One User Access
Supports Python, SQL, Scala, R

Shared

Multiple User Access
Only available in Premium. Supports Python, SQL

No Isolation Shared

Multiple User Access
Supports Python, SQL

Custom

Legacy Configuration

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Spark

Databricks Runtime

Scala, Java,
Python, R

Ubuntu
Libraries

GPU
Libraries

Delta Lake

Other Databricks Services

Databricks Runtime ML

Everything from
Databricks runtime

Popular ML Libraries (PyTorch, Keras,
TensorFlow, XGBoost etc)

Photon Runtime

Everything from
Databricks runtime

Photon Engine

Databricks Runtime Light

Runtime option for only jobs not requiring advanced features

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Termination

- Terminates the cluster after X minutes of inactivity
- Default value for Single Node and Standard clusters is 120 minutes
- Users can specify a value between 10 and 10000 mins as the duration

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Scaling

Auto Scaling

- User specifies the min and max work nodes
- Auto scales between min and max based on the workload
- Not recommended for streaming workloads

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Scaling

Cluster VM Type/ Size

Memory Optimized

Compute Optimized

Storage Optimized

General Purpose

GPU Accelerated

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Scaling

Cluster VM Type/ Size

Cluster Policy

Cluster Configuration

Policy

Unrestricted

Multi node Single node

Access mode **Single user access**

Single user Ramesh Retnasamy (az.adm1...

Performance

Databricks runtime version

Runtime: 11.3 LTS (Scala 2.12, Spark 3.3.0)

Use Photon Acceleration

Worker type **Min workers** **Max workers**

Standard_DS3_v2 14 GB Memory, 4 Cores 2 8 Spot instances

Driver type

Same as worker 14 GB Memory, 4 Cores

Enable autoscaling

Terminate after 120 minutes of inactivity

Cluster Configuration

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Scaling

Cluster VM Type/ Size

Cluster Policy

Cluster Configuration 

Policy 

Personal Compute 

Single user access 

Ramesh Retnasamy (az.adm1@outlook.com) 

Performance

Databricks runtime version 

Runtime: 11.3 LTS ML (Scala 2.12, Spark 3.3.0) 

Node type 

Standard_DS3_v2 14 GB Memory, 4 Cores 

Terminate after minutes of inactivity 



Cluster Configuration

Single/ Multi Node

Access Mode

Simplifies the user interface

Databricks Runtime

Auto Termination

Enables standard users to create clusters

Auto Scaling

Achieves cost control

Cluster VM Type/ Size

Cluster Policy

Only available on premium tier

Creating Databricks Cluster

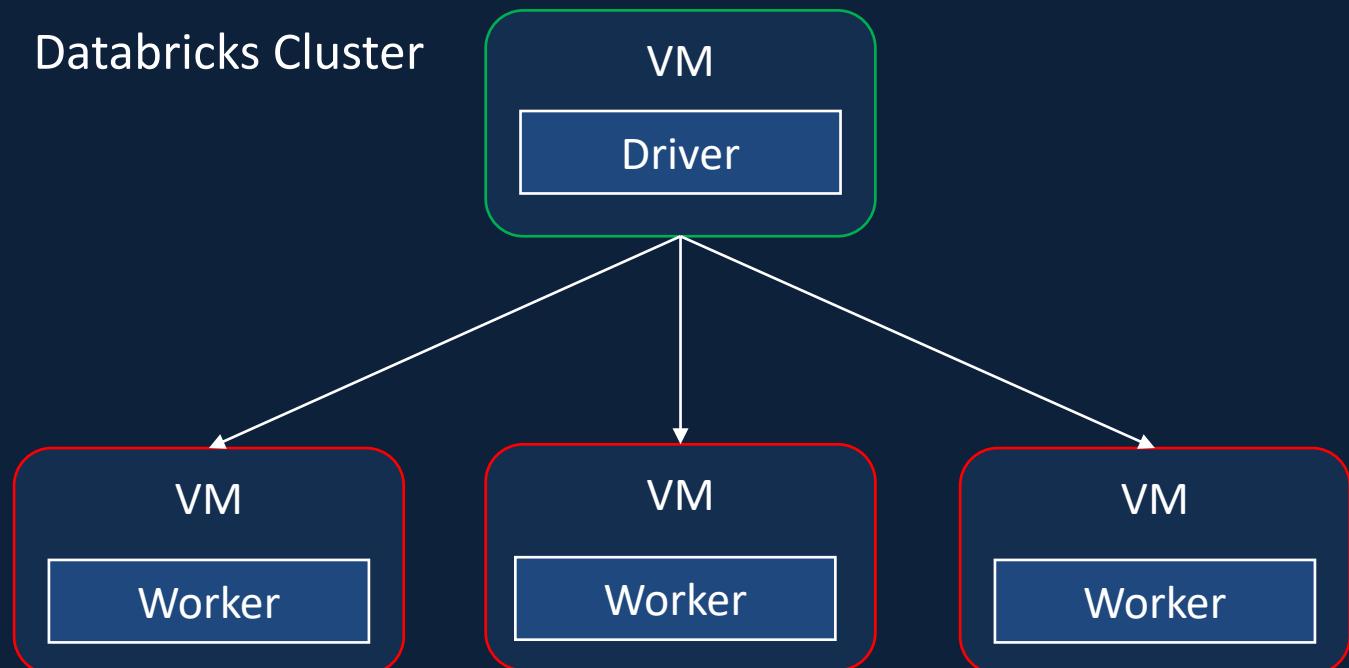


Azure Databricks Cluster Pricing

\$

Azure Databricks Pricing Factors

Databricks Cluster



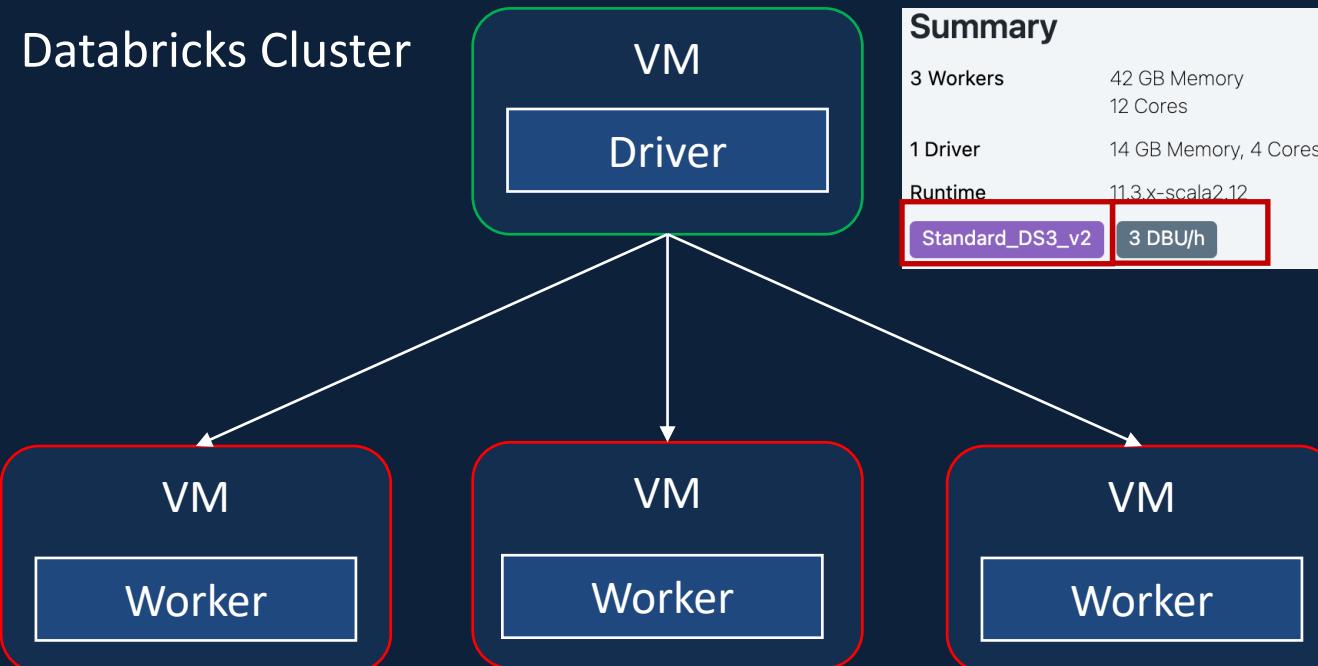
Workload (All Purpose/ Jobs/ SQL/ Photon)

Tier (Premium/ Standard)

VM Type (General Purpose/ GPU/ Optimized)

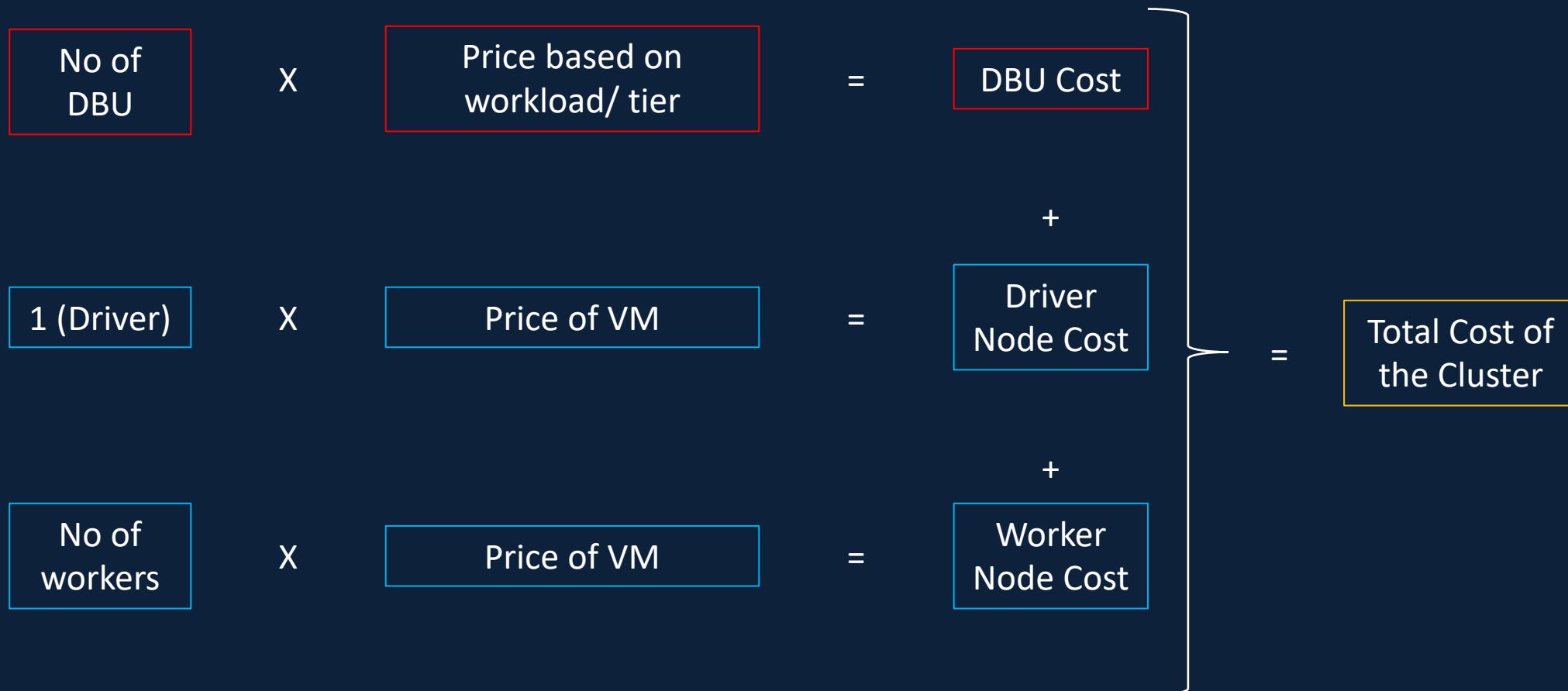
Purchase Plan (Pay As You Go/ Pre-Purchase)

Azure Databricks Pricing Calculation



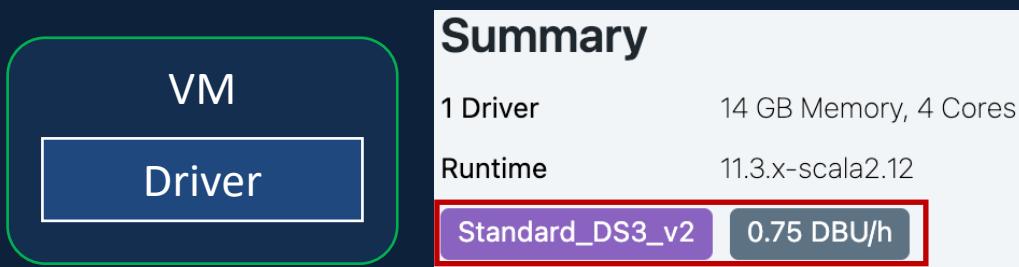
A Databricks Unit (DBU) is a normalized unit of processing power on the Databricks Lakehouse Platform used for measurement and pricing purposes

Azure Databricks Pricing Calculation



Azure Databricks Pricing Calculation

Single Node Cluster



Workload - All Purpose

Pricing Tier - Premium

VM Type - General Purpose Standard DS3_V2

Purchase Plan - Pay As You Go

Azure Databricks Pricing Calculation

Workload	DBU prices—standard tier	DBU prices—premium tier
All-Purpose Compute**	\$0.40/DBU-hour	\$0.55/DBU-hour
Jobs Compute**	\$0.15/DBU-hour	\$0.30/DBU-hour
Jobs Light Compute	\$0.07/DBU-hour	\$0.22/DBU-hour
SQL Compute	-	\$0.22/DBU-hour
SQL Pro Compute	-	\$0.37/DBU-hour
Serverless SQL	-	\$0.475/DBU-hour

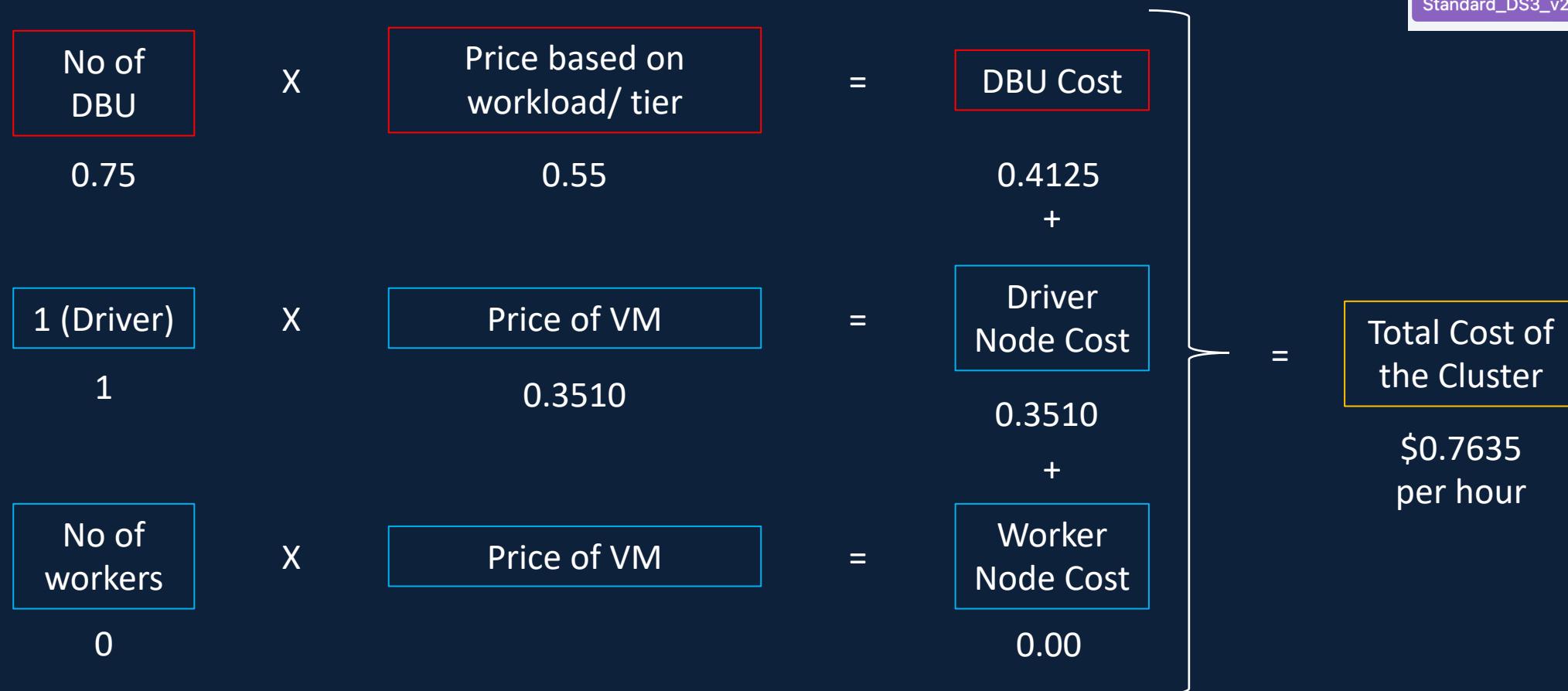
<https://azure.microsoft.com/en-gb/pricing/details/databricks/>

Azure Databricks Pricing Calculation

Instance	vCPU(\$)	RAM	Temporary storage	Pay as you go	1 year savings plan	3 year savings plan	Spot	Add to estimate
DS1 v2	1	3.5 GiB	7 GiB	\$0.0878/hour	\$0.0760/hour ~13% savings	\$0.0600/hour ~31% savings	\$0.0105/hour ~87% savings	<button>+</button>
DS2 v2	2	7 GiB	14 GiB	\$0.1760/hour	\$0.1524/hour ~13% savings	\$0.1203/hour ~31% savings	\$0.0211/hour ~87% savings	<button>+</button>
DS3 v2	4	14 GiB	28 GiB	\$0.3510/hour	\$0.3040/hour ~13% savings	\$0.2399/hour ~31% savings	\$0.0422/hour ~87% savings	<button>+</button>
DS4 v2	8	28 GiB	56 GiB	\$0.7020/hour	\$0.6080/hour ~13% savings	\$0.4798/hour ~31% savings	\$0.0843/hour ~87% savings	<button>+</button>
DS5 v2	16	56 GiB	112 GiB	\$1.4050/hour	\$1.2169/hour ~13% savings	\$0.9603/hour ~31% savings	\$0.1687/hour ~87% savings	<button>+</button>

<https://azure.microsoft.com/en-gb/pricing/details/virtual-machines/linux/#pricing>

Azure Databricks Pricing Calculation



Summary	
1 Driver	14 GB Memory, 4 Cores
Runtime	11.3.x-scala2.12
Standard_DS3_v2	0.75 DBU/h

Estimated cost for doing the course

Azure Data Lake Storage

Azure Data Factory

Azure Databricks Job Cluster

Azure Databricks Cluster Pool

Azure Databricks All Purpose Cluster

\$0.76 per hour for a small single node cluster on premium tier

Depends on the number of hours in use

Past student experience – 20 to 30 hours to complete the course

Past *Pay As You Go* student experience – \$15 to \$25

Within the credit offered by Free / Student Subscription

Cost Control

Service	Action to be taken
Azure Data Lake Storage	None – Cost Negligible
Azure Data Factory	None - Billed only for execution of the pipeline
Azure Databricks Job Cluster	None - Destroyed once the job completes
Azure Databricks Cluster Pool	Delete the cluster pool at end of the lesson
Azure Databricks All Purpose Cluster	Set Auto Termination to 20 minutes

Set budget alerts on your subscription

Cluster Pools

Single/ Multi Node

Access Mode

Databricks Runtime

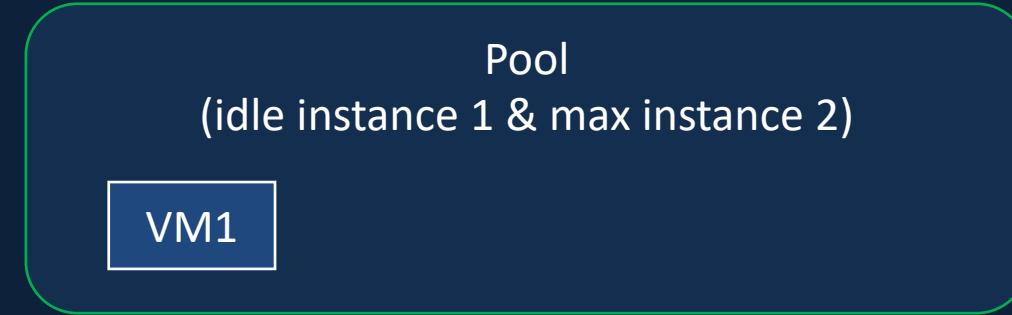
Auto Termination

Auto Scaling

Cluster VM Type/ Size

Cluster Policy

Cluster Pool



Cluster Pools

Single/ Multi Node

Access Mode

Databricks Runtime

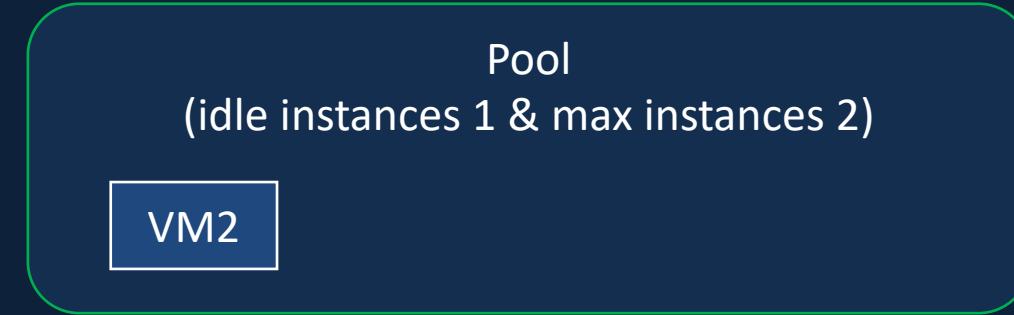
Auto Termination

Auto Scaling

Cluster VM Type/ Size

Cluster Policy

Cluster Pool



Cluster Pools

Single/ Multi Node

Access Mode

Databricks Runtime

Auto Termination

Auto Scaling

Cluster VM Type/ Size

Cluster Policy

Cluster Pool

Pool
(idle instances 1 & max instances 2)

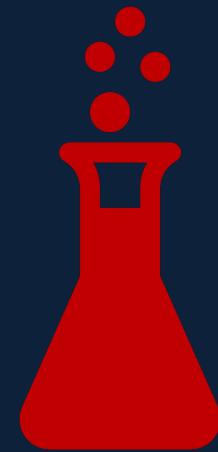
Cluster 1

VM1

Cluster 2

VM2

Creating Cluster Pool



Cluster Policy



Cluster Policy



Hide Attributes

Fix Values

Set Default Values

Simple User Interface

Achieve Cost Control

Standardize Cluster Configs

Empowers standard users

Cluster Policy



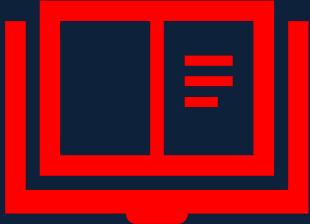
Public Preview (December 2022)

Only available on Premium Tier

Create Cluster Policy



Databricks Notebooks



What's a notebook

Creating a notebook

Magic Commands

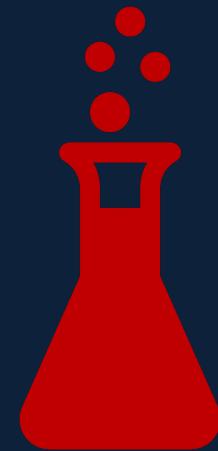
Databricks Utilities

Import Project Solution Notebooks

Creating Notebooks



Magic Commands



Databricks Utilities



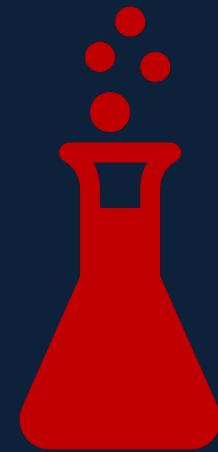
File System Utilities

Secrets Utilities

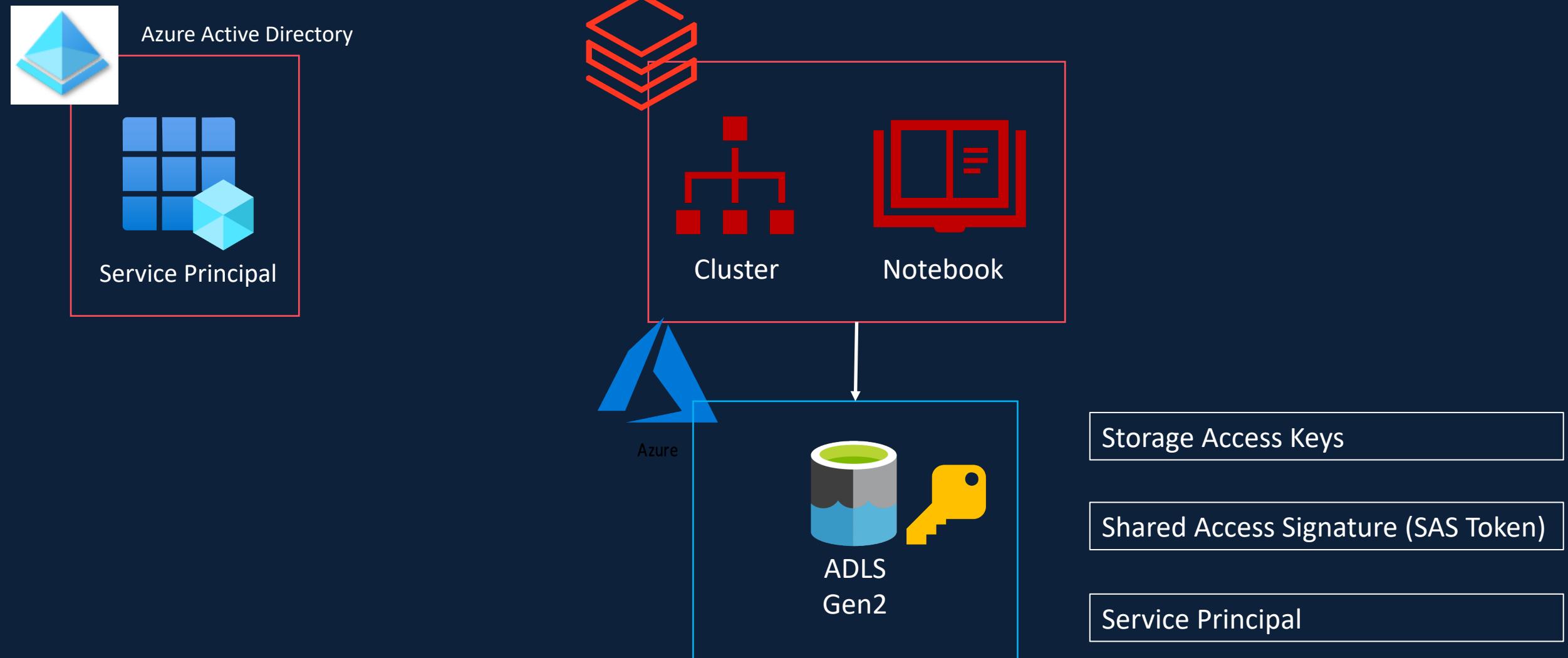
Widget Utilities

Notebook Workflow Utilities

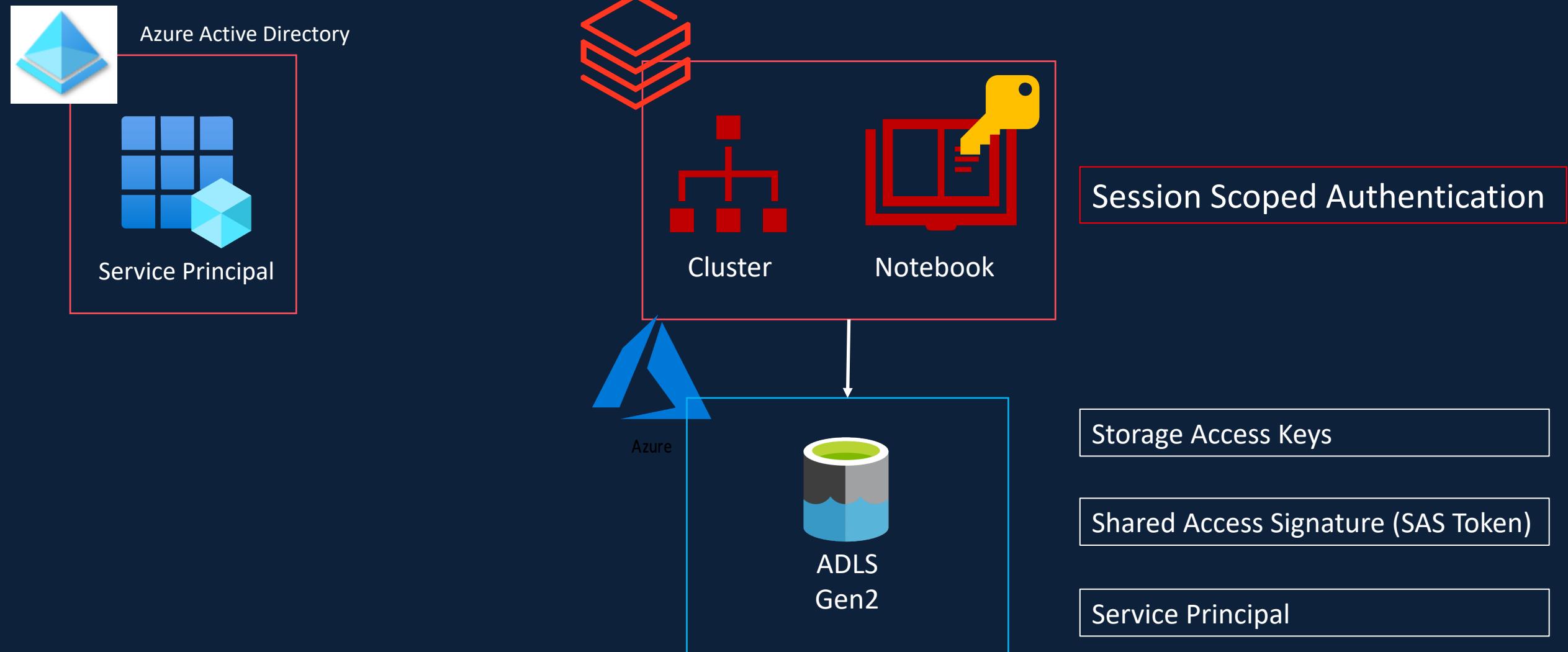
Databricks Utilities



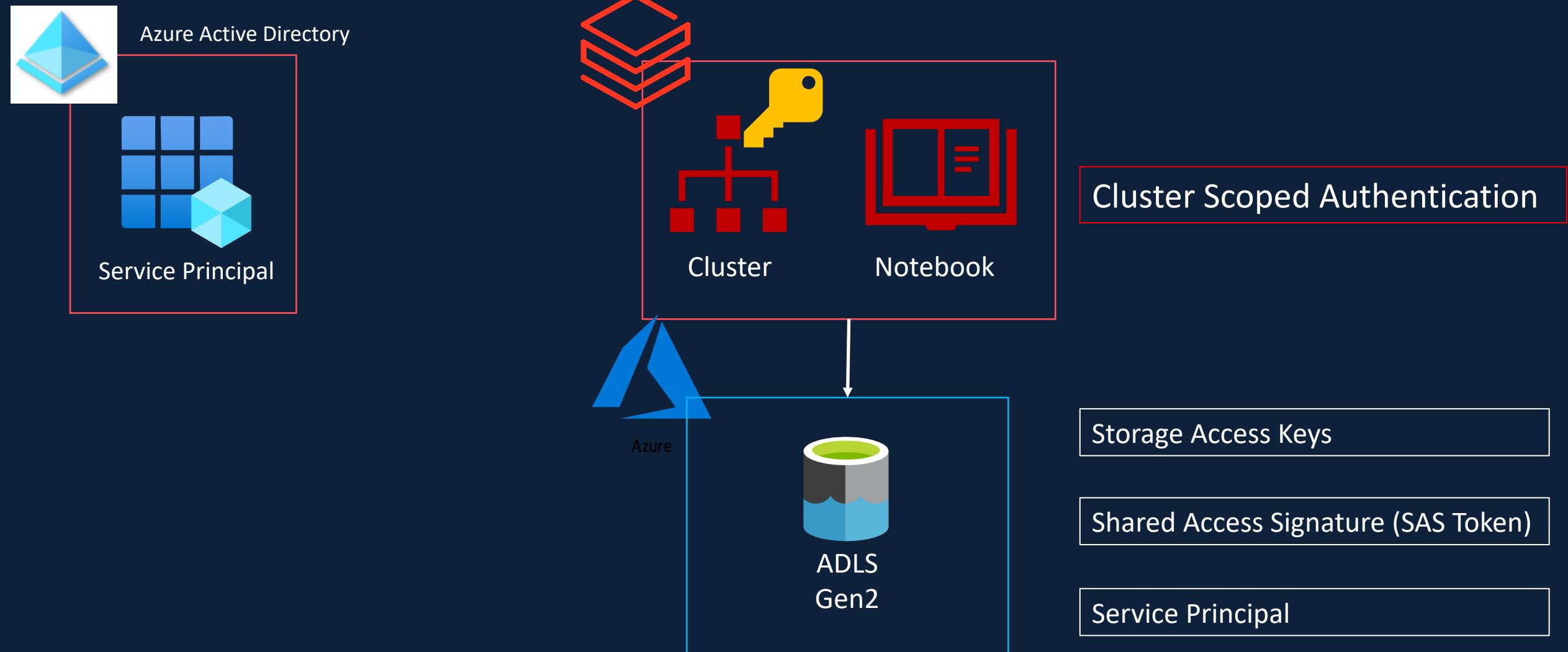
Access Azure Data Lake - Section Overview



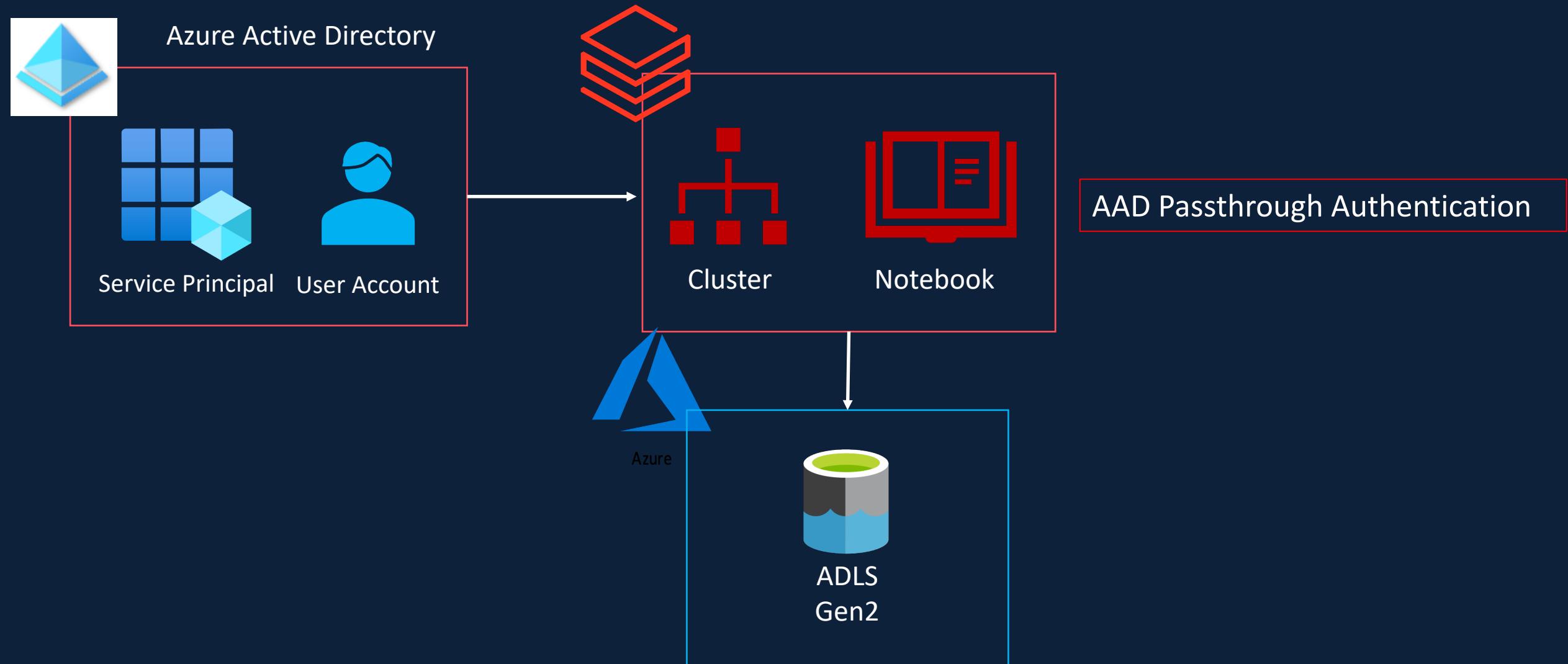
Access Azure Data Lake - Section Overview



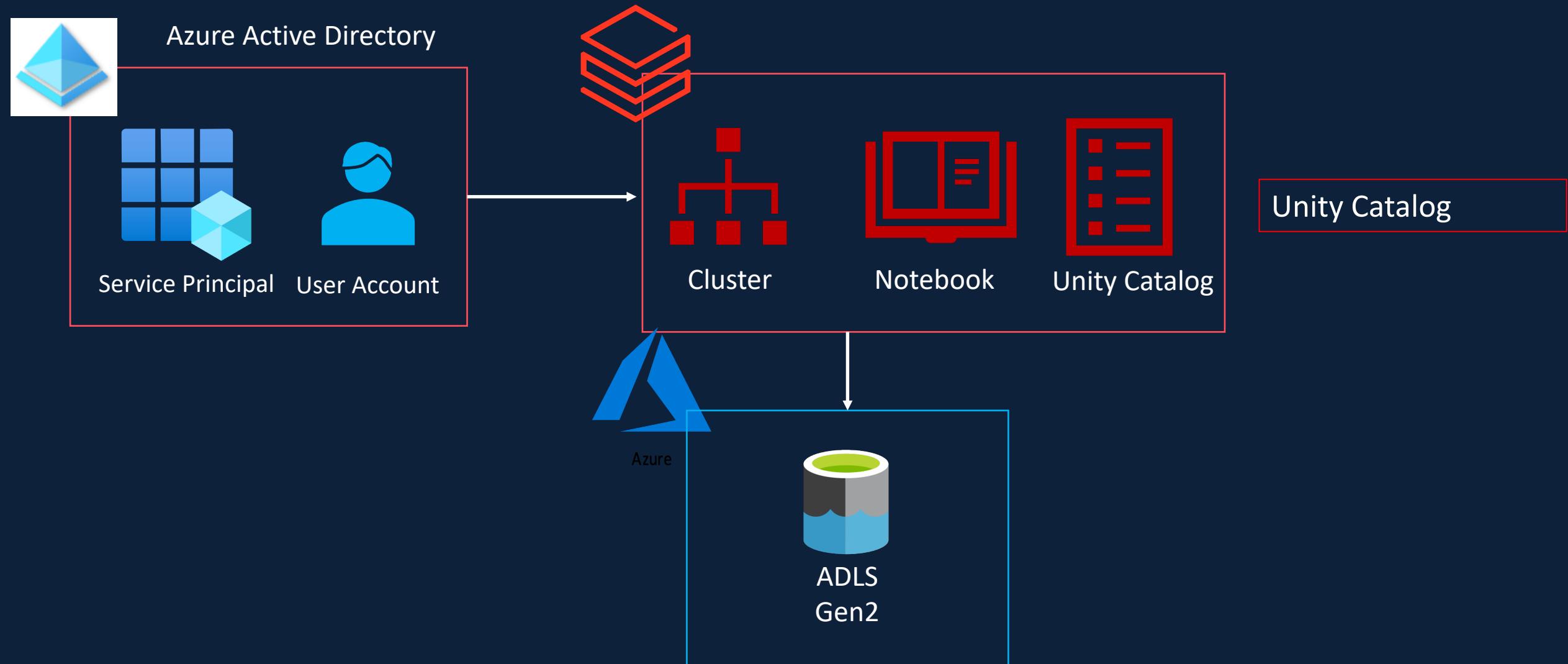
Access Azure Data Lake - Section Overview



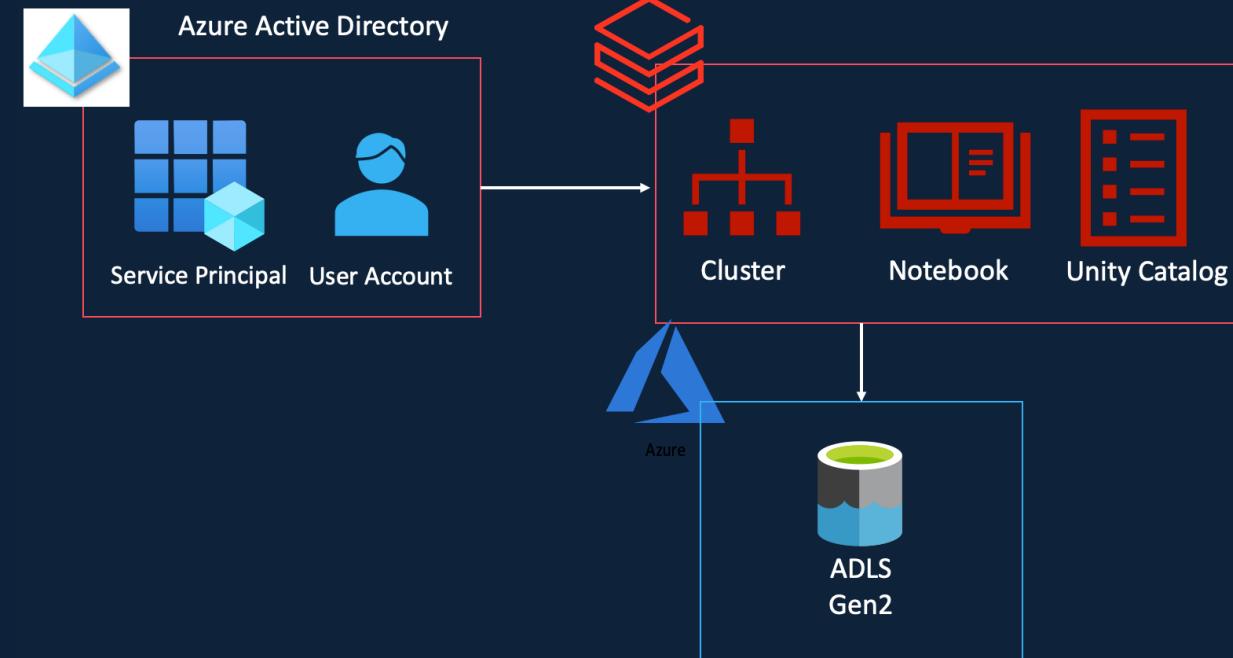
Access Azure Data Lake - Section Overview



Access Azure Data Lake - Section Overview



Access Azure Data Lake - Section Overview



Create Azure Data Lake Gen2 Storage

Access Data Lake using Access Keys

Access Data Lake using SAS Token

Access Data Lake using Service Principal

Using Cluster Scoped Authentication

Access Data Lake using AAD Credential Pass-through

Recommended approach for the course

Create Azure Data Lake Gen2 Storage (ADLS Gen2)



Access Azure Data Lake Gen2 using Access Keys



Authenticate Using Access Keys



ADLS Gen2

Each storage account comes with 2 keys

Gives full Access to the storage account

Keys can be rotated (regenerated)

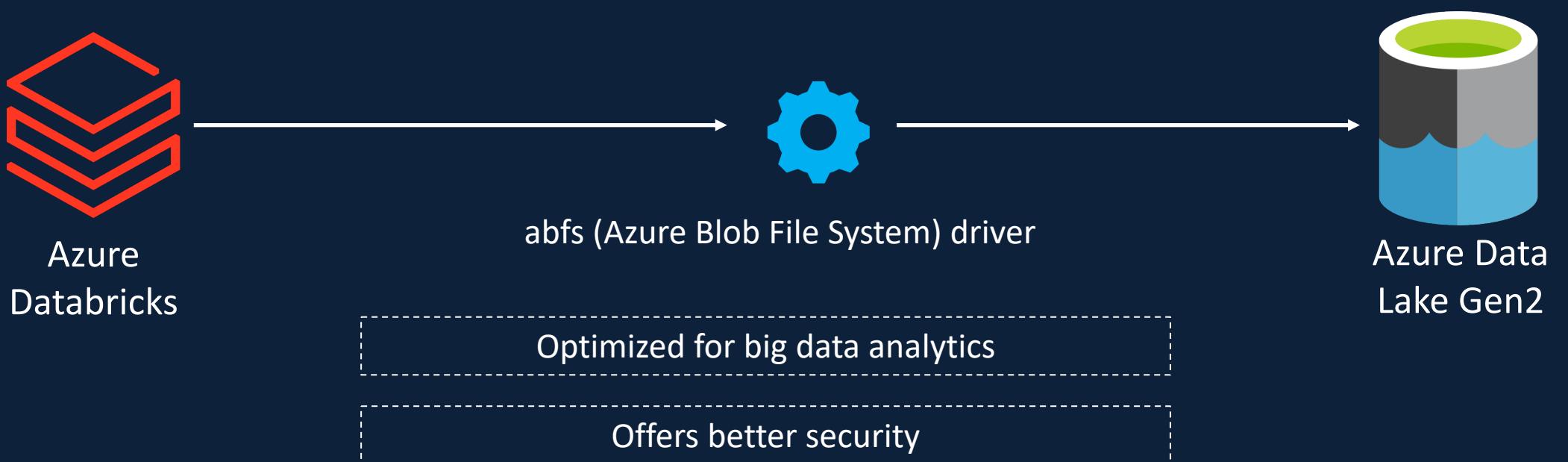
Access Keys – Spark Configuration



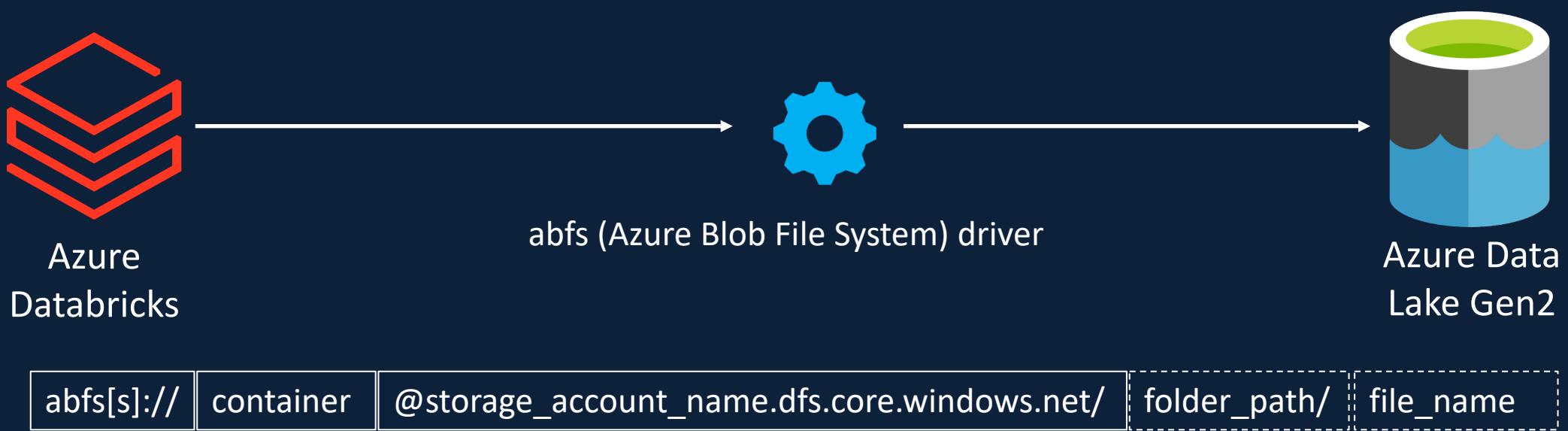
```
spark.conf.set("fs.azure.account.key.<storage-account>.dfs.core.windows.net", "<access key>")
```

```
spark.conf.set(  
    "fs.azure.account.key.formula1dl.dfs.core.windows.net",  
    "30RoyW+laxV39N0JZ7XWRSS0imUGp2lKdE65nRbHrJ9UHc1fqLyJN+j+Qunhev+YL8+CwPLenWn+ASTg8bfJg=")
```

Access Keys – abfs driver



Access Keys – abfs driver

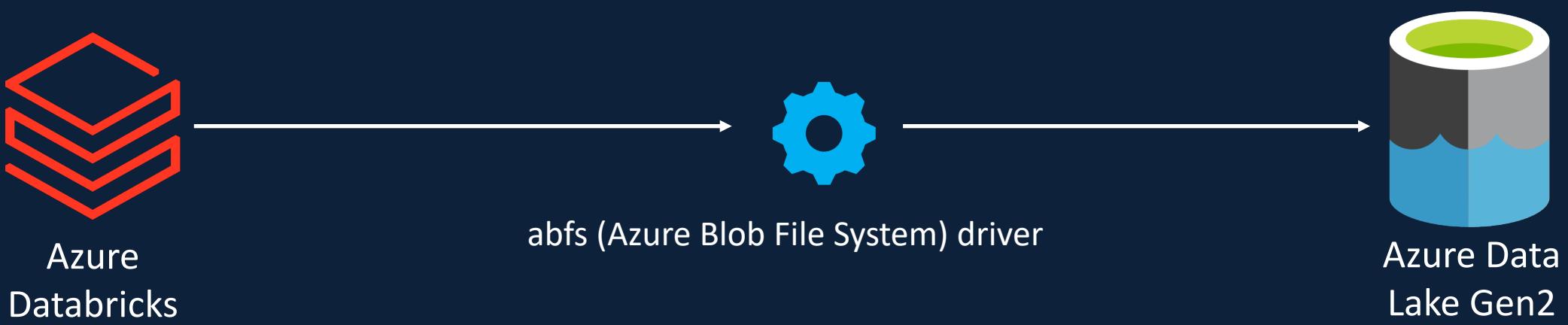


abfss://demo@formula1dl.dfs.core.windows.net/

abfss://demo@formula1dl.dfs.core.windows.net/test/

abfss://demo@formula1dl.dfs.core.windows.net/test/circuits.csv

Authenticate Using Access Keys



```
spark.conf.set("fs.azure.account.key.<storage-account>.dfs.core.windows.net", "<access key>")
```

```
dbutils.fs.ls("abfss://demo@formula1dl.dfs.core.windows.net/")
```

Access Azure Data Lake using Shared Access Signature (SAS Token)



Shared Access Signature



Provides fine grained access to the storage

Restrict access to specific resource types/ services

Allow specific permissions

Restrict access to specific time period

Limit access to specific IP addresses

Recommended access pattern for external clients

Shared Access Signature



```
spark.conf.set("fs.azure.account.auth.type.<storage-account>.dfs.core.windows.net",  
"SAS")
```

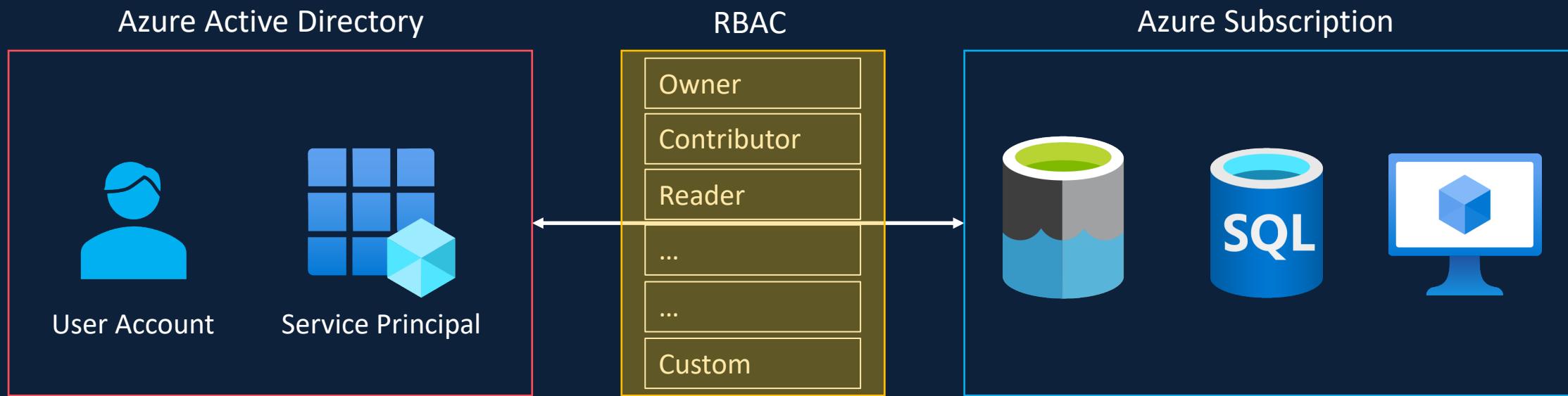
```
spark.conf.set("fs.azure.sas.token.provider.type.<storage-account>.dfs.core.windows.net",  
"org.apache.hadoop.fs.azurebfs.sas.FixedSASTokenProvider")
```

```
spark.conf.set("fs.azure.sas.fixed.token.<storage-account>.dfs.core.windows.net", "<token>")
```

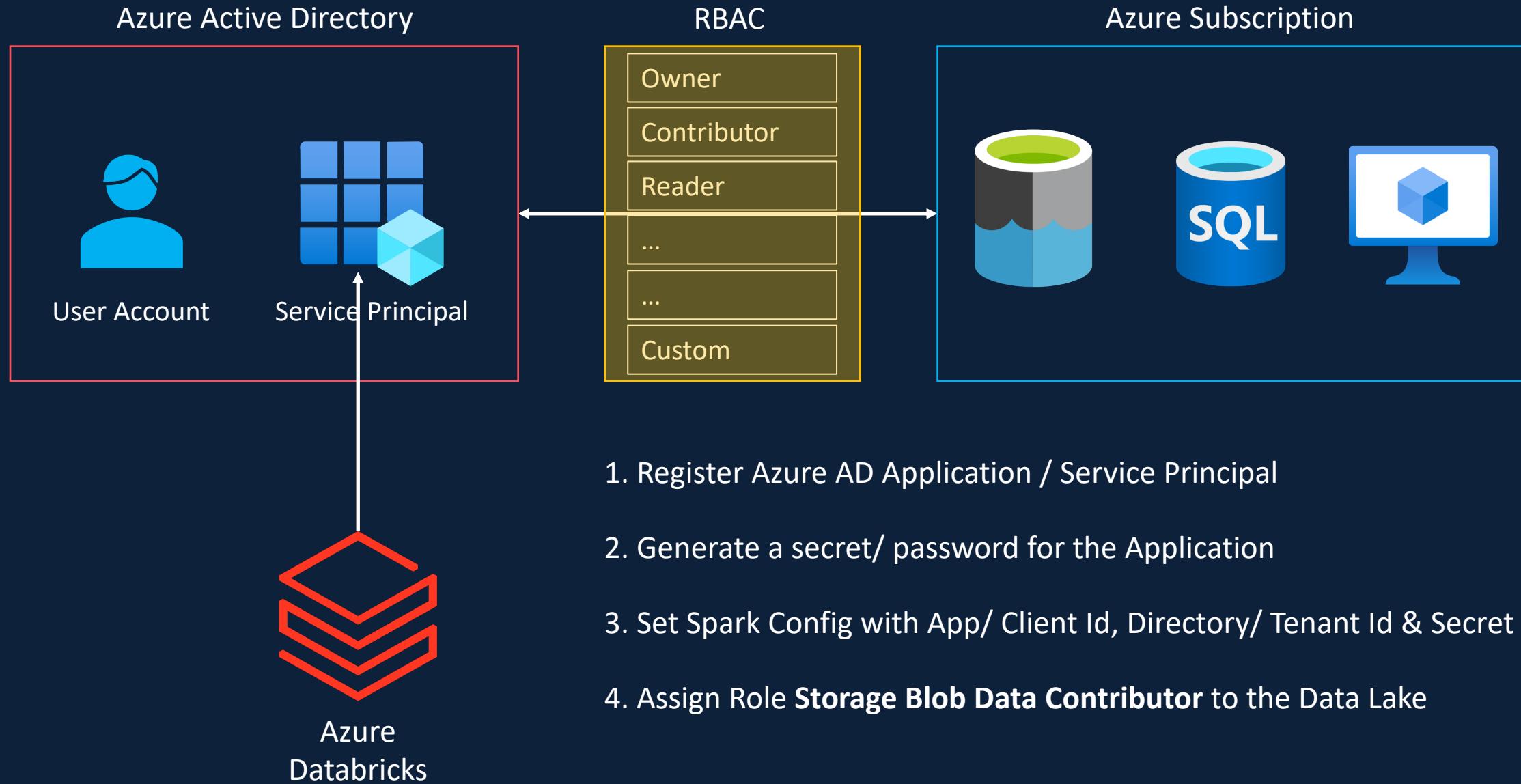
Access Azure Data Lake using Service Principal



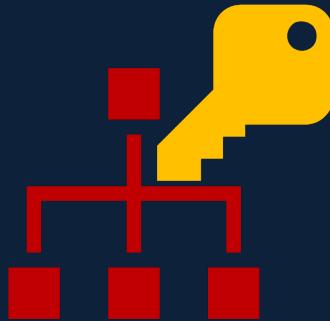
Service Principal



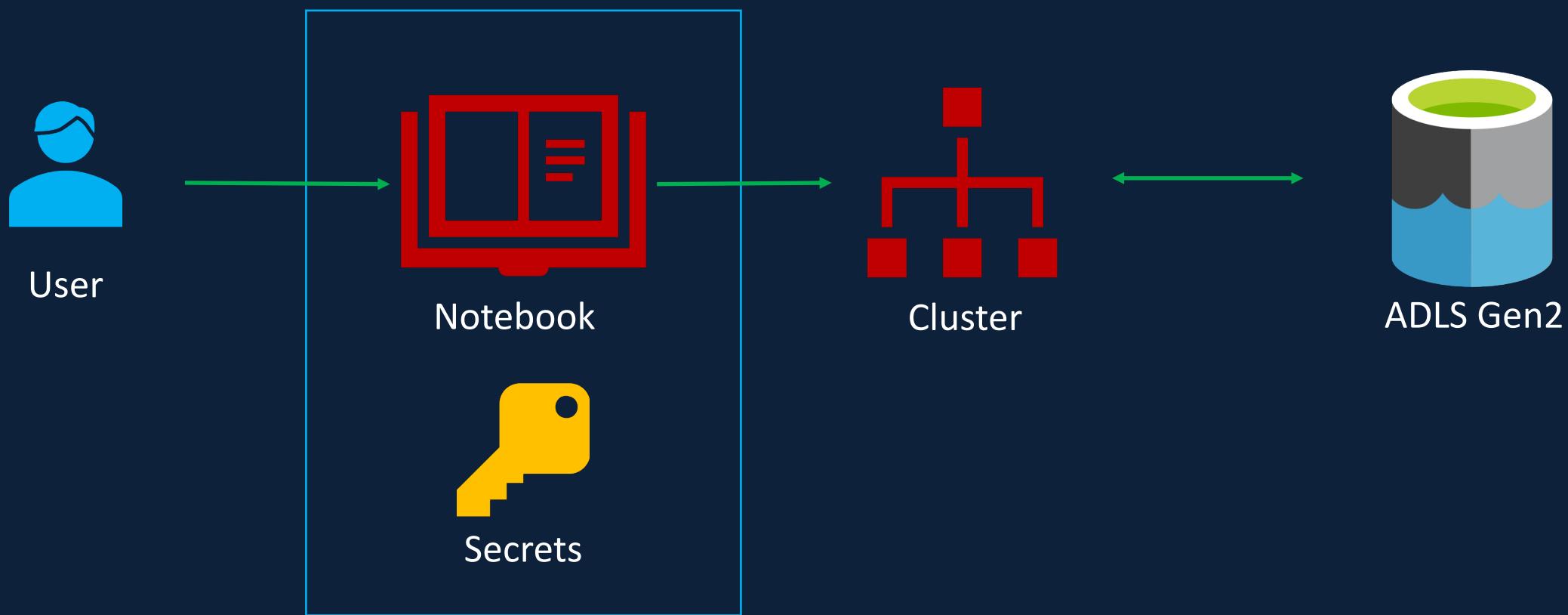
Service Principal



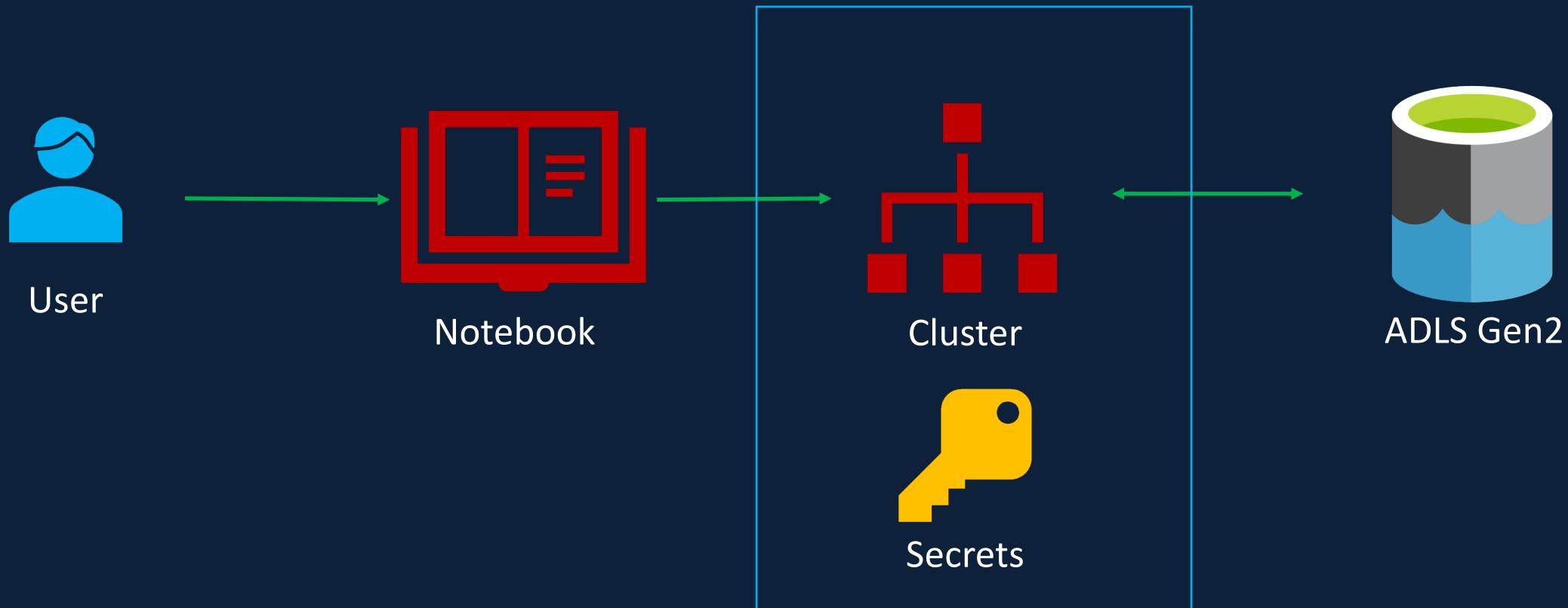
Cluster Scoped Authentication



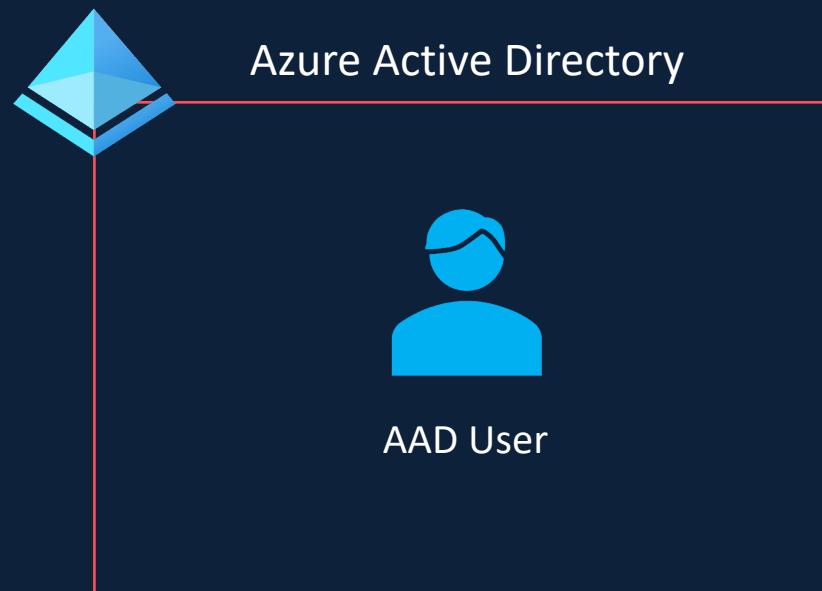
Session Scoped Authentication



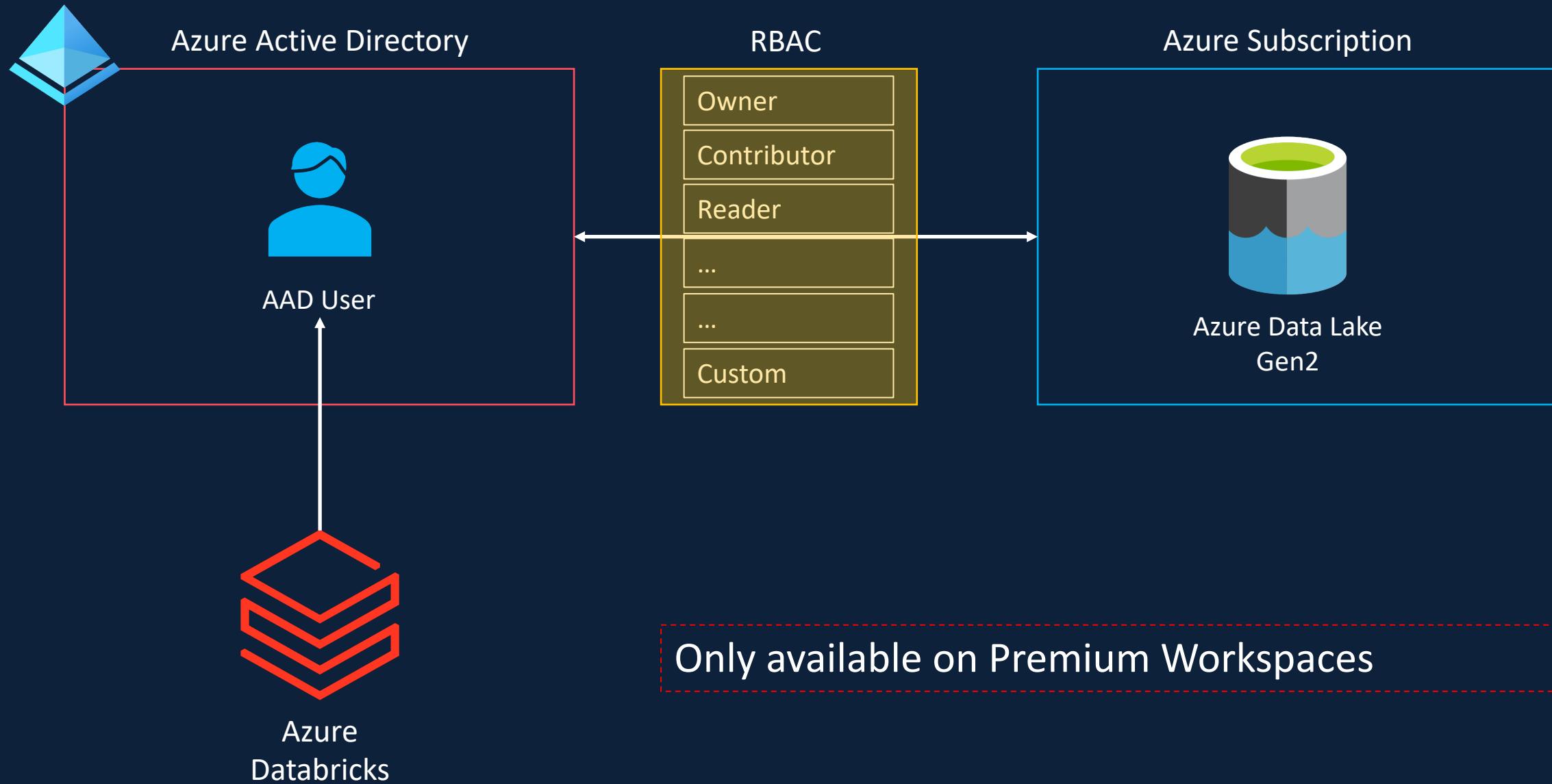
Cluster Scoped Authentication



AAD Credential Passthrough



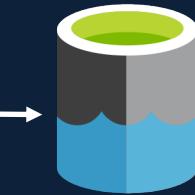
AAD Credential Passthrough



Recommended Access Pattern For The Course

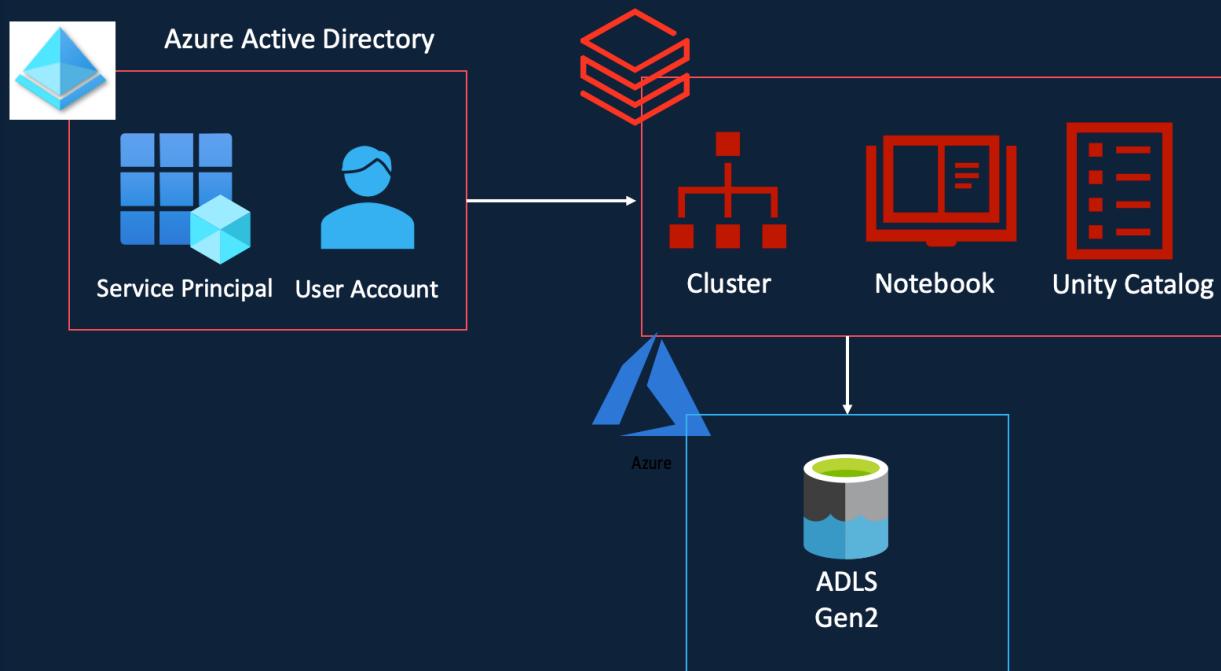


Azure
Databricks



Azure Data
Lake Gen2

Recommended Access Pattern



Access Data Lake using Access Keys

Access Data Lake using SAS Token

Access Data Lake using Service Principal

Access Data Lake using Cluster Scoped Authentication

Access Data Lake using AAD Credential Pass-through

Securing Credentials & Secrets

Databricks Mounts

Recommended Access Pattern

Student Subscription

Company Subscription
without Access to AAD

Free Subscription

Pay-as-you-go
Subscription

Any other Subscription
with Access to AAD



Using Cluster Scoped Authentication (via
Access Keys)



Using Service Principal

Securing Secrets



Databricks Secret Scope

Azure Key Vault

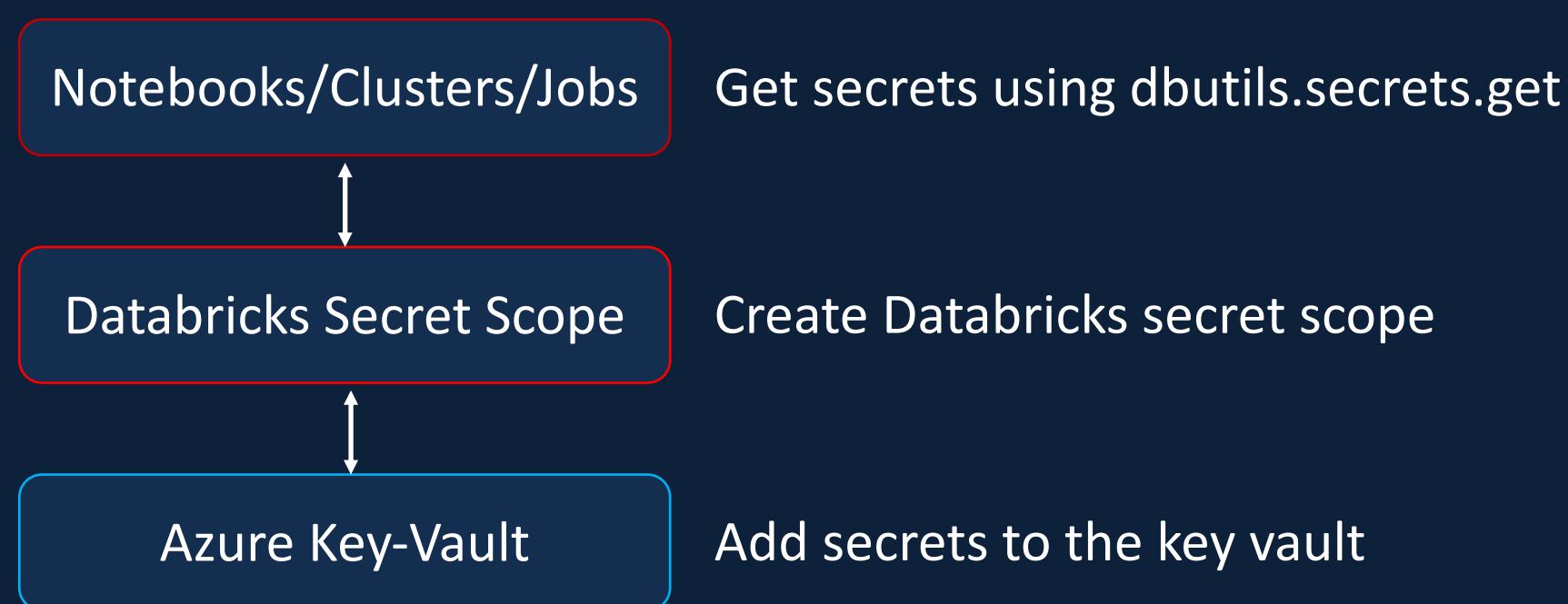
Securing Secrets – Section Overview

Secret scopes help store the credentials securely and reference them in notebooks, clusters and jobs when required

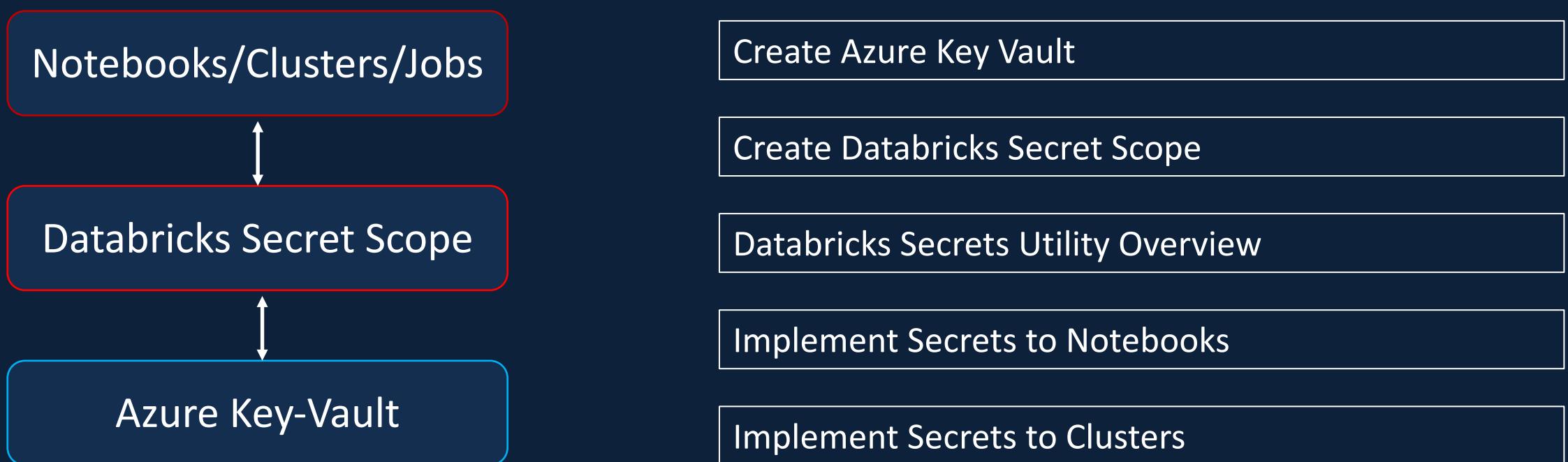
Databricks backed Secret Scope

Azure Key-vault backed Secret Scope

Securing Secrets – Section Overview



Securing Secrets – Section Overview



Creating Azure Key Vault



Creating Secret Scope

Databricks Secrets Utility

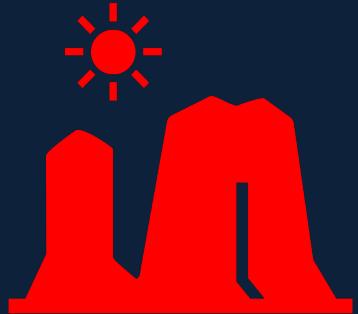
(dbutils.secrets)

Implement Secrets Utility in Databricks Notebooks

Implement Secrets Utility in Databricks Notebooks (Assignment)

Implement Secrets Utility in Databricks Clusters

Databricks Mounts

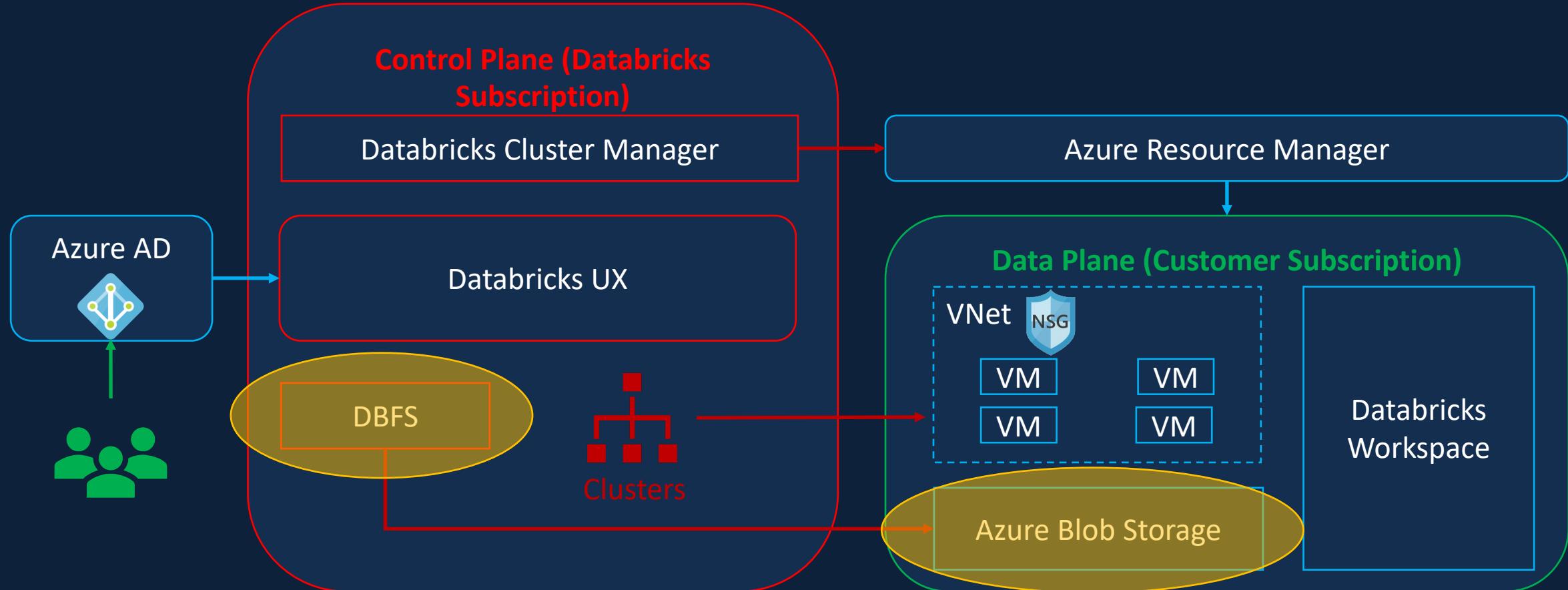


What is Databricks File System (DBFS)

What are Databricks mounts

Mount ADLS container to Databricks

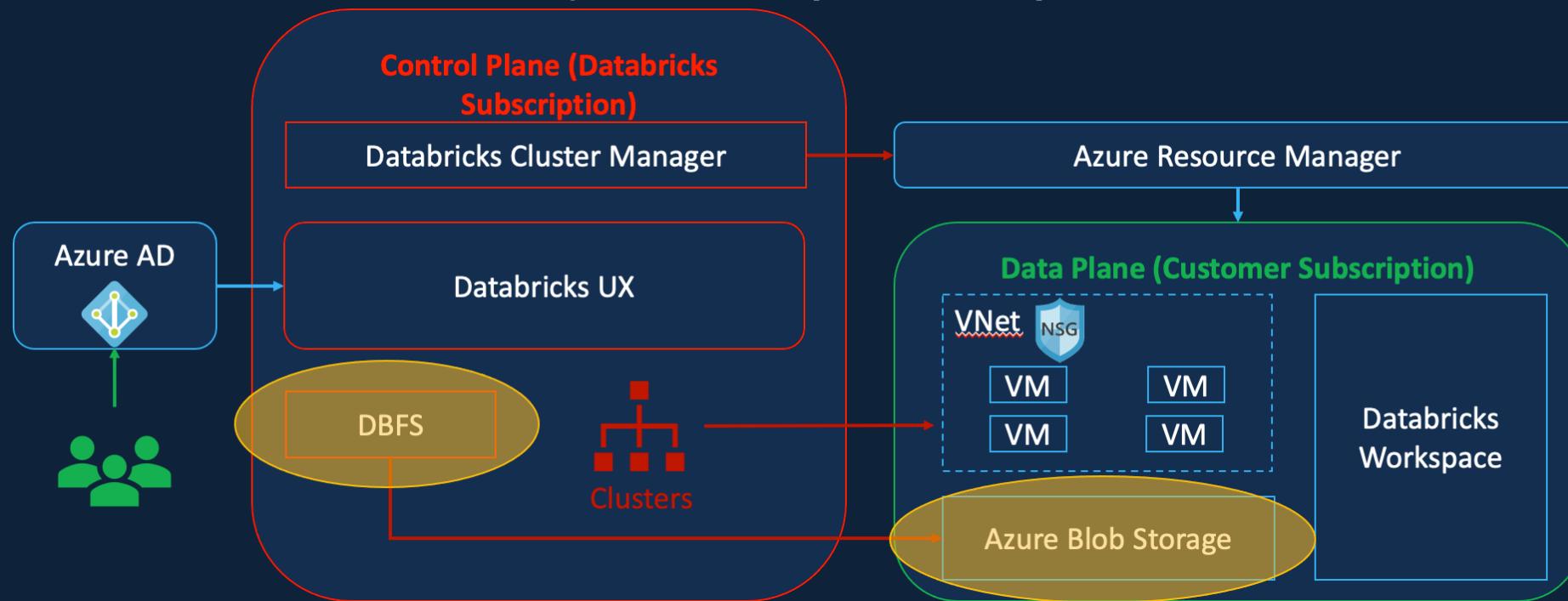
Databricks File System (DBFS)



Databricks File System (DBFS) is a distributed file system mounted on the Databricks workspace

DBFS Root is the default storage for a Databricks workspace created during workspace deployment

DBFS Root



Backed by the default Azure Blob Storage

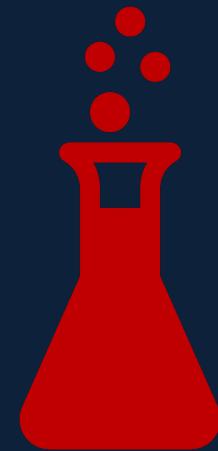
Can be accessed via the web UI

Query results are stored here

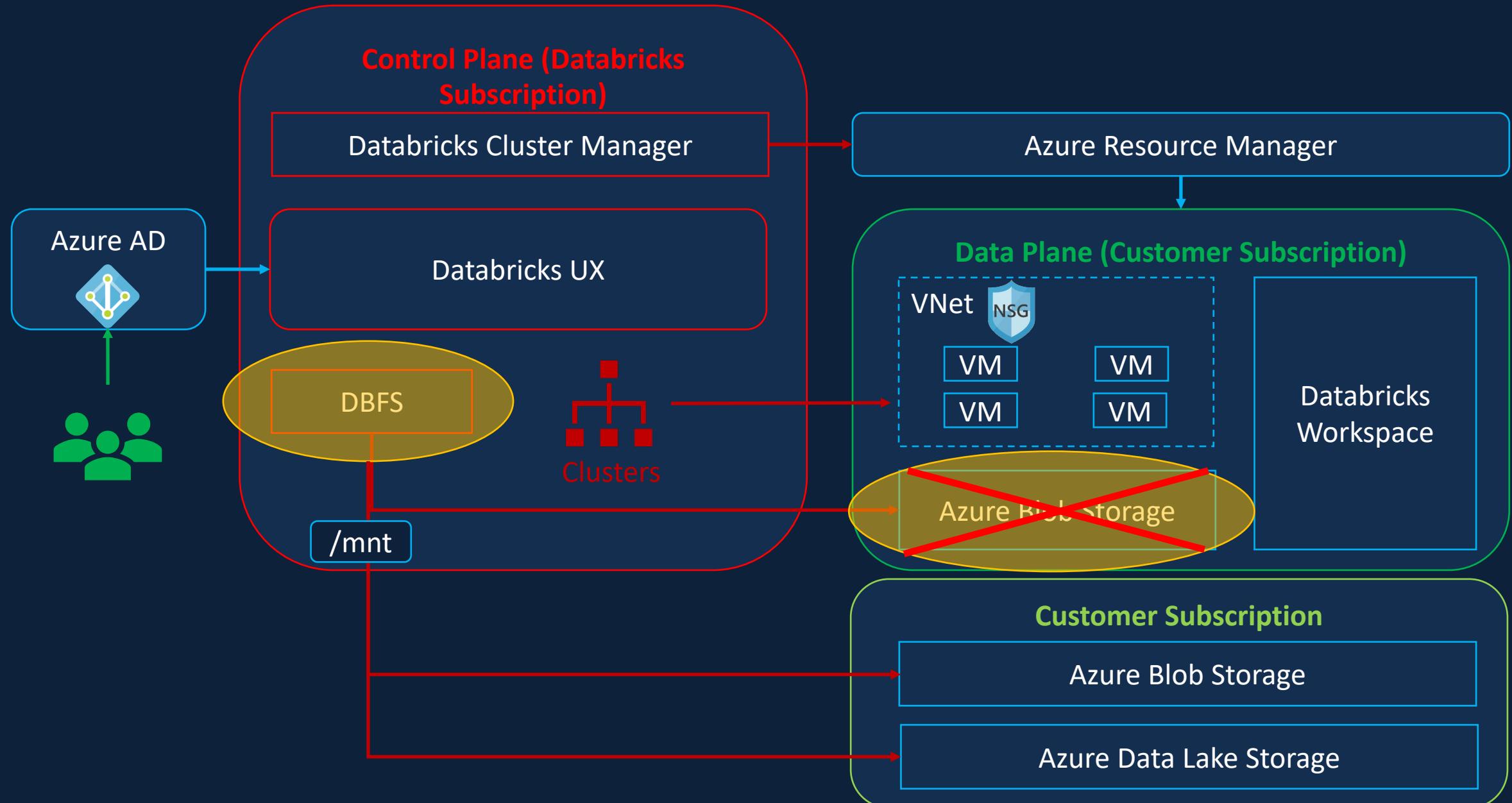
Default storage location for managed tables

Default storage location, but not recommended for user data

DBFS Root Demo



Databricks Mounts



Databricks Mounts

Benefits

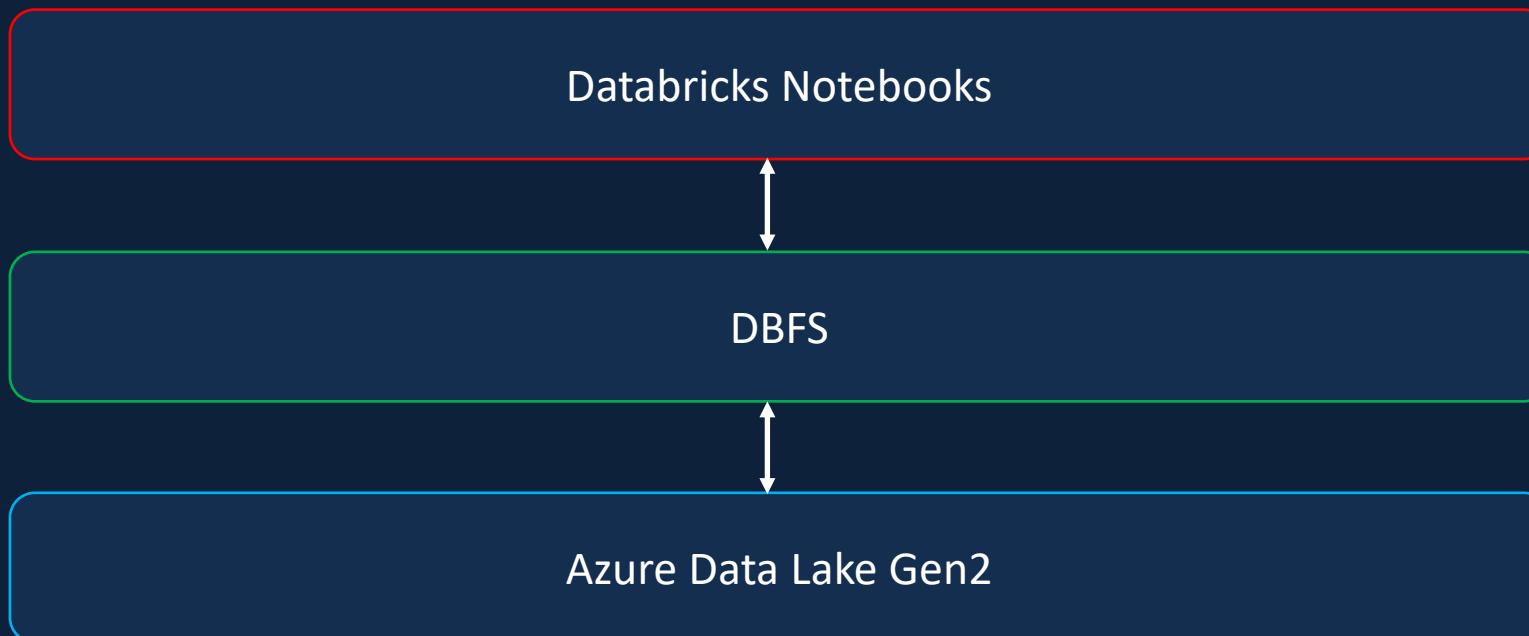
Access data without requiring credentials

Access files using file semantics rather than storage URLs (e.g. /mnt/storage1)

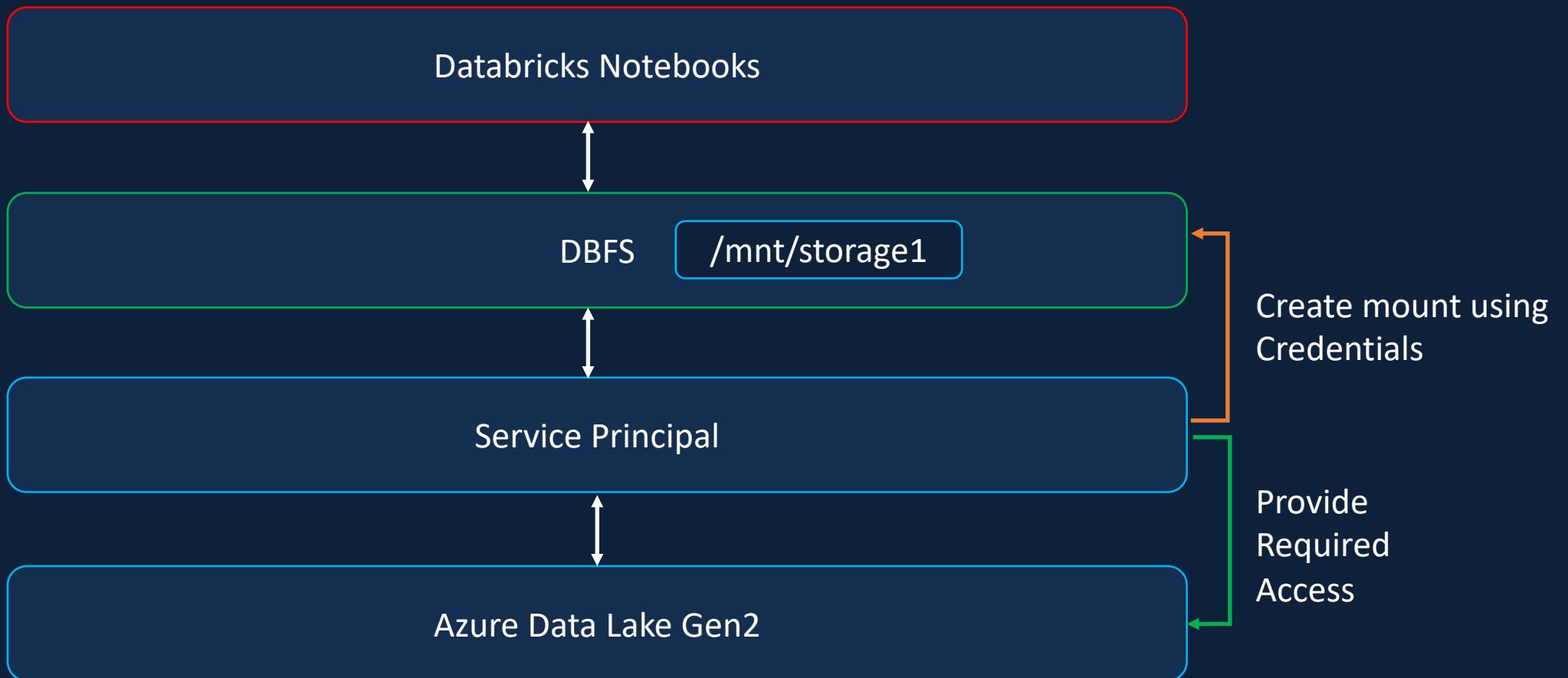
Stores files to object storage (e.g. Azure Blob), so you get all the benefits from Azure

Recommended solution for access Azure Storage until the introduction
of Unity Catalog (Generally Available from end of 2022)

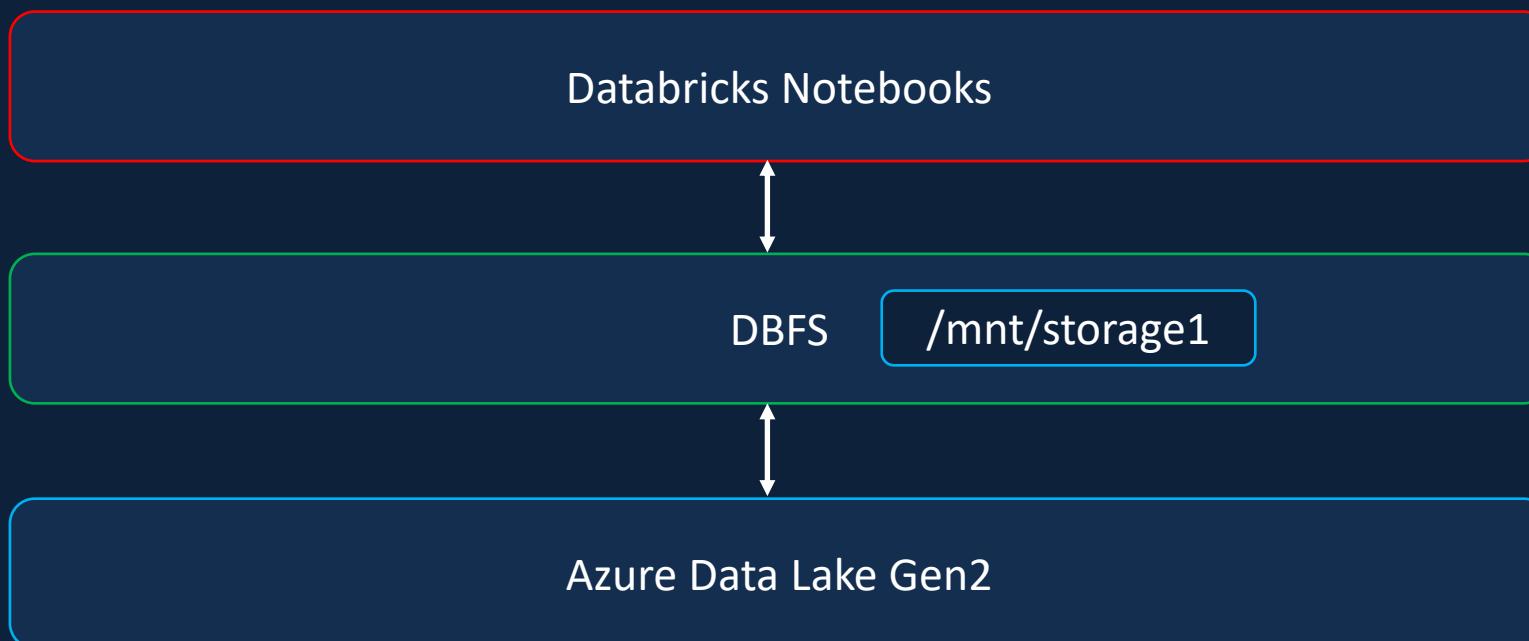
Databricks Mounts



Databricks Mounts



Databricks Mounts



Mounting Azure Data Lake Storage Gen2



Project Overview



What is formula1

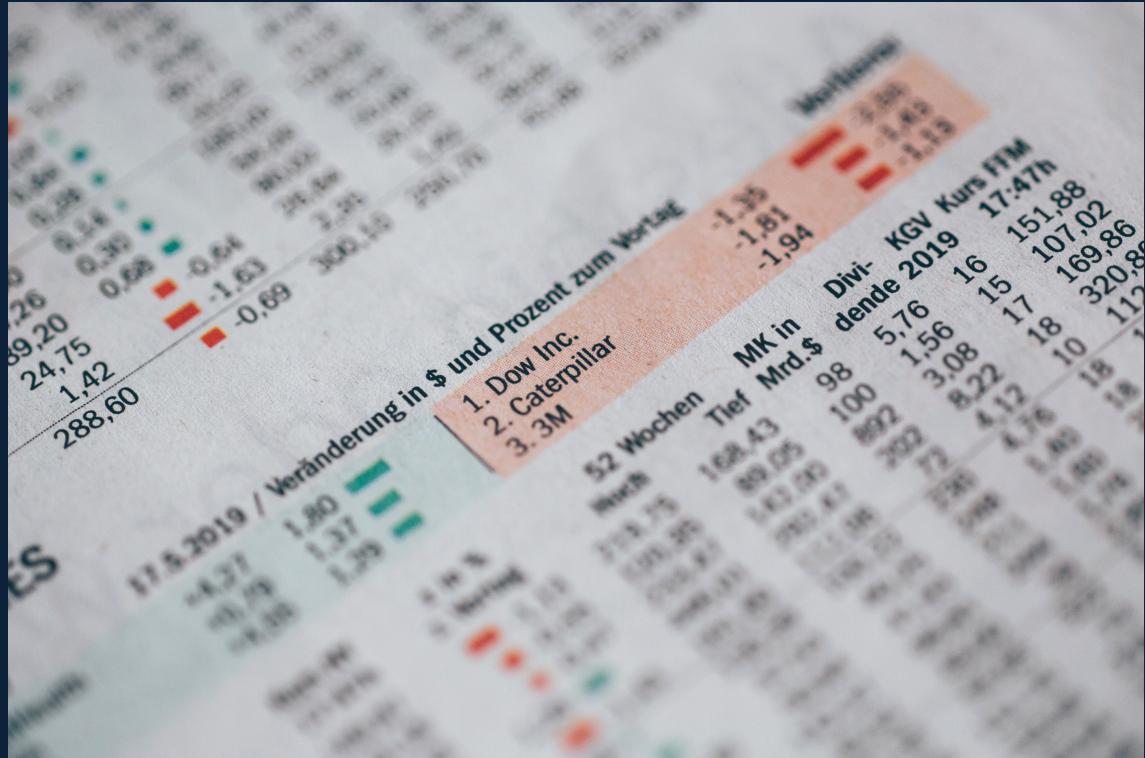
Formula1 data source & datasets

Prepare the data for the project

Project Requirements

Solution Architecture

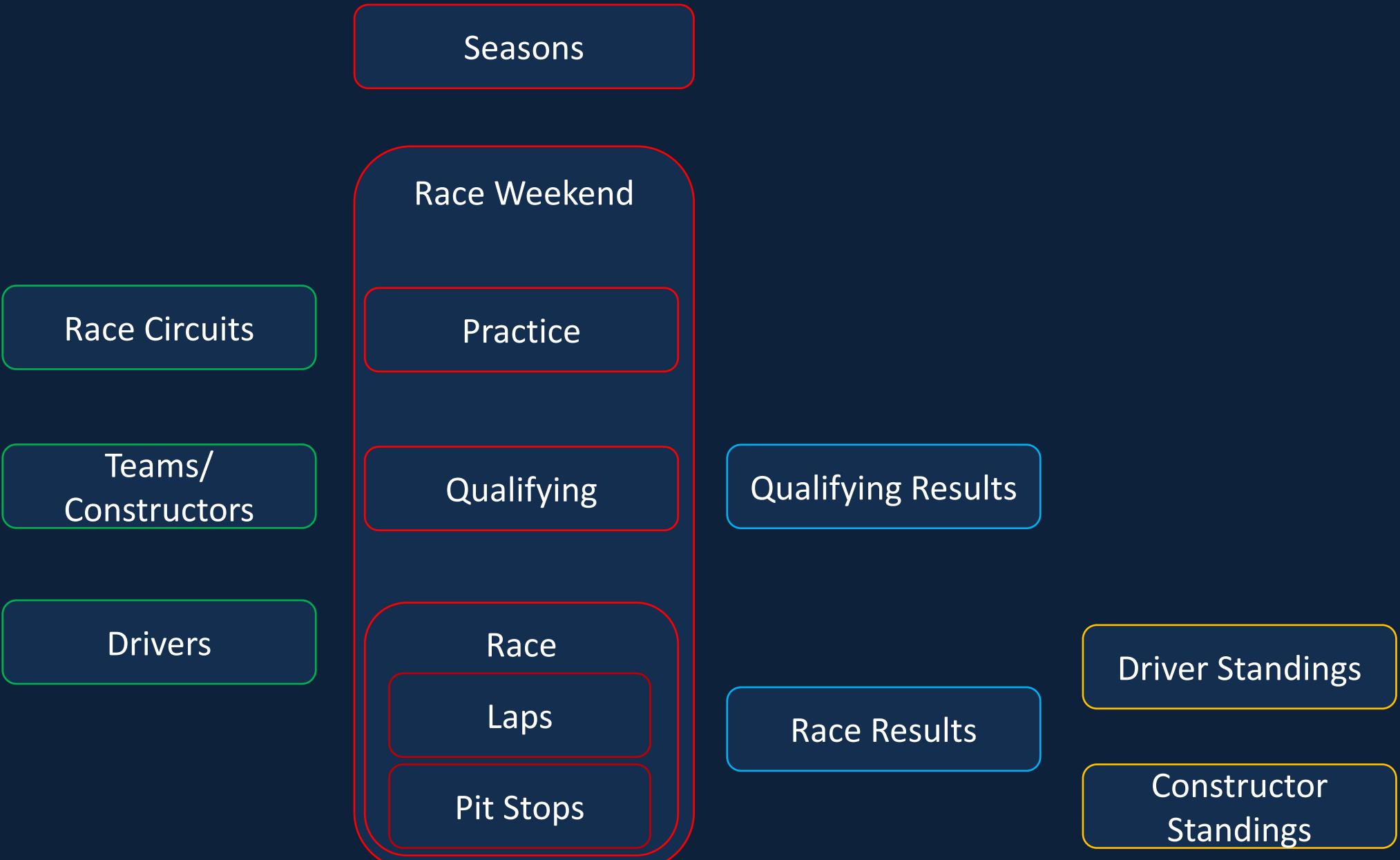
Data Overview





Formula1

Formula1 Overview



Formula1 Data Source

<http://ergast.com/mrd/>



 **Ergast Developer API** 

API Documentation

The Ergast Developer API is an experimental [web service](#) which provides a historical record of motor racing data for non-commercial purposes. Please read the [terms and conditions of use](#). The API provides data for the [Formula One](#) series, from the beginning of the world championships in 1950.

Non-programmers can query the database using the [manual interface](#) or [download the database tables in CSV format](#) for import into spreadsheets or analysis software.

If you have any comments or suggestions please post them on the [Feedback page](#). If you find any bugs or errors in the data please report them on the [Bug Reports page](#). Any enhancements to the API will be reported on the [News page](#). Example applications are shown in the [Application Gallery](#).

Overview

All API queries require a GET request using a URL of the form:

`http[s]://ergast.com/api/<series>/<season>/<round>/...`

where:

`<series>` should be set to "f1"
`<season>` is a 4 digit integer
`<round>` is a 1 or 2 digit integer

For queries concerning a whole season, or final standings, the round element may be omitted. For example:

`http://ergast.com/api/f1/2008/...`

For queries concerning the whole series both the round and the season elements may be omitted. For example:

`http://ergast.com/api/f1/...`

Index

- [API Documentation](#)
- [Season List](#)
- [Race Schedule](#)
- [Race Results](#)
- [Qualifying Results](#)
- [Standings](#)
- [Driver Information](#)
- [Constructor Information](#)
- [Circuit Information](#)
- [Finishing Status](#)
- [Lap Times](#)
- [Pit Stops](#)
- [Query Database](#)
- [Database Images](#)
- [Terms & Conditions](#)
- [Application Gallery](#)
- [Feedback](#)
- [FAQ](#)
- [Latest News](#)
- [Bug Reports](#)

Links

- [Contact Us](#)
- [Programmable Web](#)

Meta

- [Log in](#)
- [Entries RSS](#)
- [Comments RSS](#)
- [WordPress.org](#)

Search for:

Formula1 Data Source

Formula1 Data Files



Circuits	CSV
Races	CSV
Constructors	Single Line JSON
Drivers	Single Line Nested JSON
Results	Single Line JSON
PitStops	Multi Line JSON
LapTimes	Split CSV Files
Qualifying	Split Multi Line JSON Files

Import Raw Data to Data Lake



Project Requirements



Data Ingestion Requirements



Ingest All 8 files into the data lake

Ingested data must have the schema applied

Ingested data must have audit columns

Ingested data must be stored in columnar format (i.e., Parquet)

Must be able to analyze the ingested data via SQL

Ingestion logic must be able to handle incremental load

Data Transformation Requirements



Join the key information required for reporting to create a new table.

Join the key information required for Analysis to create a new table.

Transformed tables must have audit columns

Must be able to analyze the transformed data via SQL

Transformed data must be stored in columnar format (i.e., Parquet)

Transformation logic must be able to handle incremental load

Reporting Requirements



Driver Standings

Constructor Standings

Analysis Requirements



Dominant Drivers

Dominant Teams

Visualize the outputs

Create Databricks Dashboards

Scheduling Requirements



Scheduled to run every Sunday 10PM

Ability to monitor pipelines

Ability to re-run failed pipelines

Ability to set-up alerts on failures

Other Non-Functional Requirements



Ability to delete individual records

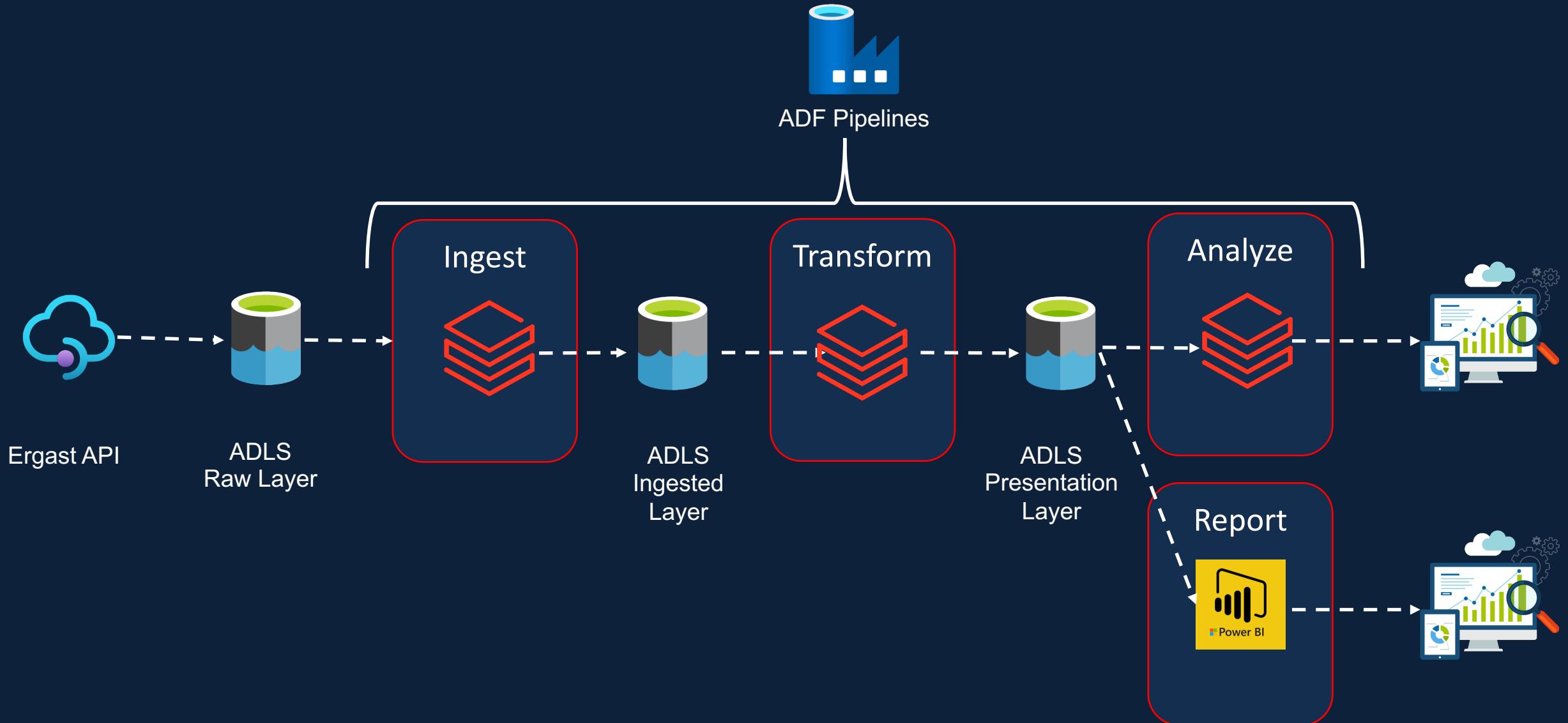
Ability to see history and time travel

Ability to roll back to a previous version

Blank

Solution Architecture Overview

Solution Architecture

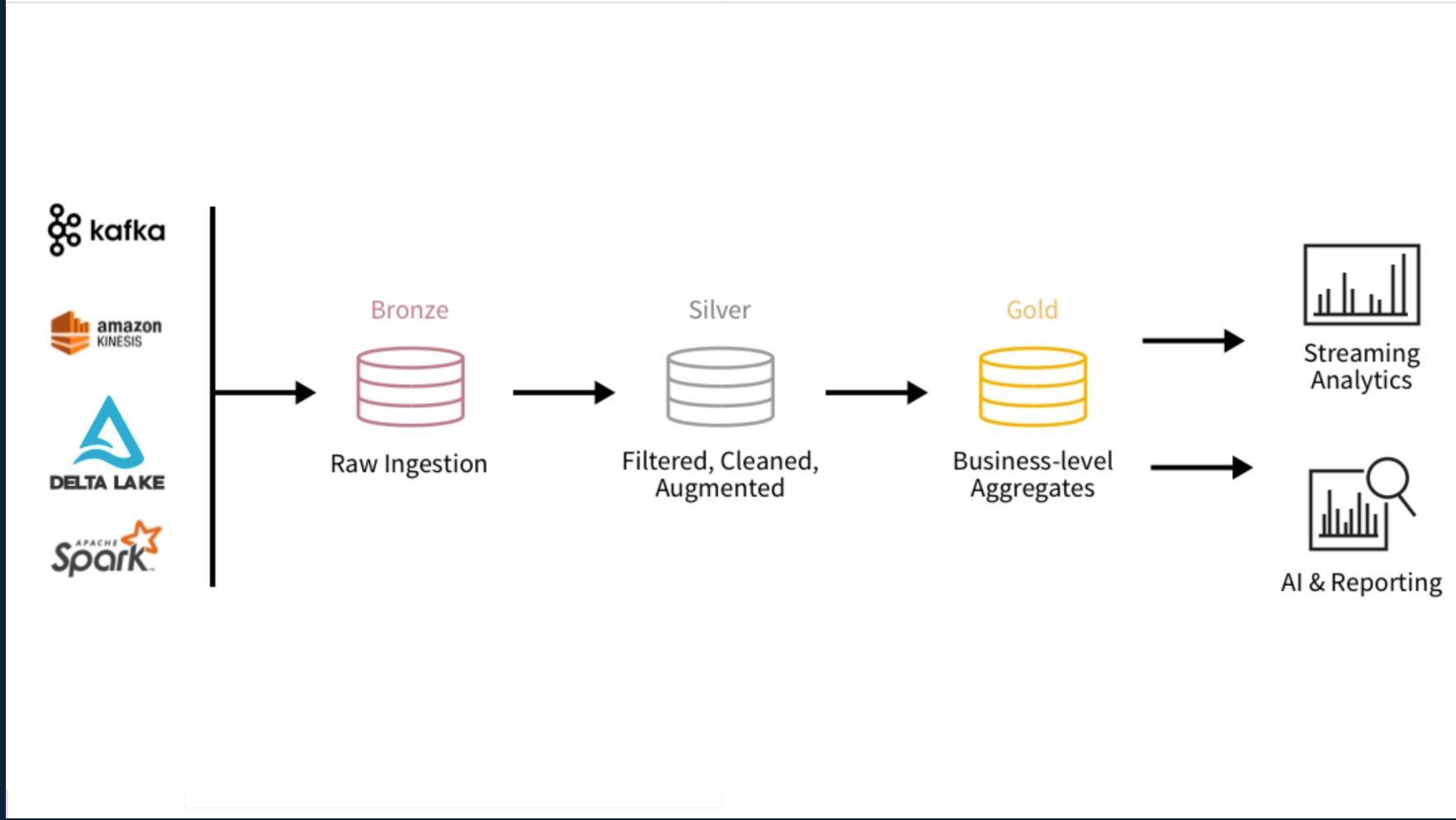


Azure Databricks Modern Analytics Architecture



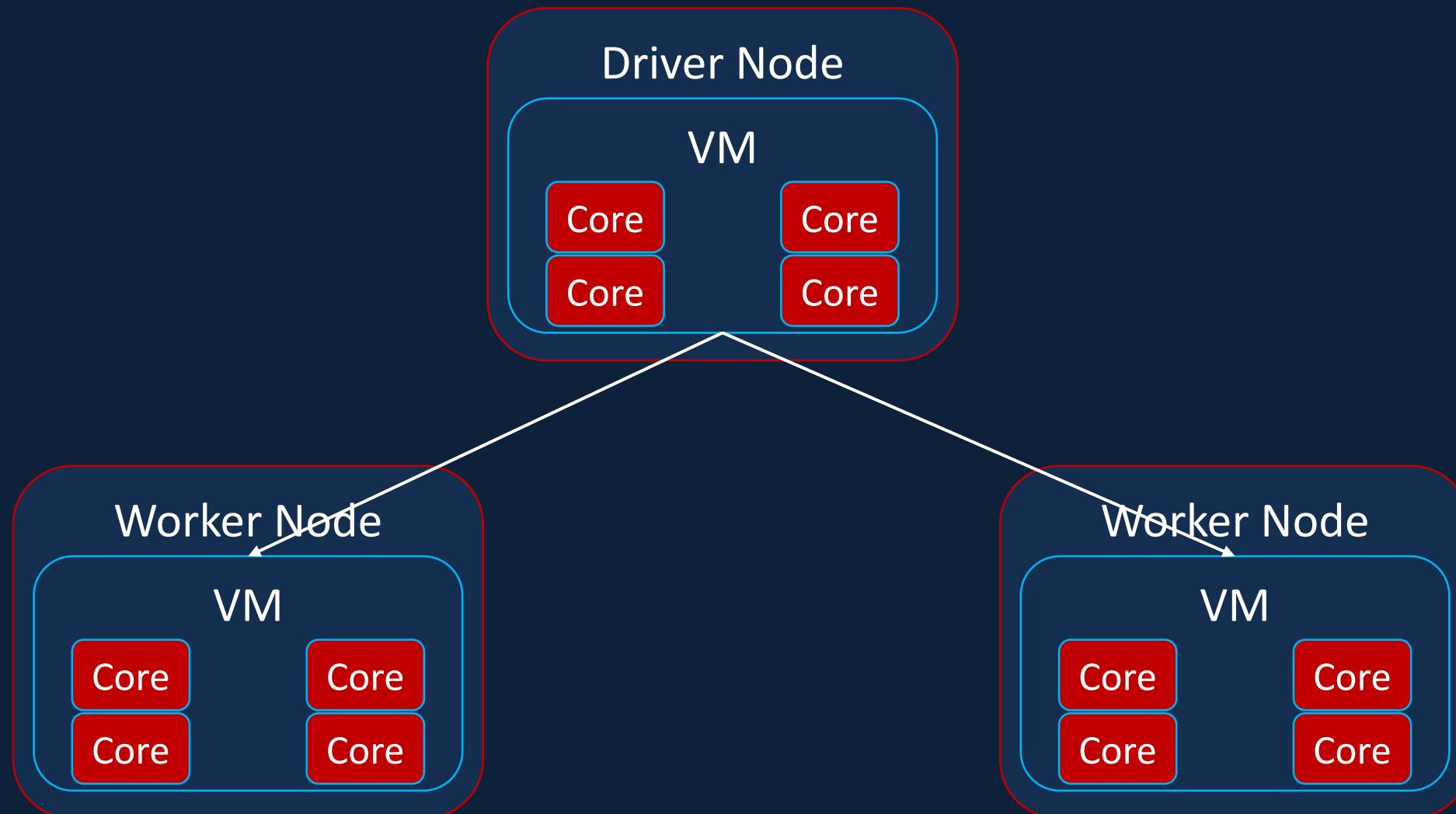
<https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/azure-databricks-modern-analytics-architecture>

Databricks Architecture

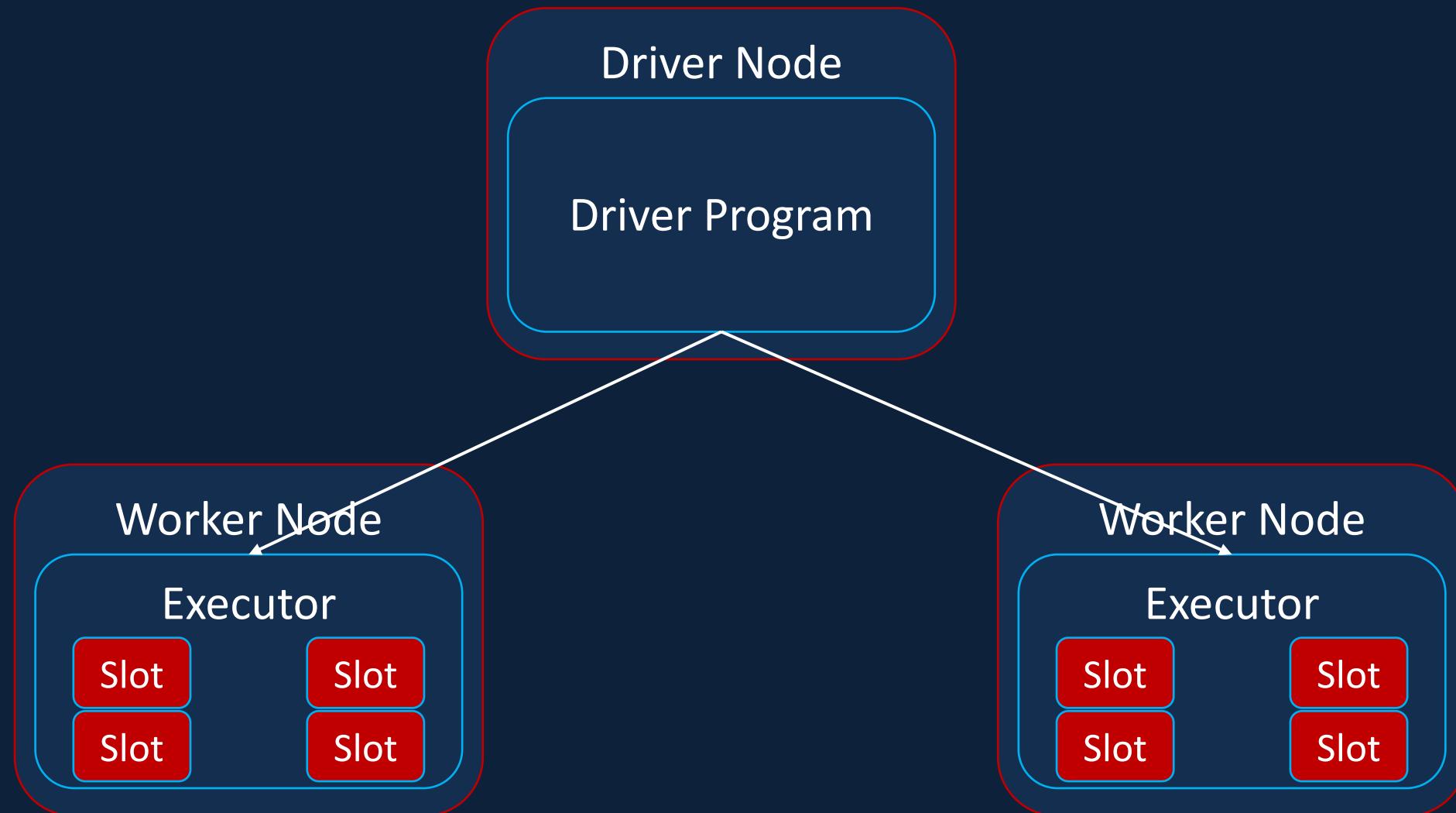


Spark Architecture

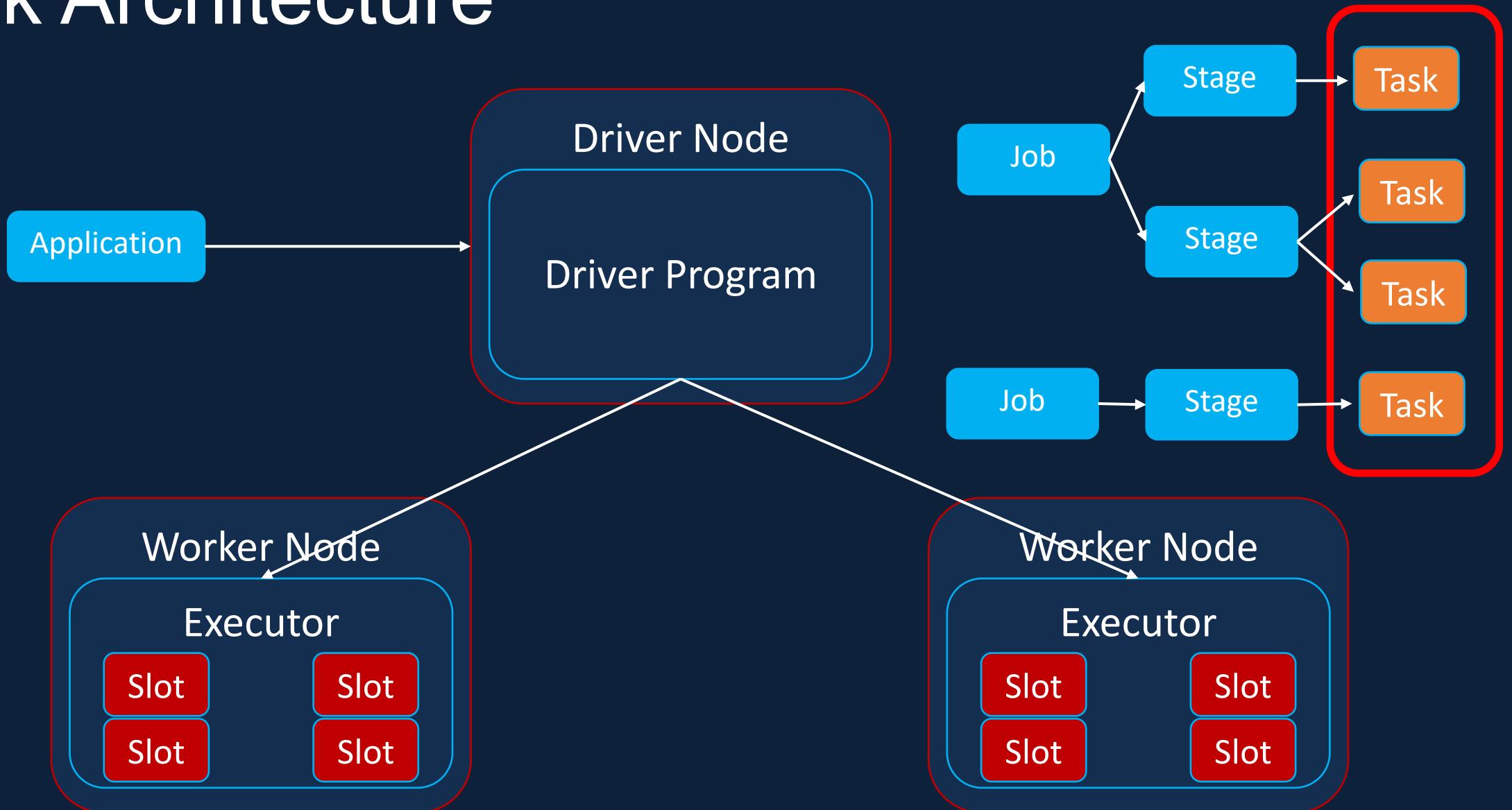
Spark Architecture



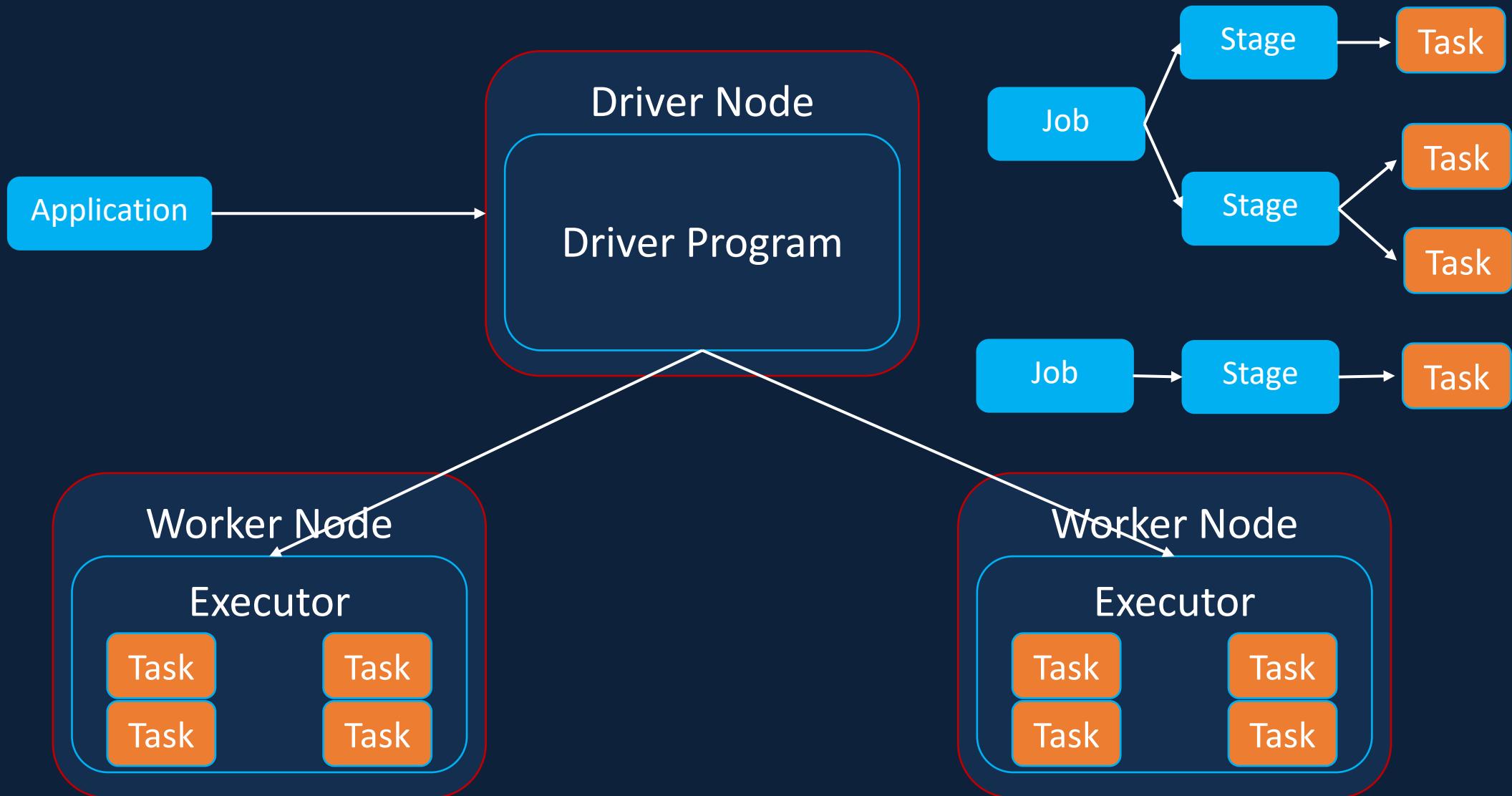
Spark Architecture



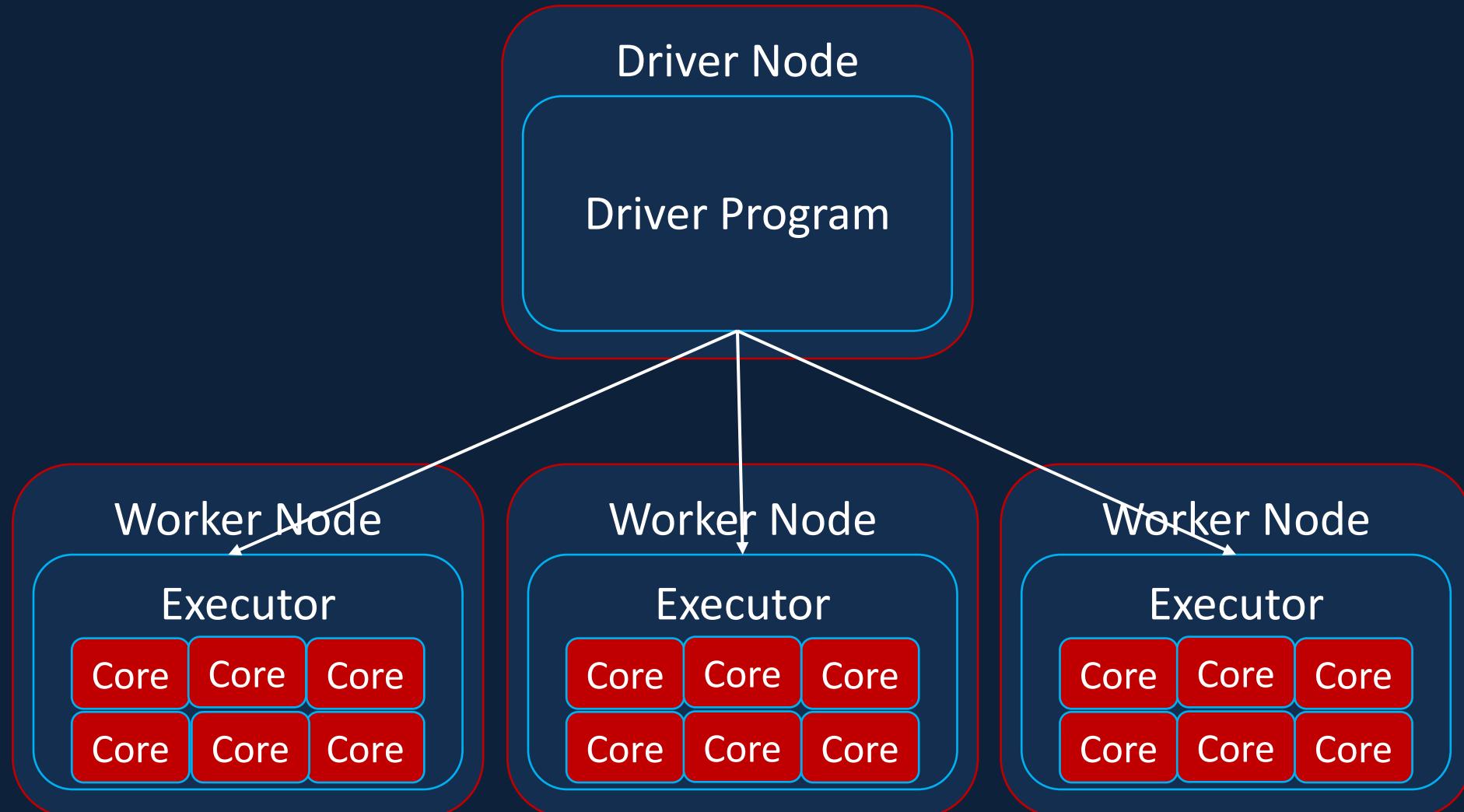
Spark Architecture



Spark Architecture



Spark Architecture – Cluster Scaling



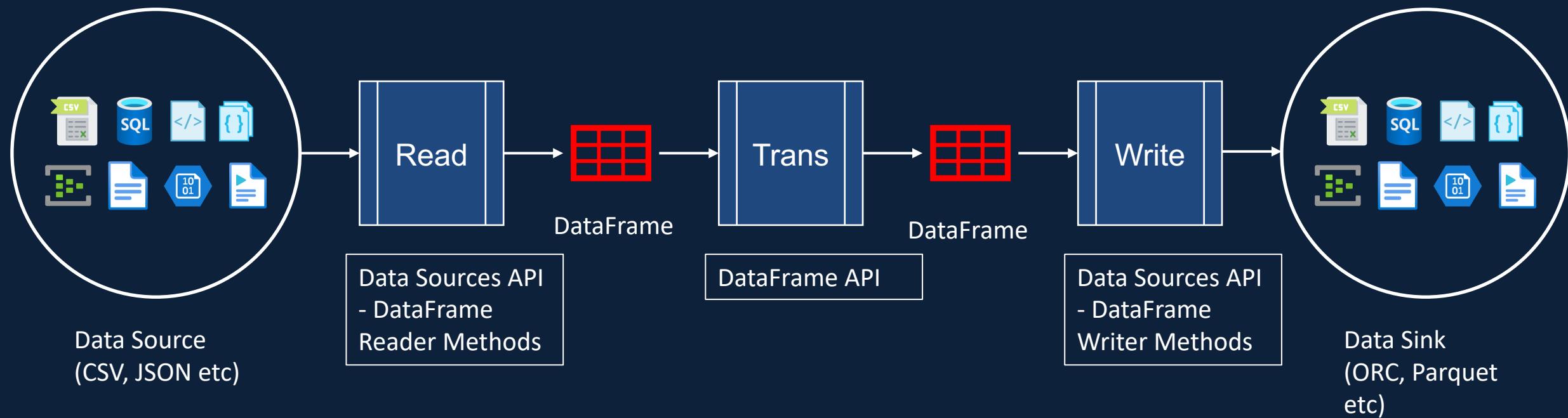
Spark DataFrame

Spark DataFrame

Driver	Team	Wins	Points
1 Max Verstappen	Red Bull	5	182
2 Lewis Hamilton	Mercedes	3	150
3 Sergio Perez	Red Bull	1	104
4 Lando Norris	McLaren	0	101
5 Valtteri Bottas	Mercedes	0	92
6 Charles Leclerc	Ferrari	0	62
7 Carlos Sainz Jr.	Ferrari	0	60
8 Daniel Ricciardo	McLaren	0	40
9 Pierre Gasly	AlphaTauri	0	39
10 Sebastian Vettel	Aston Martin	0	30
11 Fernando Alonso	Alpine	0	20
12 Lance Stroll	Aston Martin	0	14

Source: <https://www.bbc.co.uk/sport/formula1/drivers-world-championship/standings>

Spark DataFrame



Spark Documentation



Data Ingestion Overview

Data Ingestion Requirements



Ingest All 8 files into the data lake

Ingested data must have the schema applied

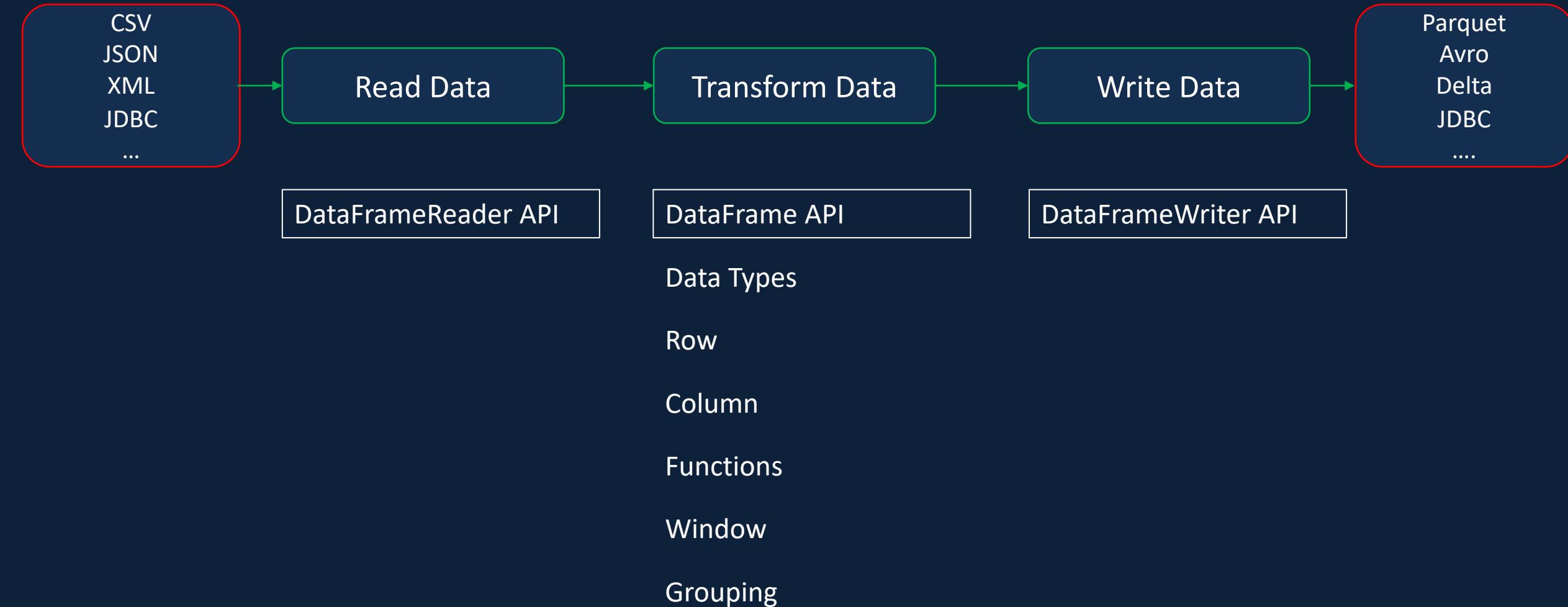
Ingested data must have audit columns

Ingested data must be stored in columnar format (i.e., Parquet)

Must be able to analyze the ingested data via SQL

Ingestion logic must be able to handle incremental load

Data Ingestion Overview



Data Ingestion Overview

File Name	File Type	Assignment/ Class work
Circuits	CSV	Class work
Races	CSV	Assignment
Constructors	Single Line JSON	Class work
Results	Single Line JSON	Assignment
Drivers	Single Line Nested JSON	Class work
PitStops	Multi Line JSON	Class work
LapTimes	Split CSV Files	Class work
Qualifying	Split Multi Line JSON Files	Assignment

Data Ingestion - Circuits



Column Name
circuitId
circuitRef
name
location
country
lat
long
alt
url (dropped)

Column Name	Data Type
circuit_id (Renamed)	Integer
circuit_ref (Renamed)	String
name	String
location	String
country	String
latitude (Renamed)	Double
longitude (Renamed)	Double
altitude (Renamed)	Integer
ingestion_date (new)	Timestamp

Data Ingestion – Races (Assignment)



Column Name
raceId
year
round
circuitId
name
date
time
url (dropped)

Column Name	Data Type
race_id (Renamed)	Integer
race_year (Renamed)	Integer
round	Integer
circuit_id (Renamed)	Integer
name	String
race_timestamp (Transformed)	Timestamp
ingestion_date (new)	Timestamp

`withColumn('race_timestamp', to_timestamp(concat(col('date'), lit(' '), col('time'))), 'yyyy-MM-dd HH:mm:ss'))`

Data Ingestion – Races (Partition By)



Column Name
raceId
year
round
circuitId
name
date
time
url (dropped)

Column Name
race_id (Renamed)
race_year (Renamed)
round
circuit_id (Renamed)
name
race_timestamp (Transformed)
ingestion_date (new)

Partition By
race_year

`withColumn('race_timestamp', to_timestamp(concat(col('date'), lit(' '), col('time'))), 'yyyy-MM-dd HH:mm:ss'))`

Data Ingestion - Constructors



Column Name
constructorId
constructorRef
name
nationality
url (dropped)

Column Name
constructor_id (Renamed)
constructor_ref (Renamed)
name
nationality
ingestion_date (new)

Data Ingestion - Drivers



Column Name
driverId
driverRef
number
code
name.forename
name.surname
dob
nationality
url (dropped)

Column Name
driver_id (Renamed)
driver_ref (Renamed)
number
code
name(transformed)
dob
nationality
ingestion_date (new)

Data Ingestion – Results (Assignment)

JSON

Read Data

Transform Data

Write Data

Parquet

Column Name

resultId

raceId

driverId

constructorId

number

grid

position

positionText

positionOrder

points

laps

time

milliseconds

fastestLap

rank

fastestLapTime

FastestLapSpeed

statusId (dropped)

Transform Data

Write Data

Column Name

result_id (renamed)

race_id (renamed)

driver_id (renamed)

constructor_id (renamed)

number

grid

position

position_text (renamed)

position_order (renamed)

points

laps

time

milliseconds

fastest_lap (renamed)

rank

fastest_lap_time (renamed)

fastest_lap_speed (renamed)

Ingestion_date (new)

Partition By
race_id

Data Ingestion - Pitstops



Column Name
raceId
driverId
stop
lap
time
duration
milliseconds

Column Name
race_id (renamed)
driver_id (renamed)
stop
lap
time
duration
milliseconds
Ingestion_date (new)

Data Ingestion - Laptimes



Column Name
raceId
driverId
lap
position
time
milliseconds

Column Name
race_id (renamed)
driver_id (renamed)
lap
position
time
milliseconds
Ingestion_date (new)

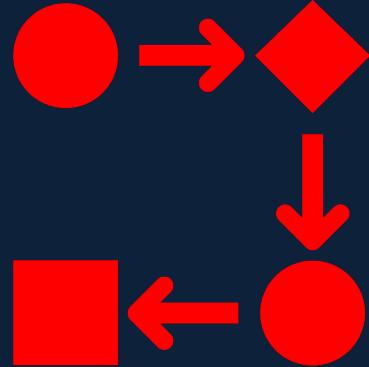
Data Ingestion – Qualifying (Assignment)



Column Name
qualifyingId
raceId
driverId
constructorId
number
position
q1
q2
q3

Column Name
qualifying_id(renamed)
race_id(renamed)
driver_id(renamed)
constructor_id(renamed)
number
position
q1
q2
q3
Ingestion_date(new)

Databricks Workflows



Include notebook

Defining notebook parameters

Notebook workflow

Databricks Jobs

Include notebook (%run)



Passing Parameters (widgets)



Notebook Workflow



Databricks Jobs



Filter/ Join Transformations



Filter Transformation

Join Transformations

Apply Transformations to F1 Project

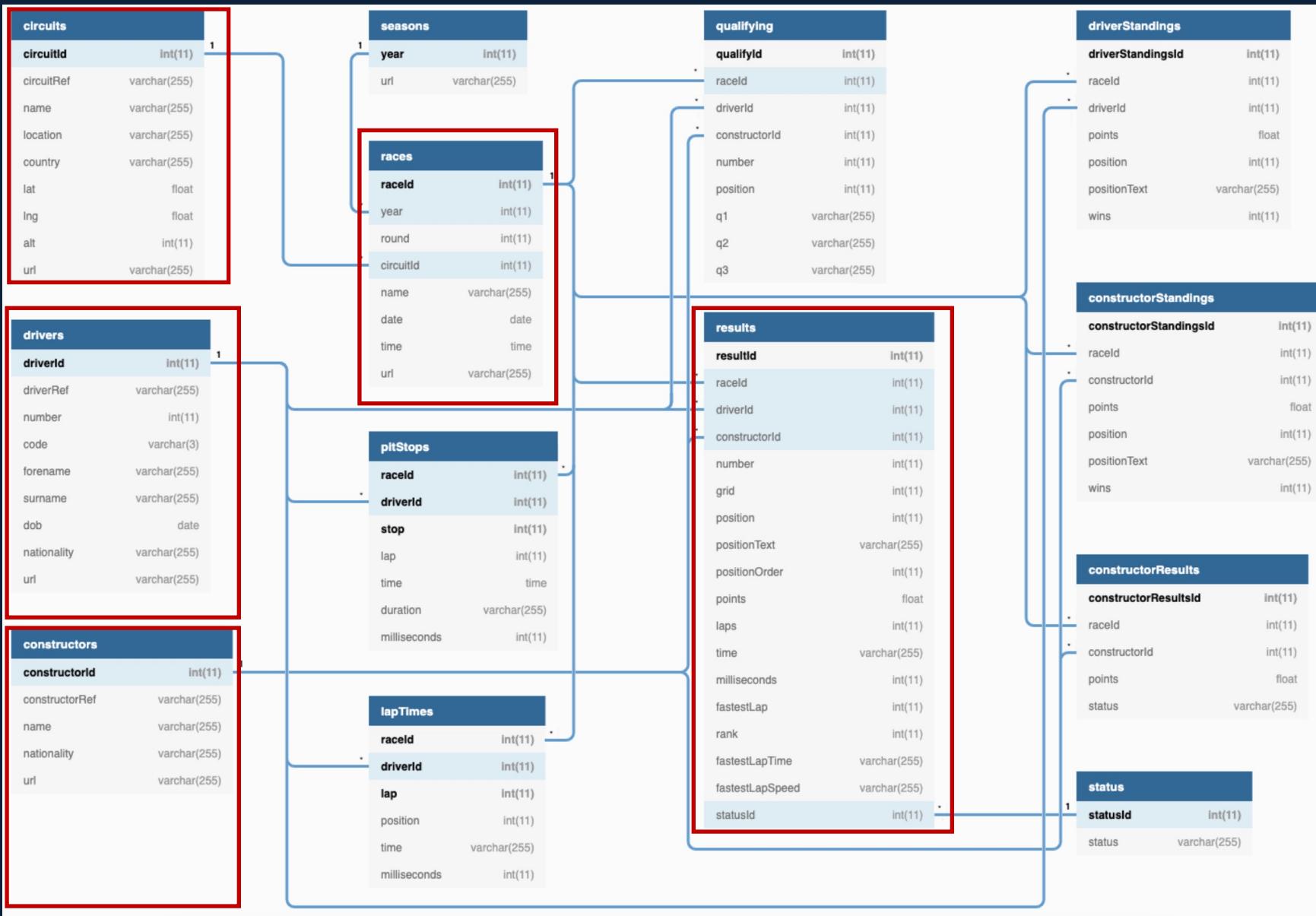
Filter Transformations

Join Transformations

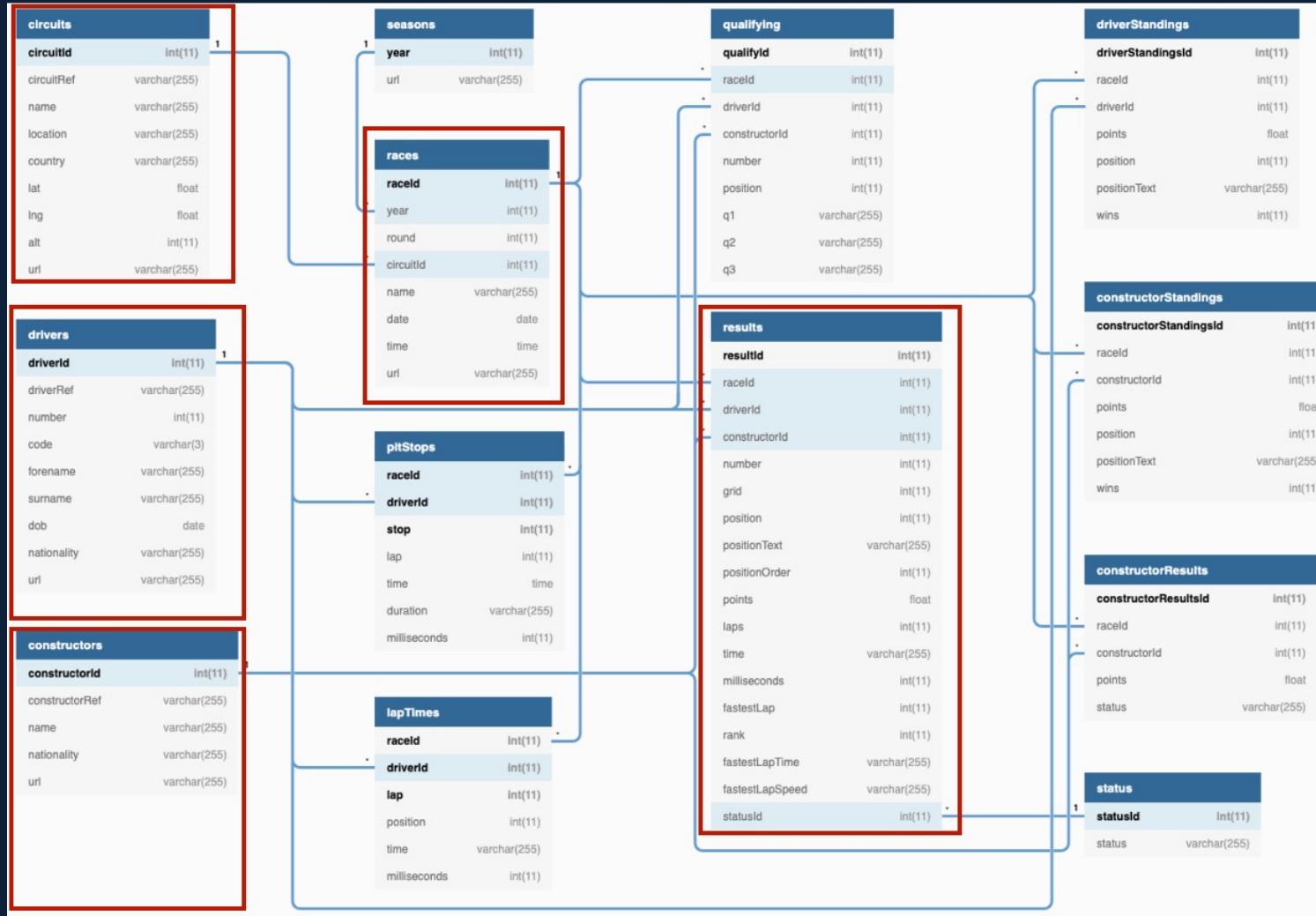
Join Transformation

Race Results

Join –Race Results



Join -Race Results

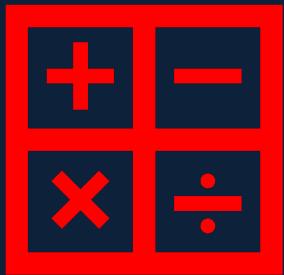


Column Name	Source
race_year	races
race_name	races
race_date	races
circuit_location	circuits
driver_name	drivers
driver_number	drivers
driver_nationality	drivers
team	constructors
grid	results
fastest_lap	results
race_time	results
points	results
created_date	current_timestamp

Set-up Environment Presentation Layer



Aggregations



Simple Aggregations

Grouped Aggregations

Window Functions

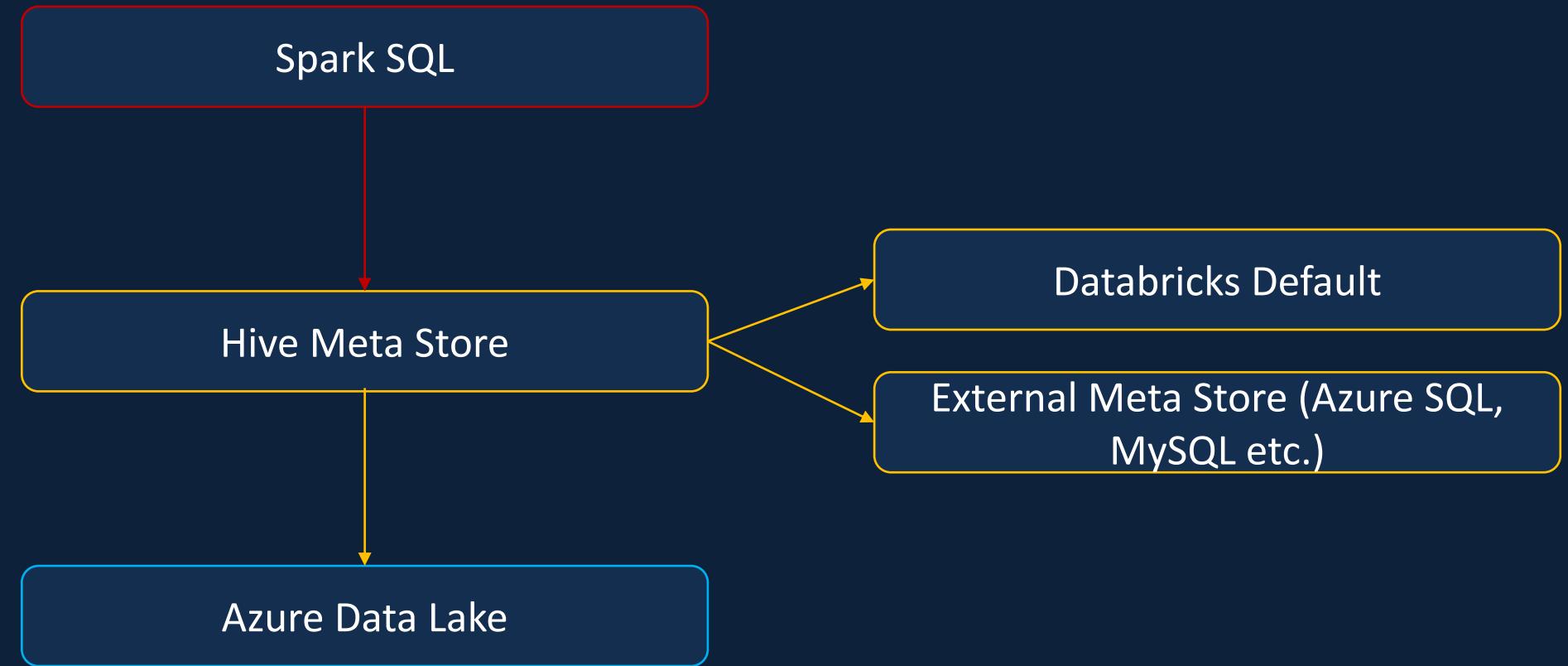
Apply Aggregations to F1 Project

Built-in Aggregate Functions

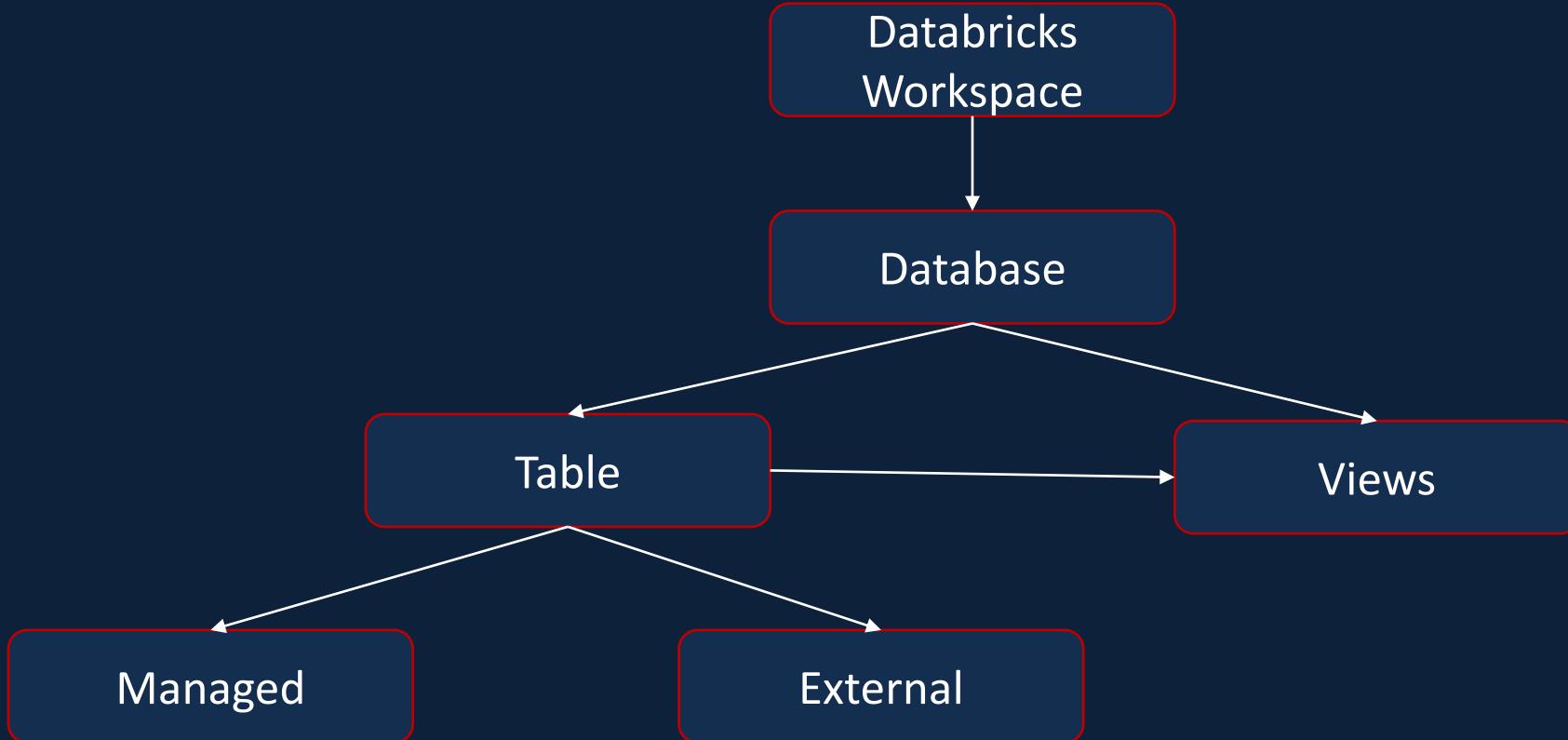
Group By

Databases/ Tables/ Views

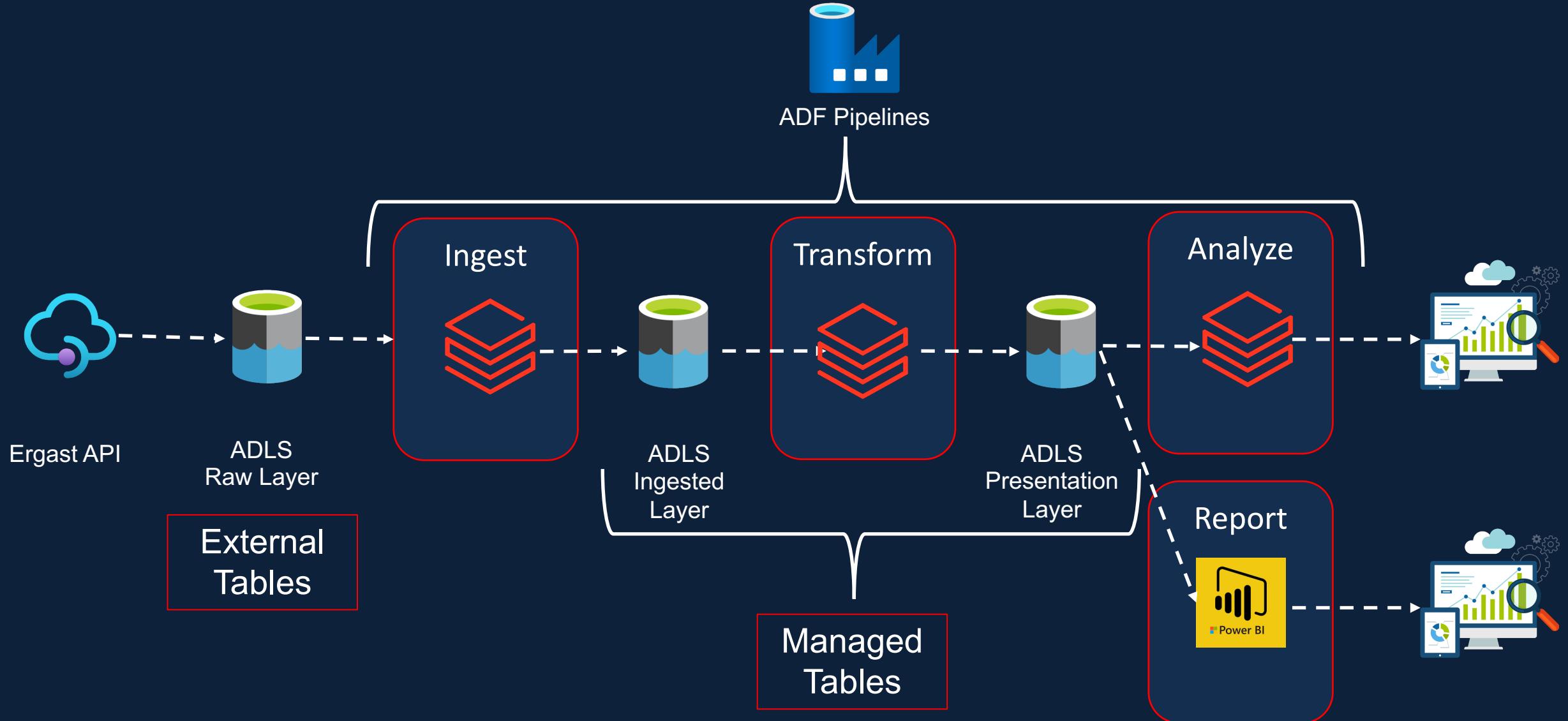
Hive Meta Store



Spark Databases/ Tables/ Views



Managed/ External Tables



Spark SQL Introduction



SQL Basics

Simple Functions

Aggregate Functions

Window Functions

Joins

Dominant Drivers/ Teams Analysis

Create a table with the data required

Granularity of the data – race_year, driver, team

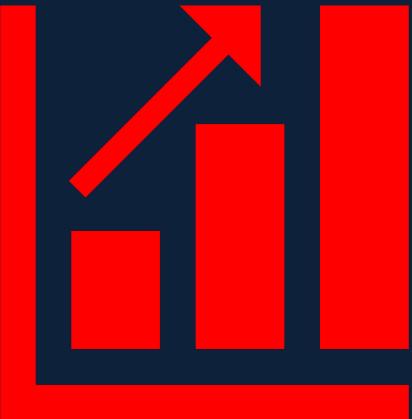
Rank the dominant drivers of all time/ last decade etc

Rank the dominant teams of all time/ last decade etc

Visualization of dominant drivers

Visualization of dominant teams

Incremental Load



Data Loading Patterns

F1 Project Load Pattern

Implementation

Data Load Types

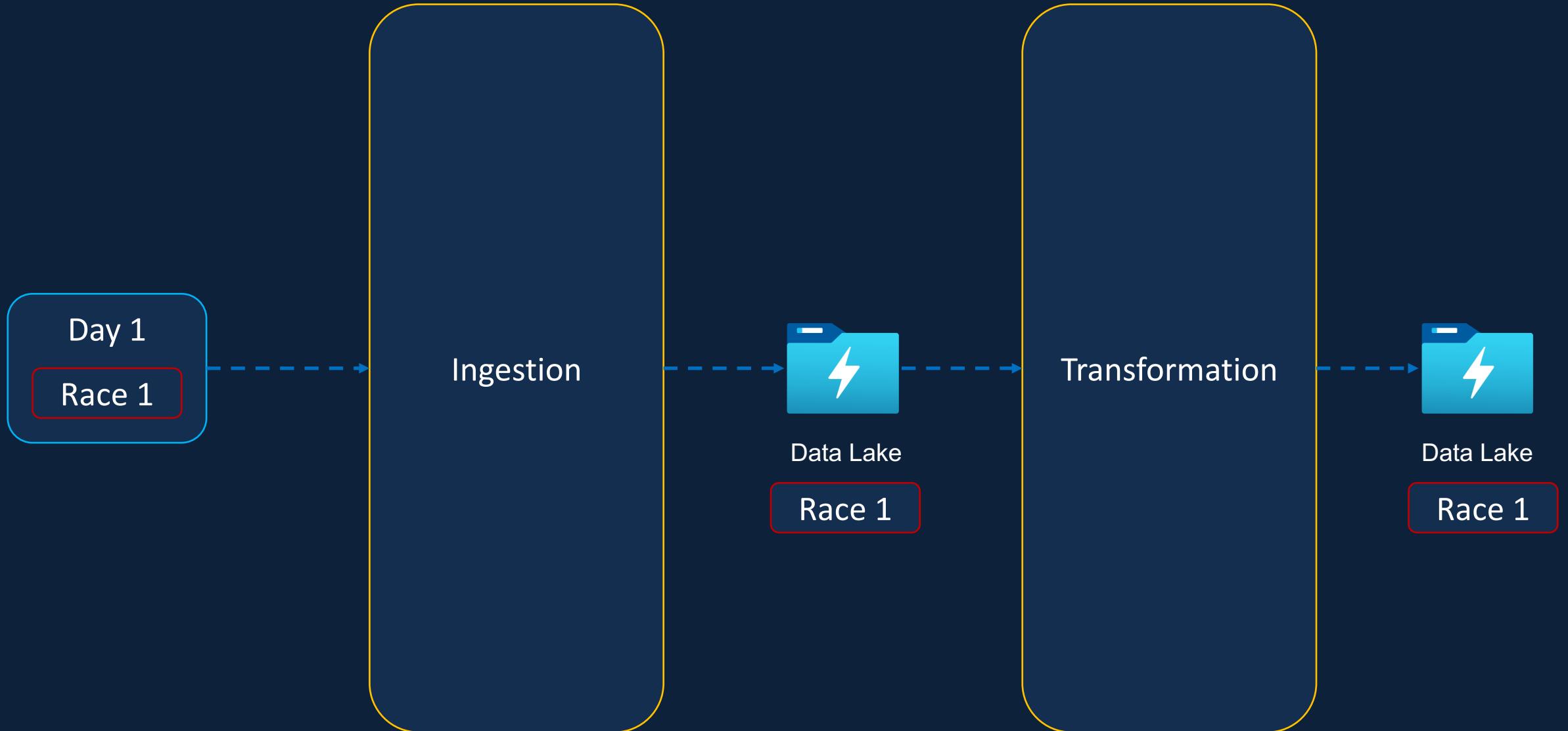
Full Load

Incremental Load

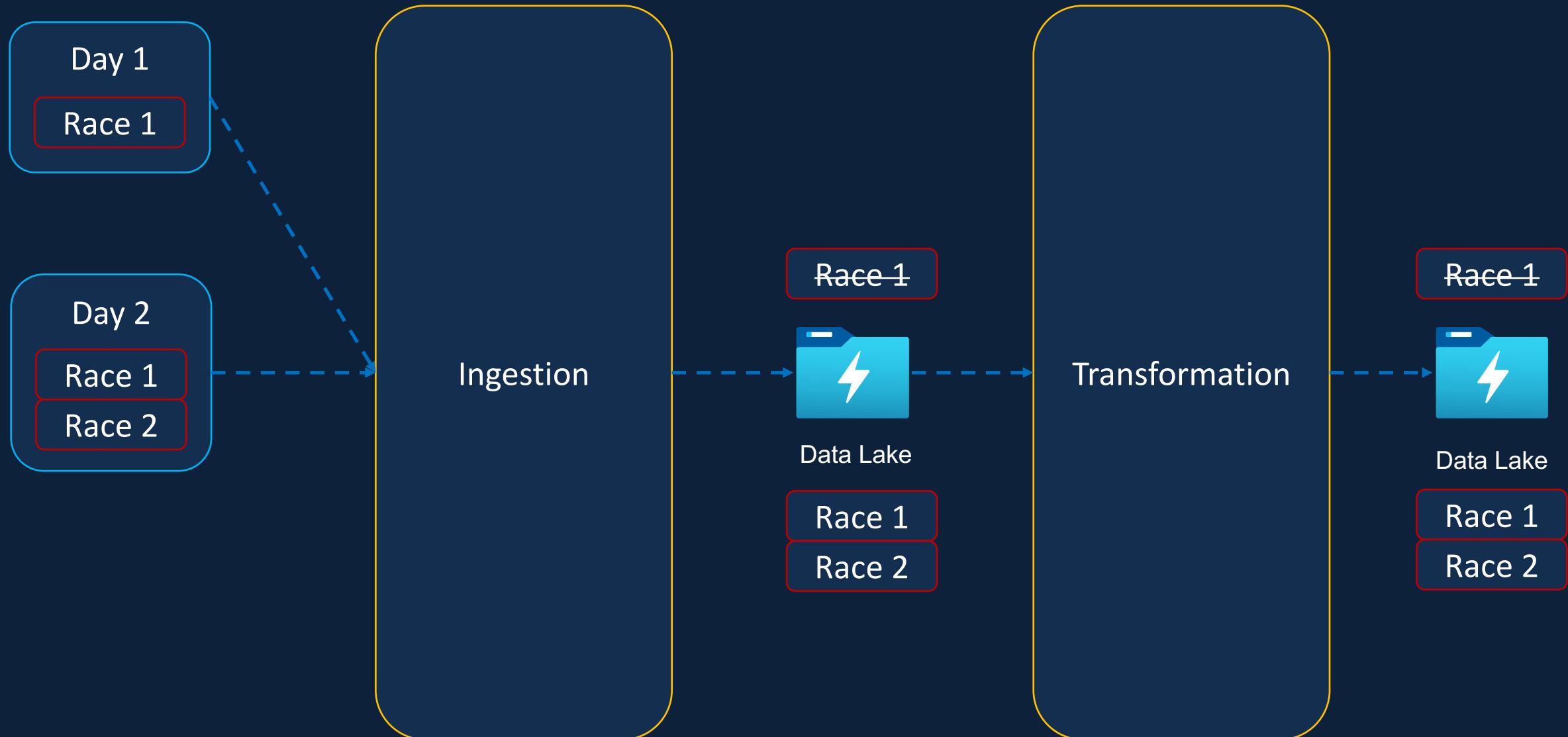
Full Dataset



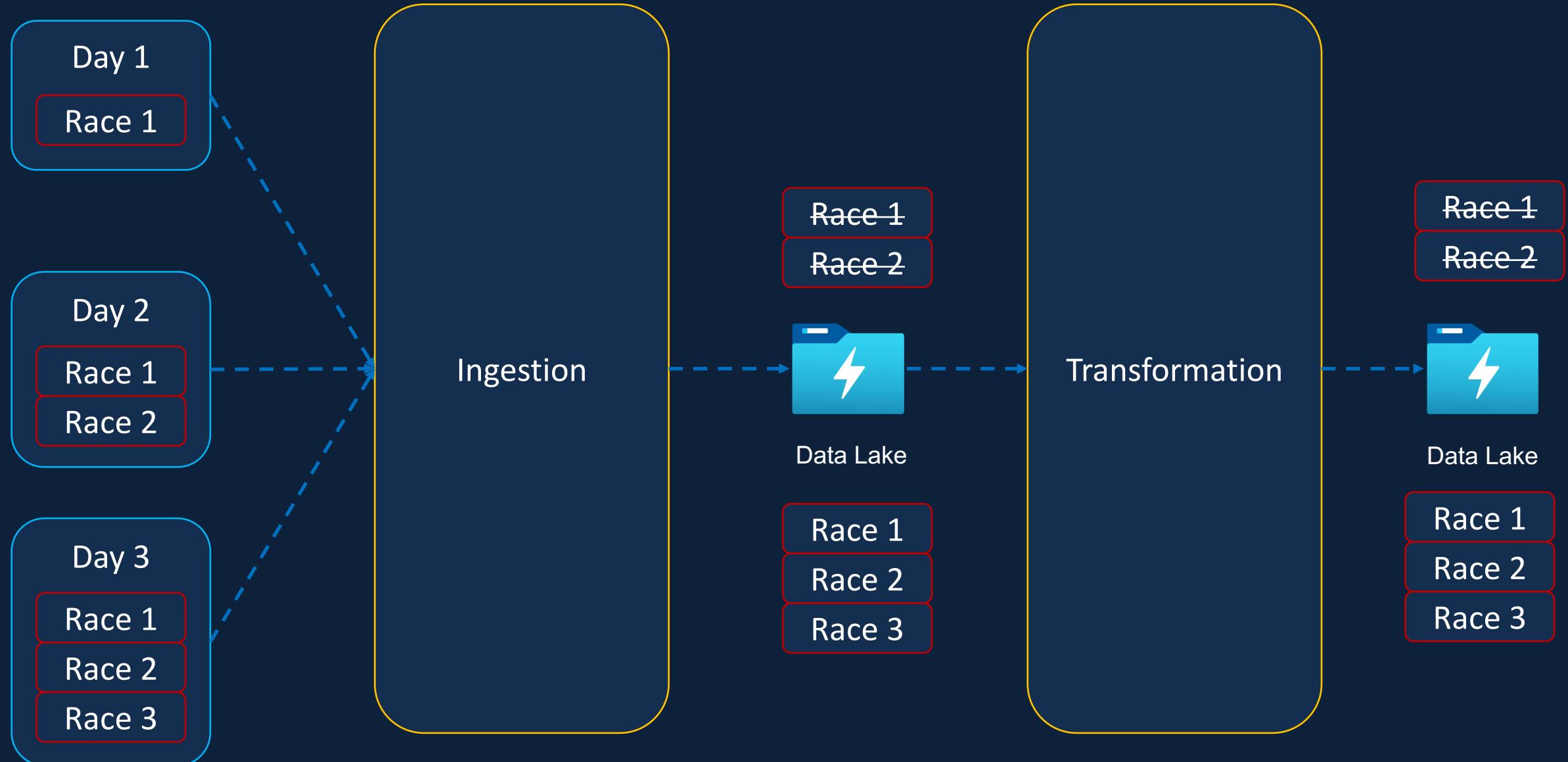
Full Refresh/ Load – Day 1



Full Refresh/ Load – Day 2



Full Refresh/ Load – Day 3



Incremental Dataset

Day 1

Race 1

Day 2

Race 2

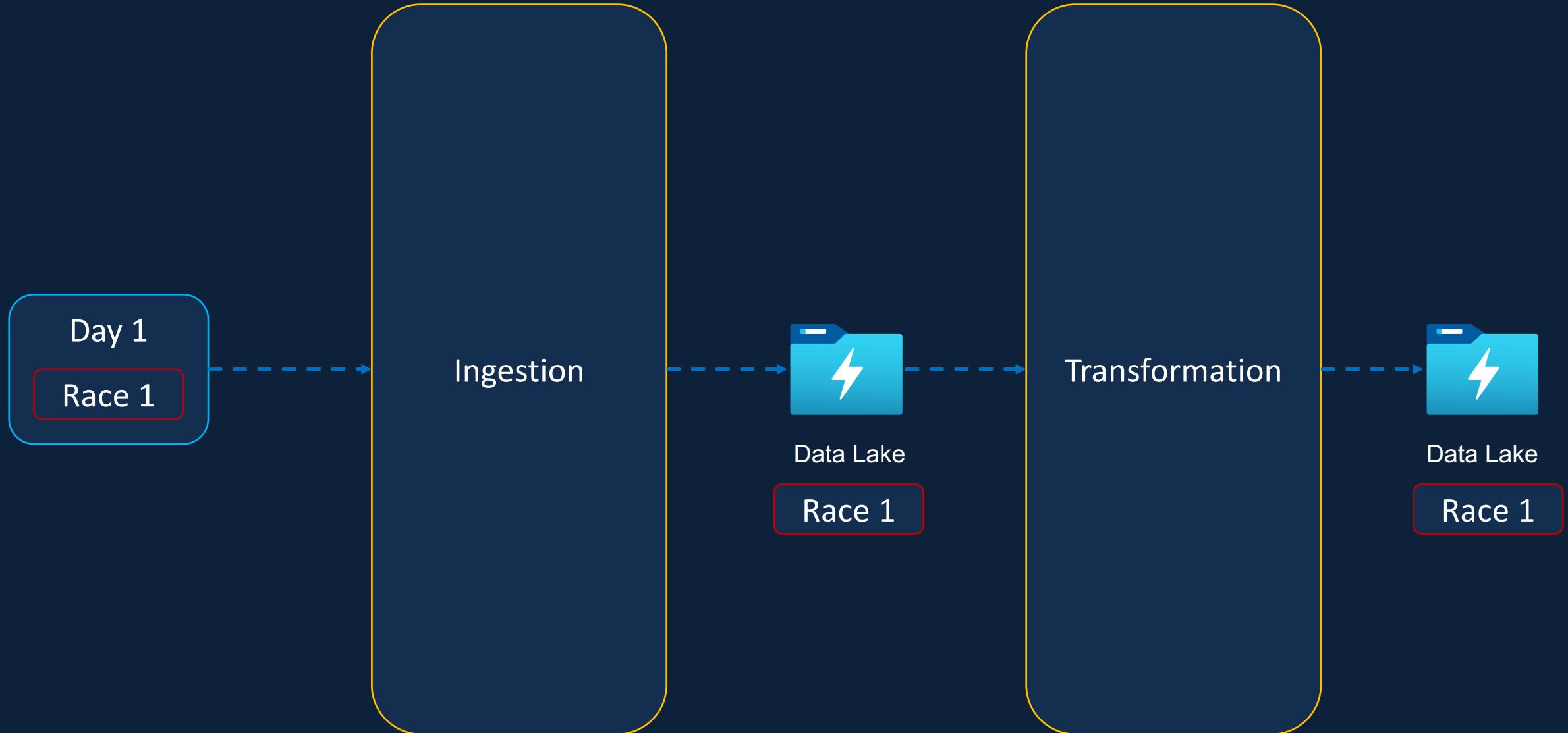
Day 3

Race 3

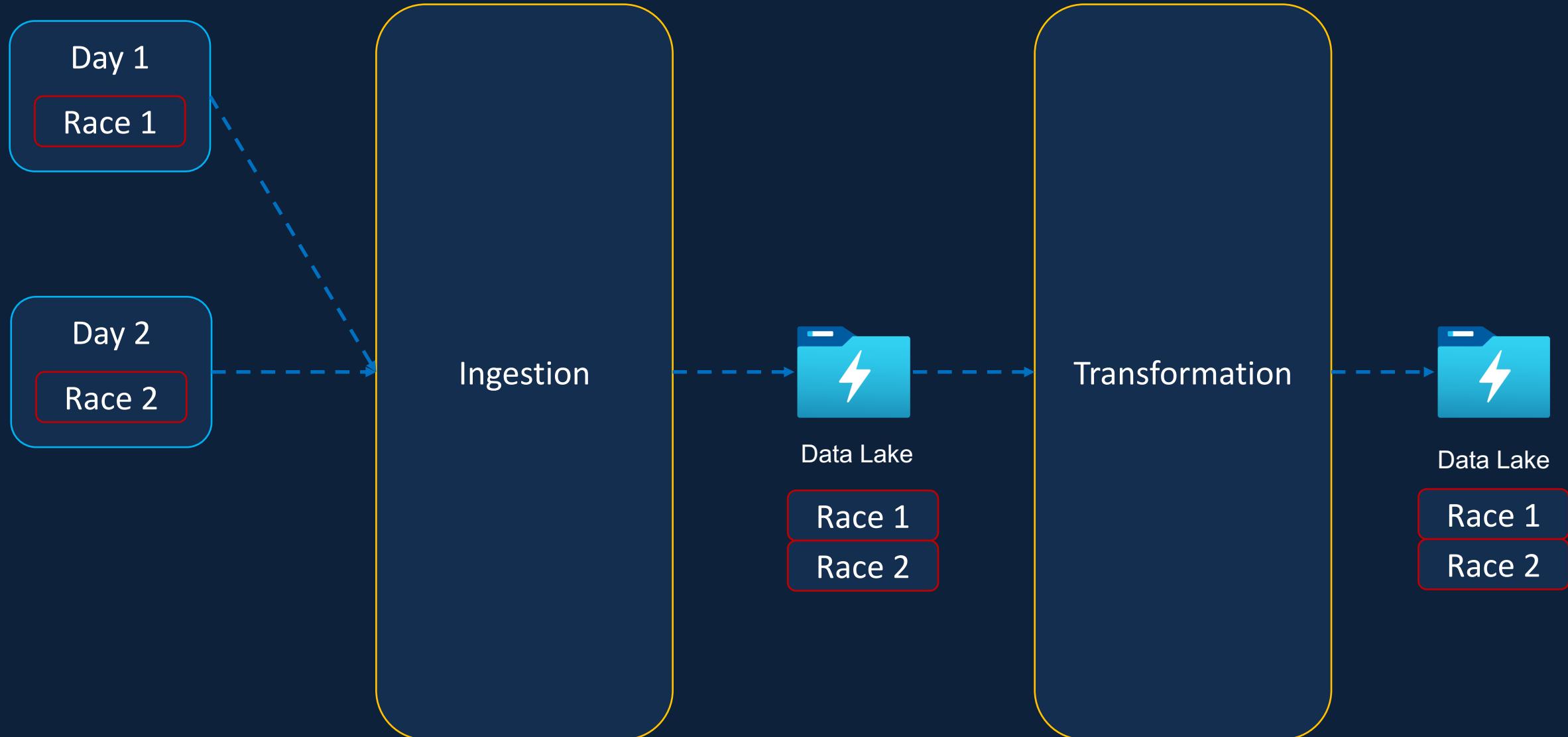
Day 4

Race 4

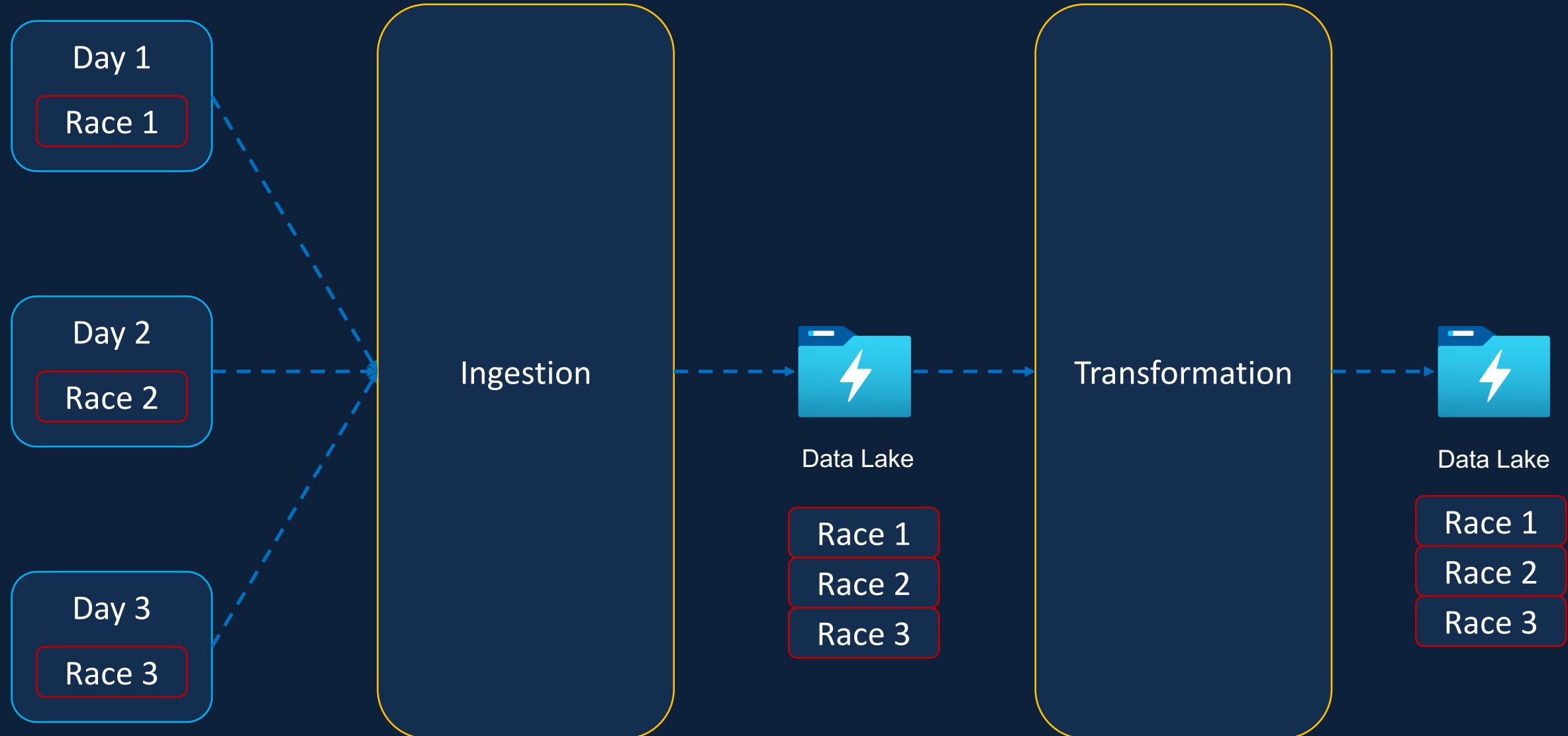
Incremental Load – Day 1



Incremental Load – Day 2



Incremental Load – Day 3



Hybrid Scenarios

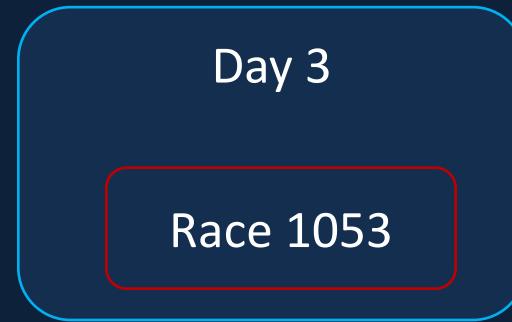
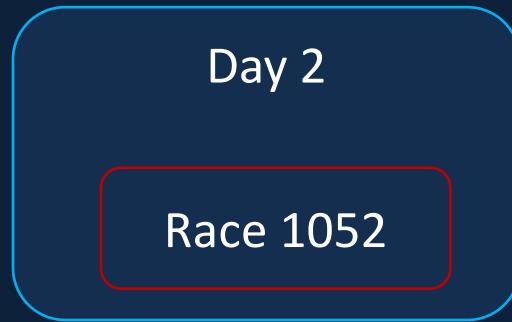
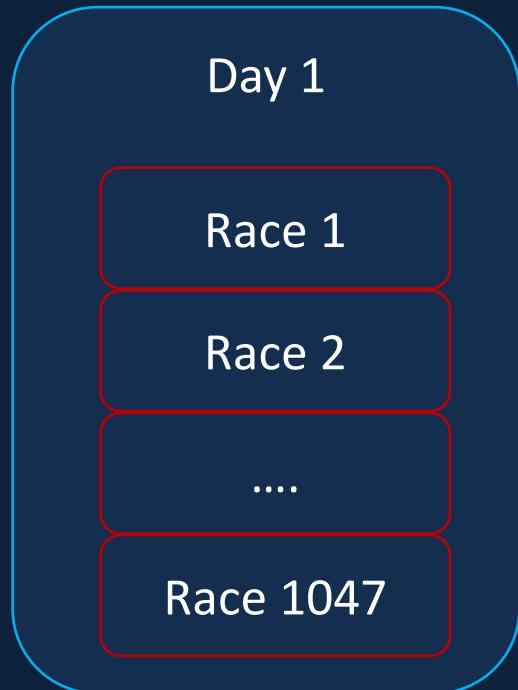
Full Dataset received, but data loaded & transformed incrementally

Incremental dataset received, but data loaded & transformed in full

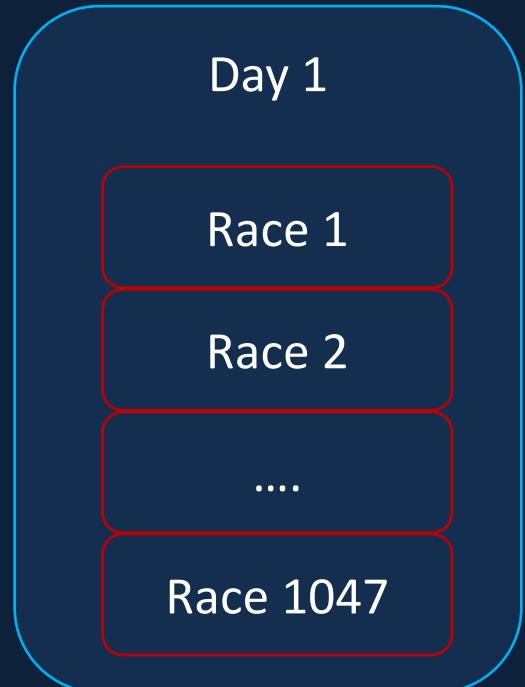
Data received contains both full and incremental files

Incremental data received. Ingested incrementally & transformed in full

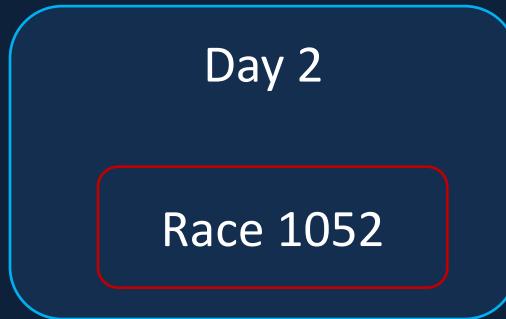
Formula1 Scenario



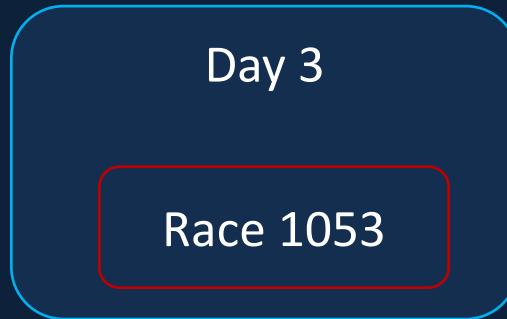
Formula1 Scenario / Solution 1



Full Refresh

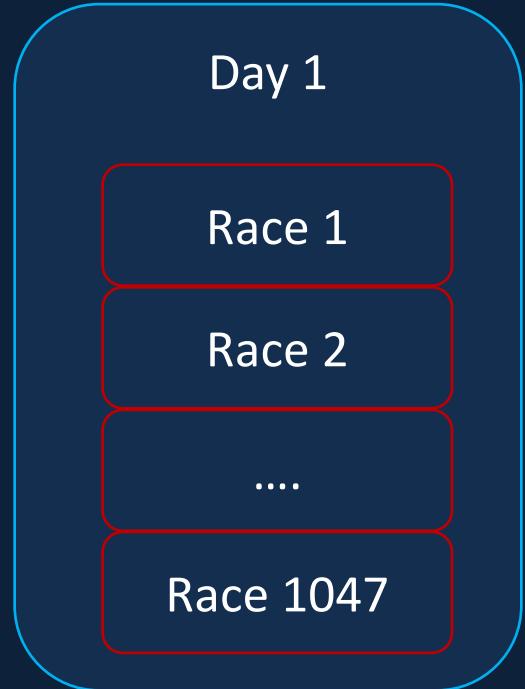


Incremental Load

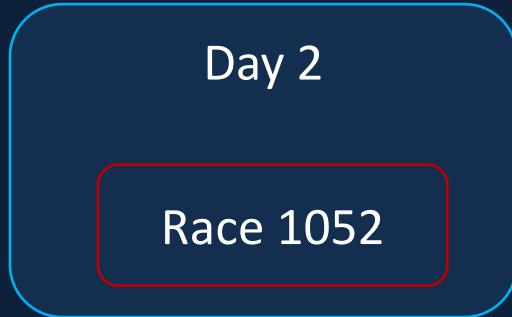


Incremental Load

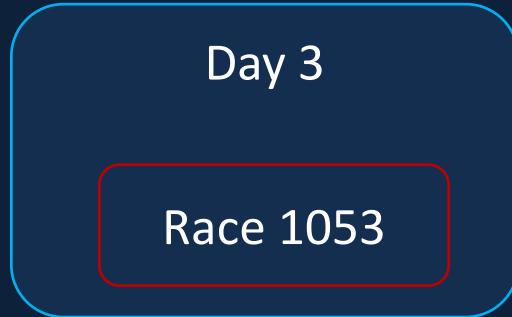
Formula1 Scenario / Solution 2



Incremental Load



Incremental Load



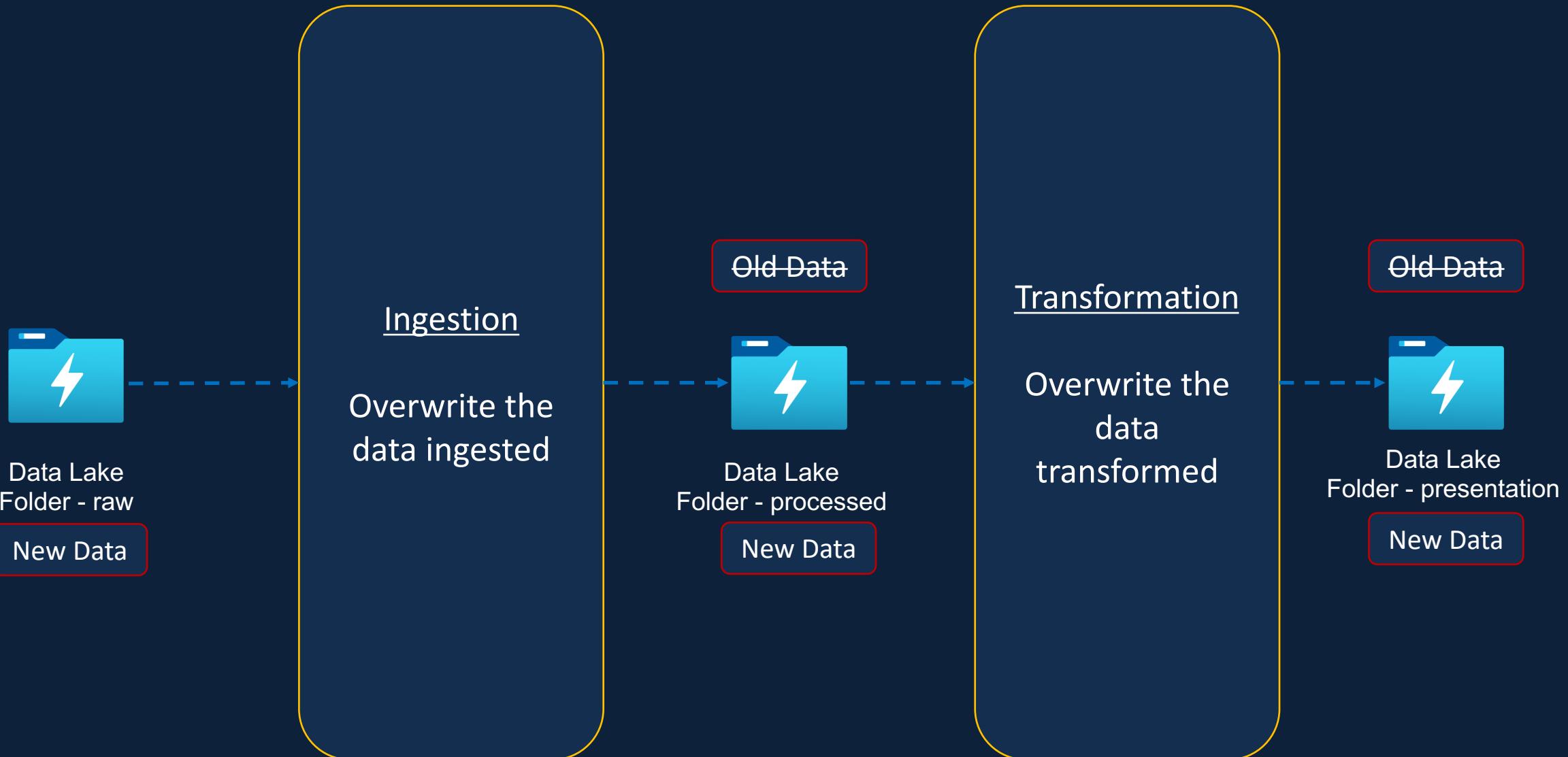
Incremental Load

Formula1 Data Files

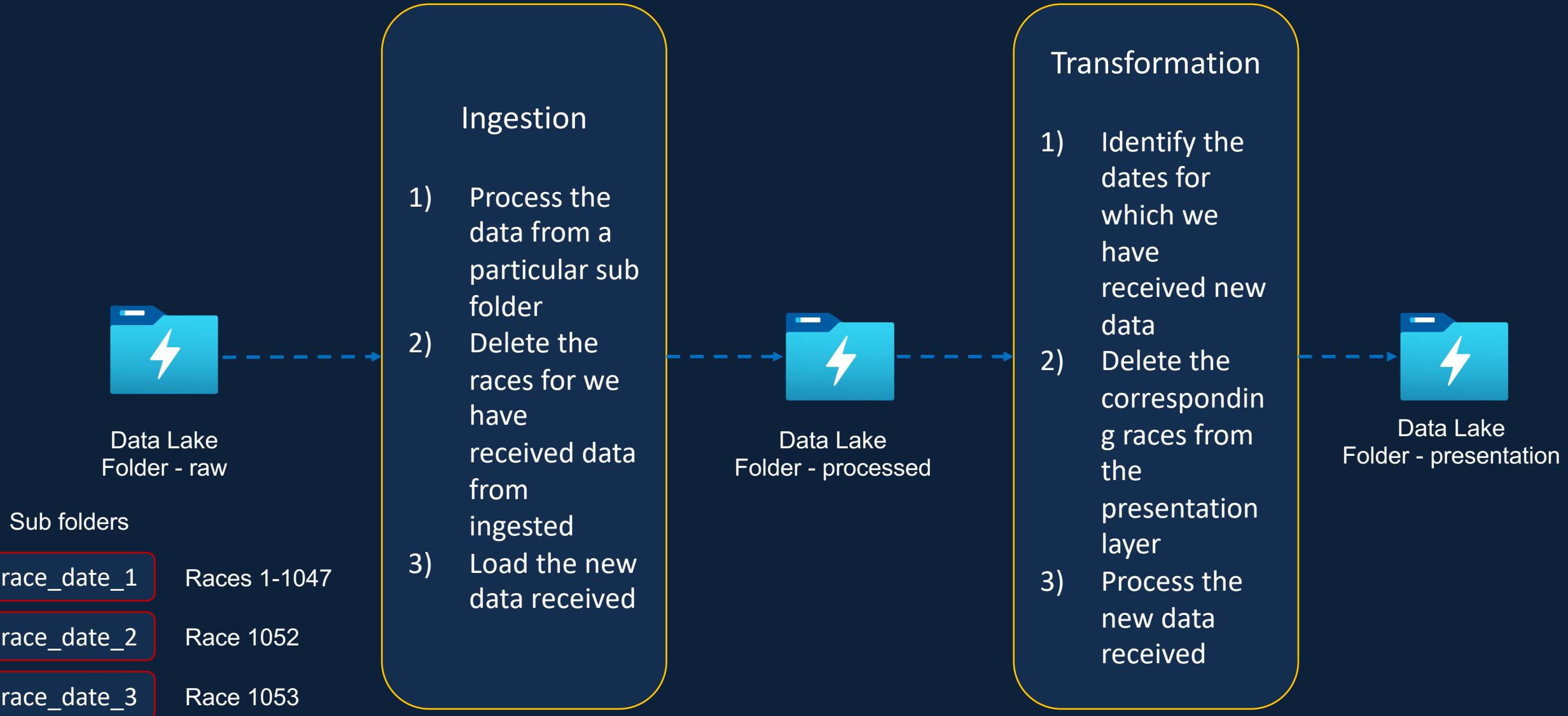


Circuits	Data from all races
Races	Data from all races
Constructors	Data from all races
Drivers	Data from all races
Results	Data only from that race
PitStops	Data only from that race
LapTimes	Data only from that race
Qualifying	Data only from that race

Current Solution



New Solution



Delta Lake



Pitfalls of Data Lakes

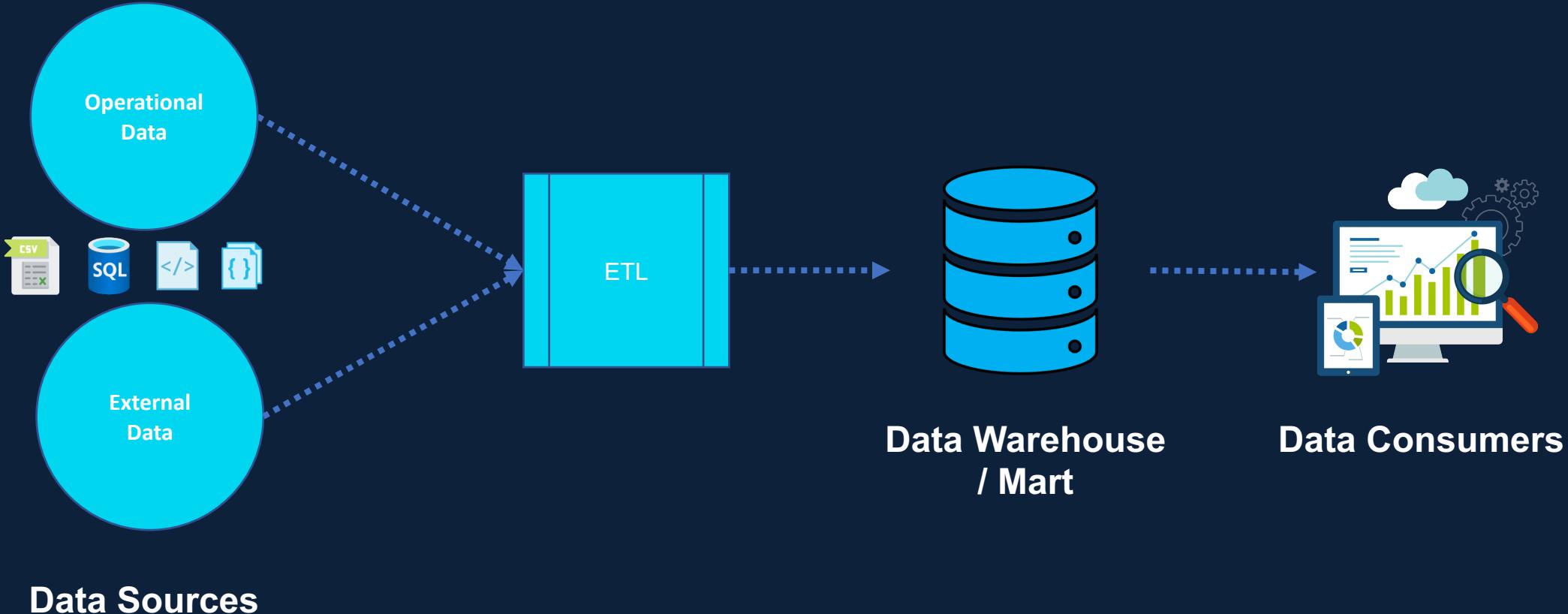
Lakehouse Architecture

Delta Lake Capabilities

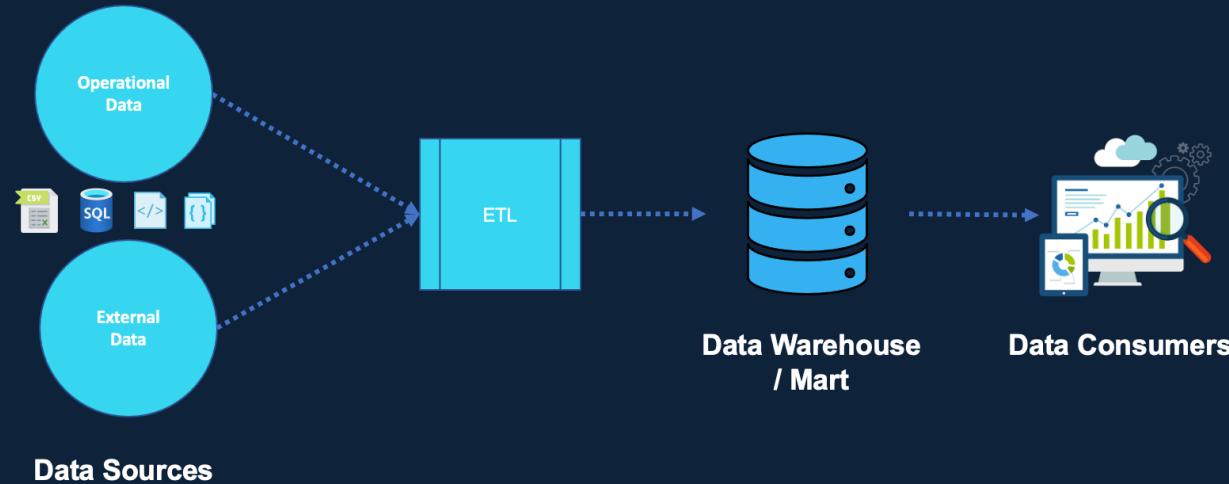
Convert F1 project to Delta Lake

Delta Lake

Data Warehouse



Data Warehouse



Lack of support for unstructured data

Longer to ingest new data

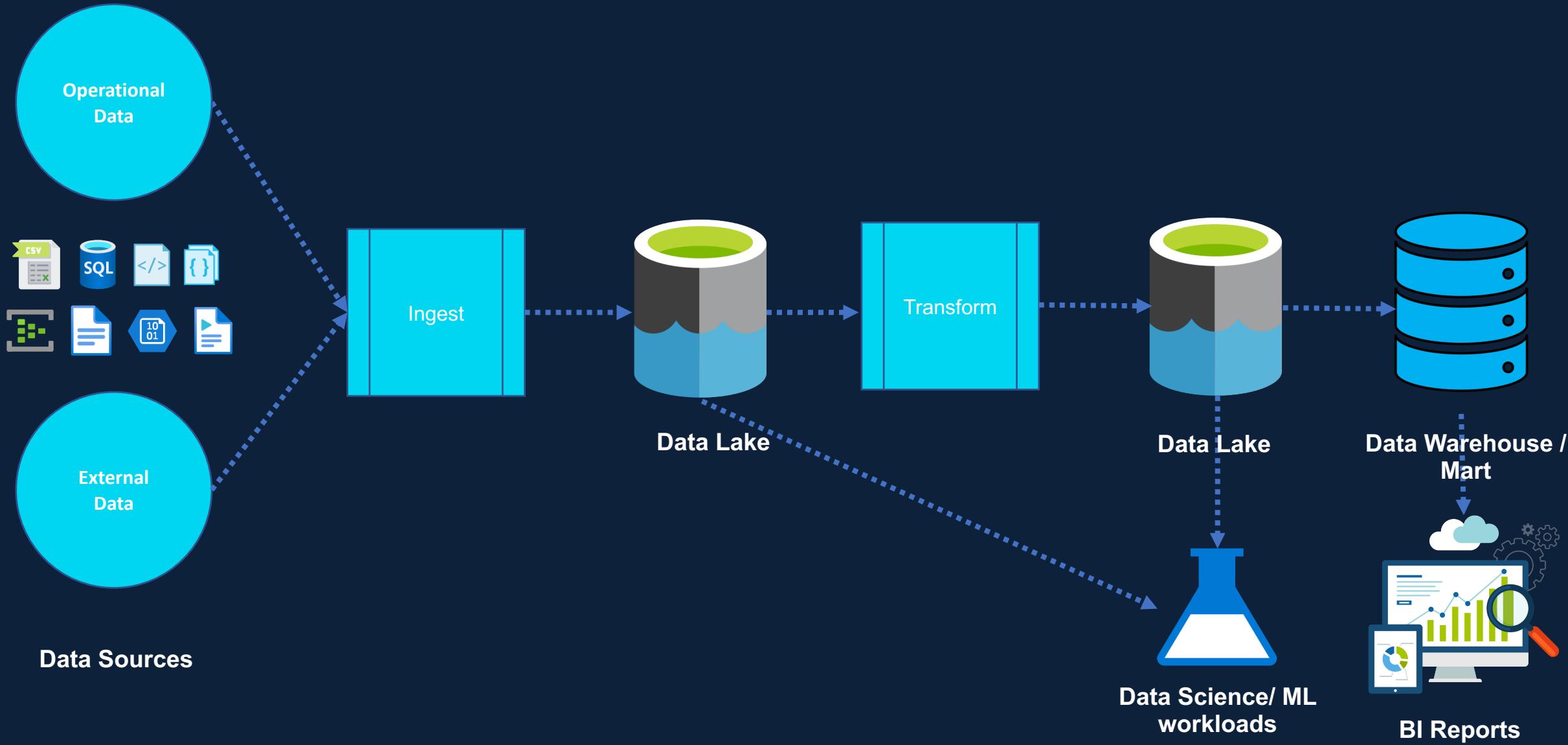
Proprietary data formats

Scalability

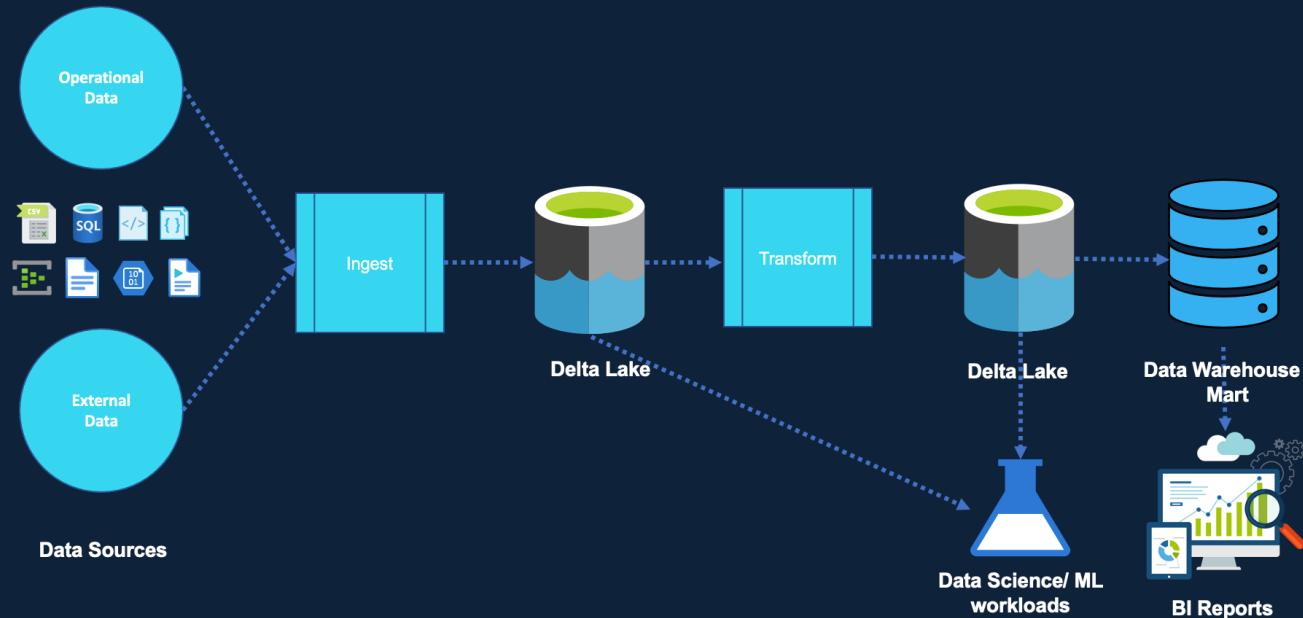
Expensive to store data

Lack of support for ML/ AI workloads

Data Lake

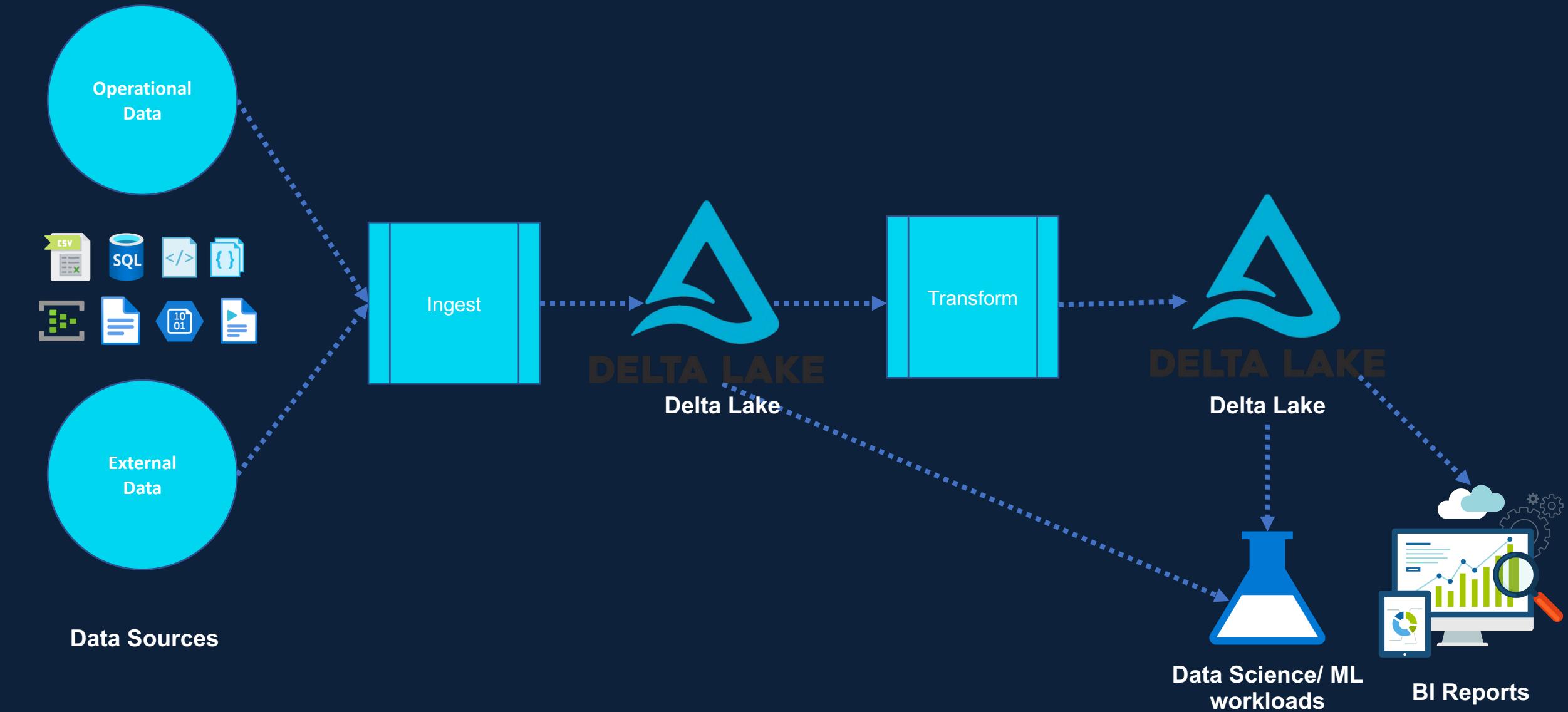


Data Lake

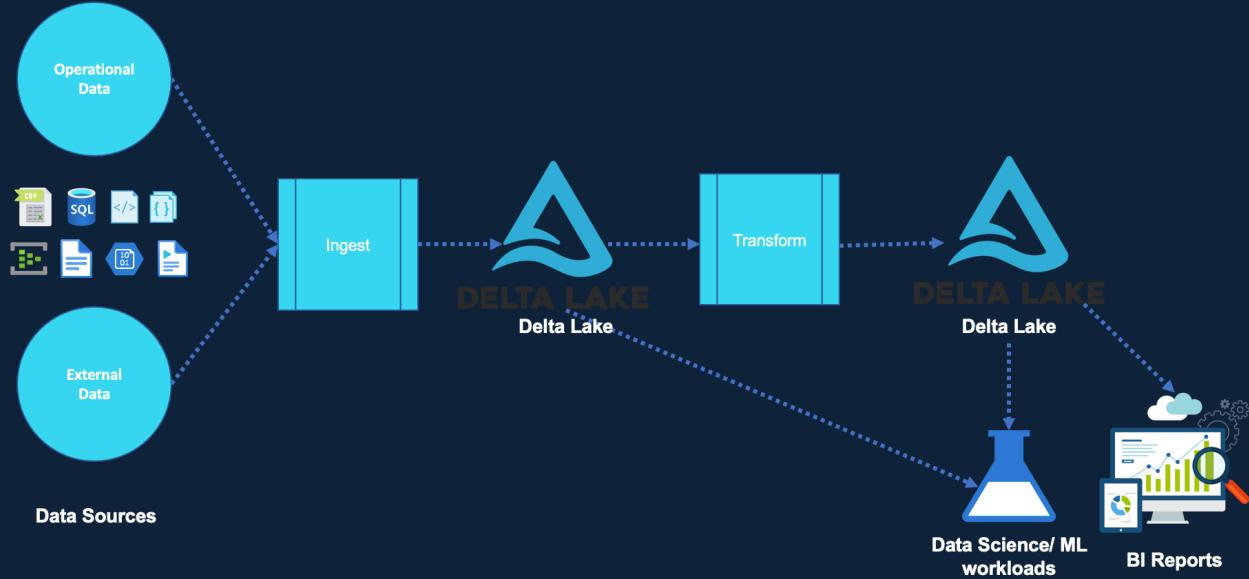


- No support for ACID transactions
- Failed jobs leave partial files
- Inconsistent reads
- Unable to handle corrections to data
- Unable to roll back any data.
- Lack of ability remove data for GDPR etc
- No history or versioning
- Poor performance
- Poor BI support
- Complex to set-up
- Lambda architecture for streaming workloads

Data Lakehouse

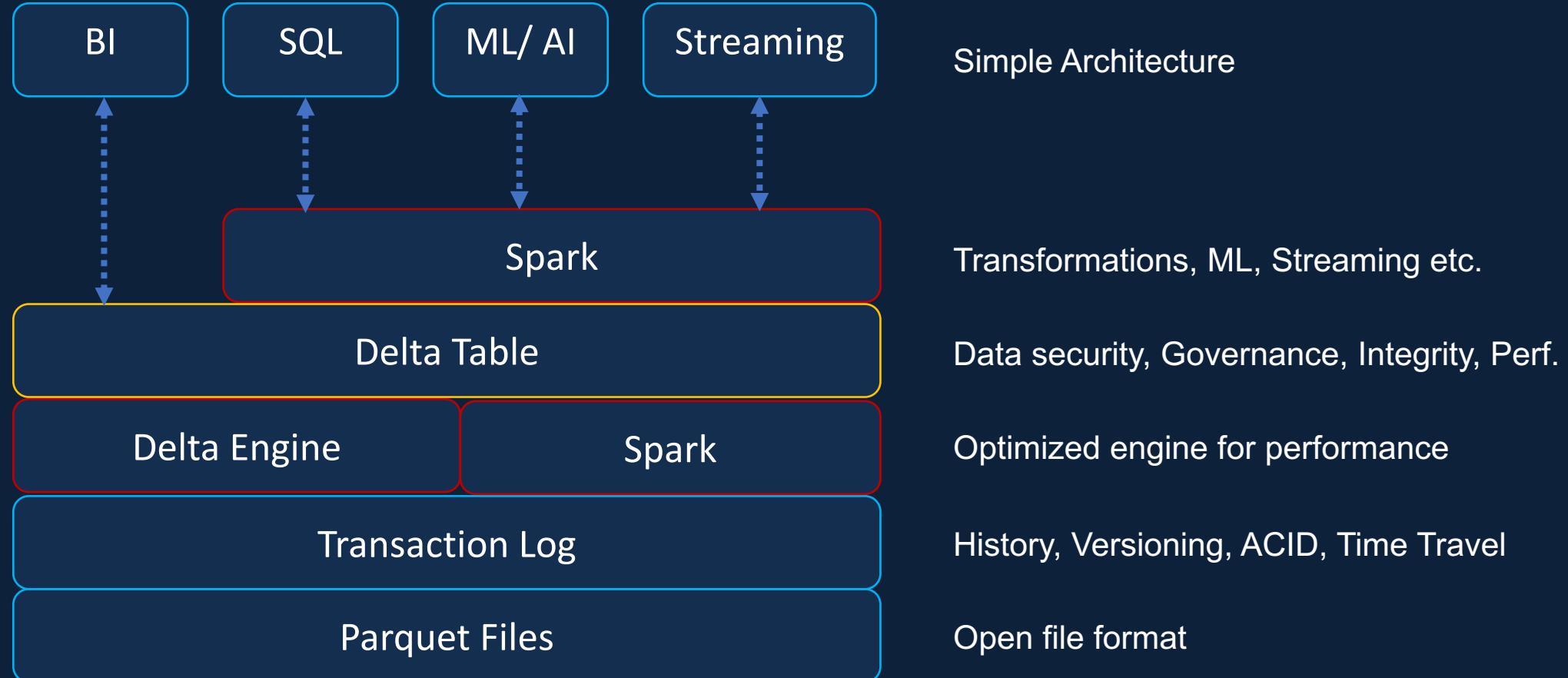


Data Lakehouse



- Handles all types of data
- Cheap cloud object storage
- Uses open source format
- Support for all types of workloads
- Ability to use BI tools directly
- ACID support
- History & Versioning
- Better performance
- Simple architecture

Delta Lake



Delta Lake Demo

Azure Data Factory



Overview

Creating Data Factory Service

Data Factory Components

Creating Pipelines

Creating Triggers

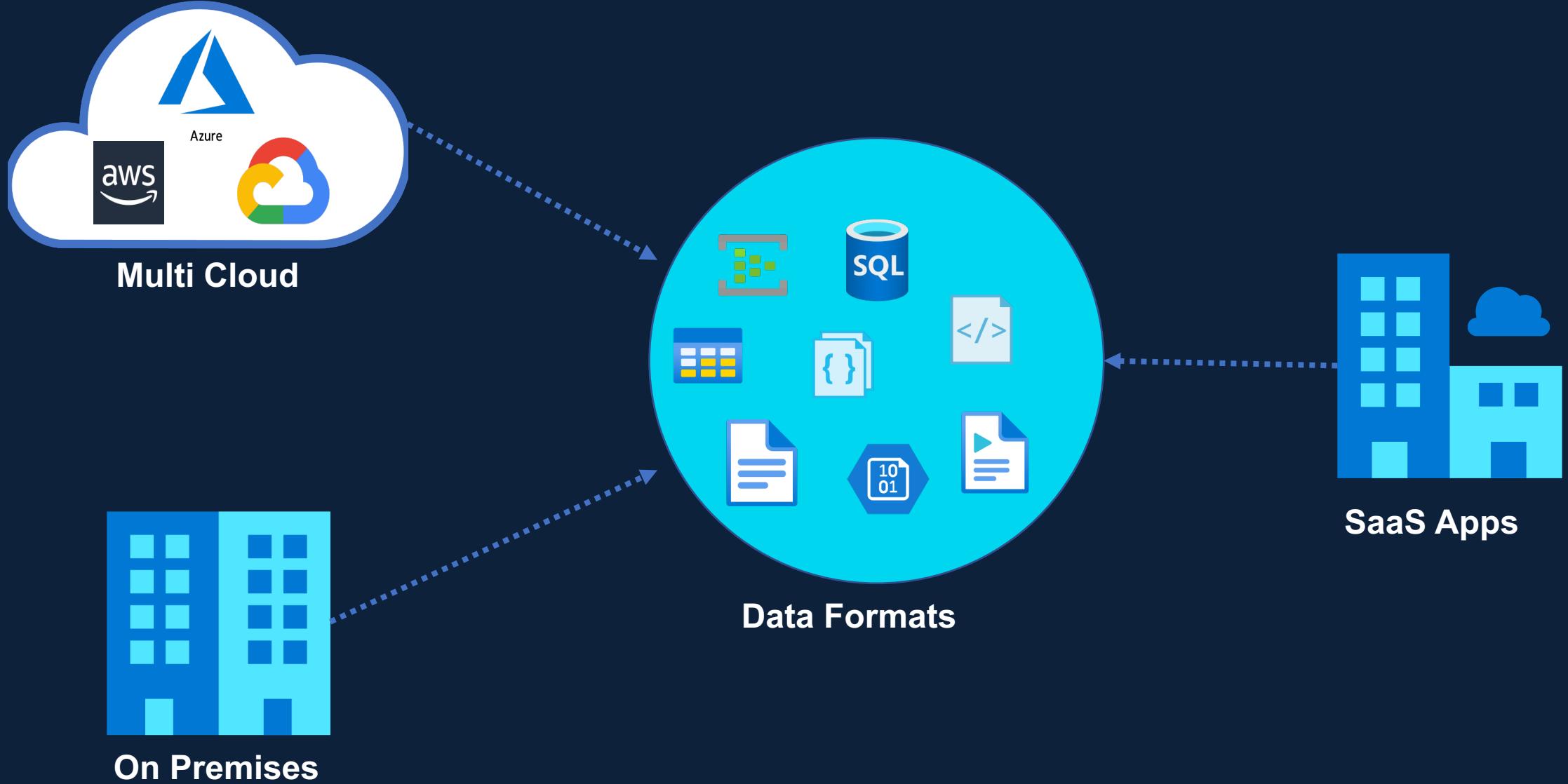
Azure Data Factory Overview

What is Azure Data Factory

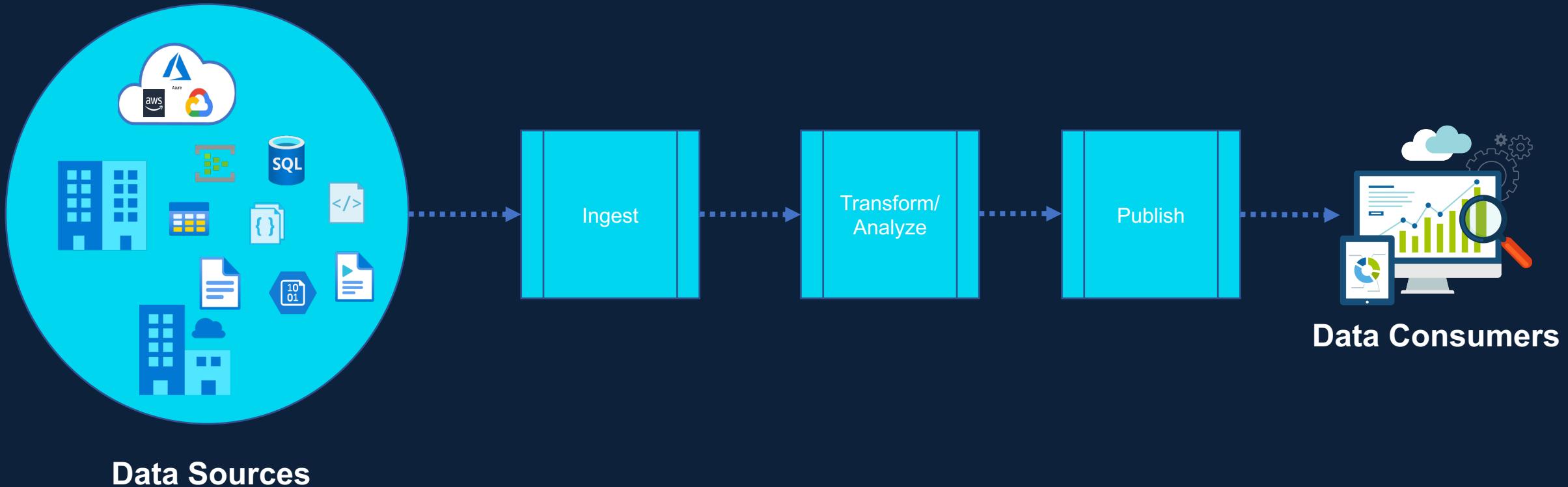


A fully managed, serverless data integration solution for ingesting, preparing and transforming all of your data at scale.

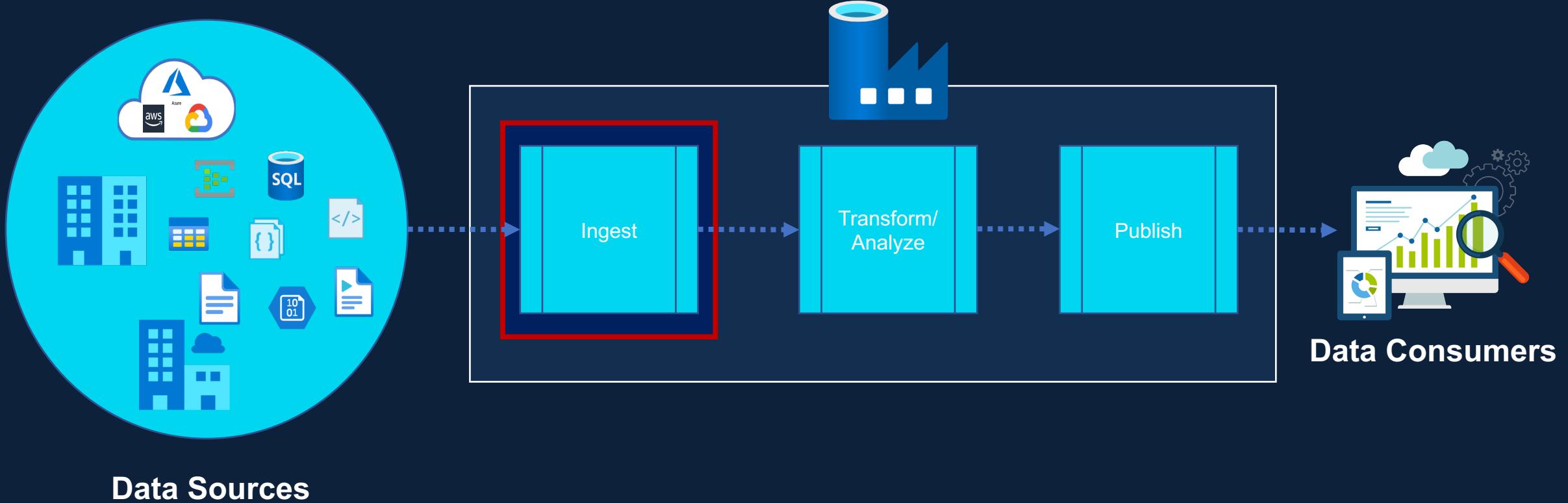
The Data Problem



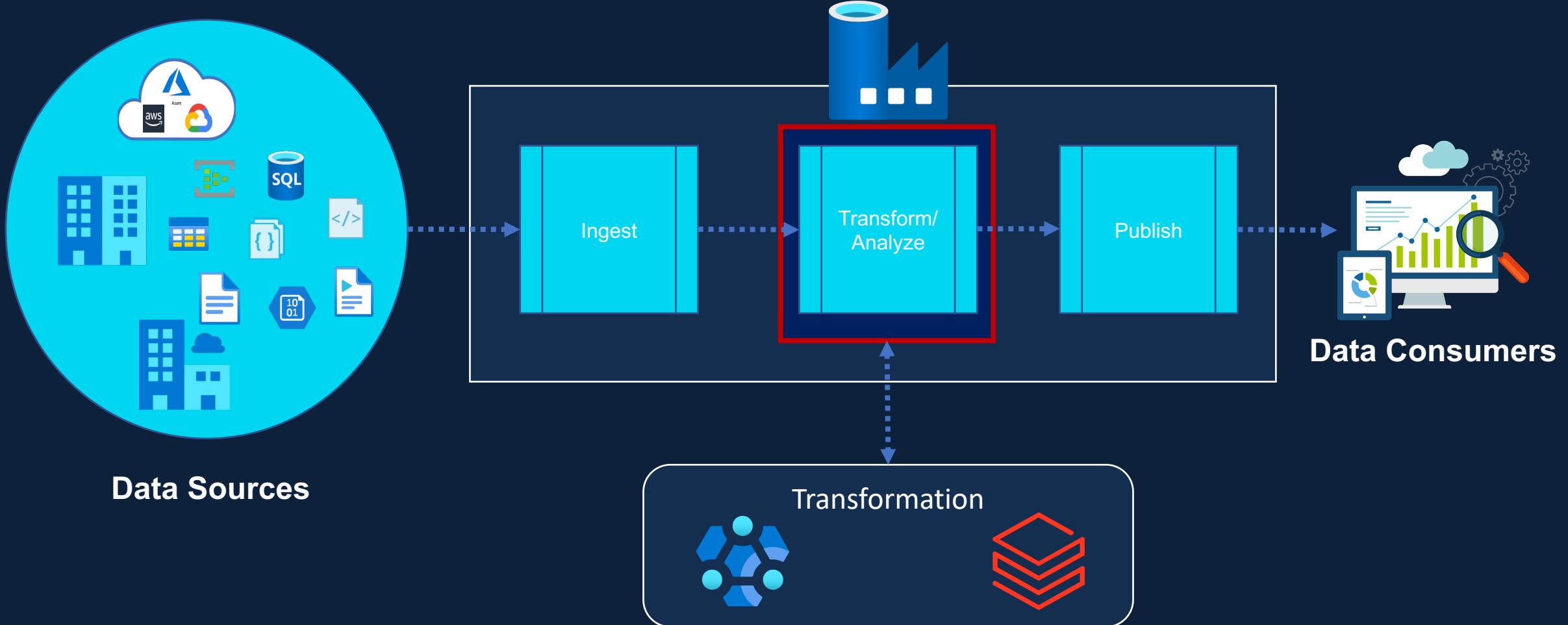
The Data Problem



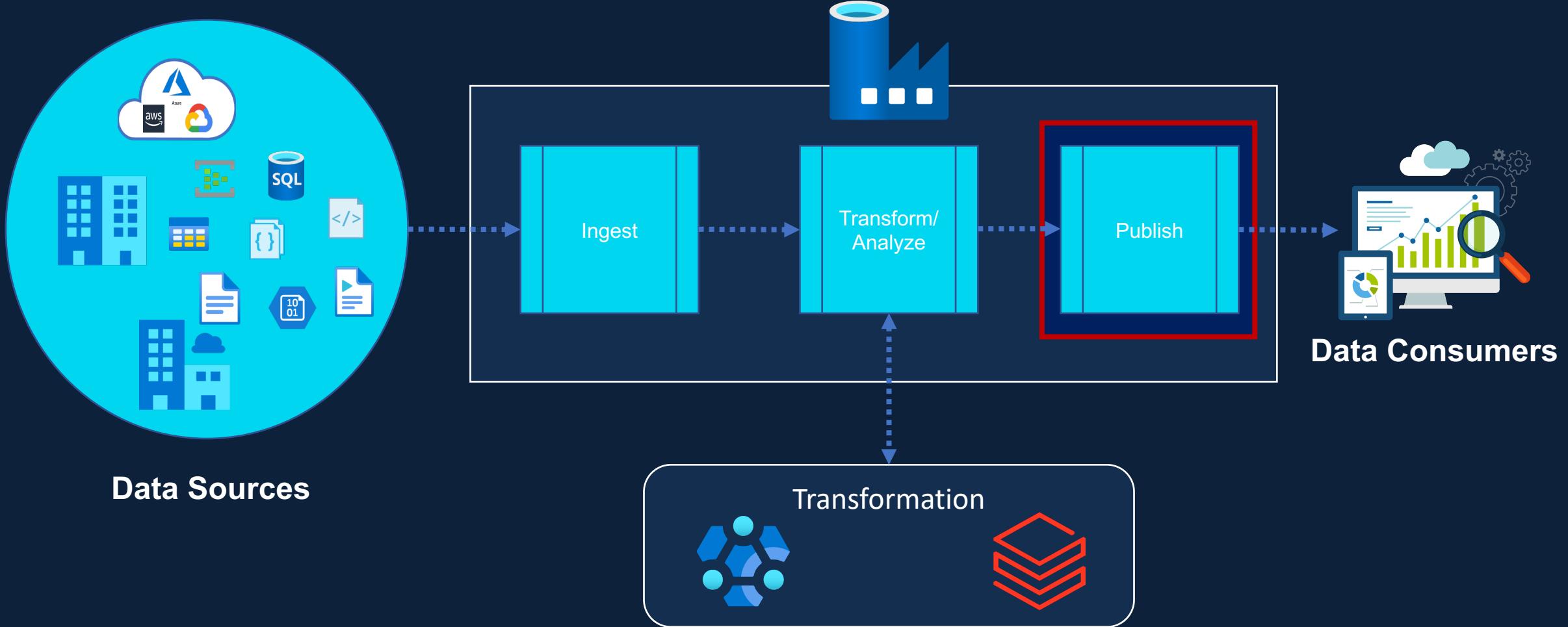
The Data Problem



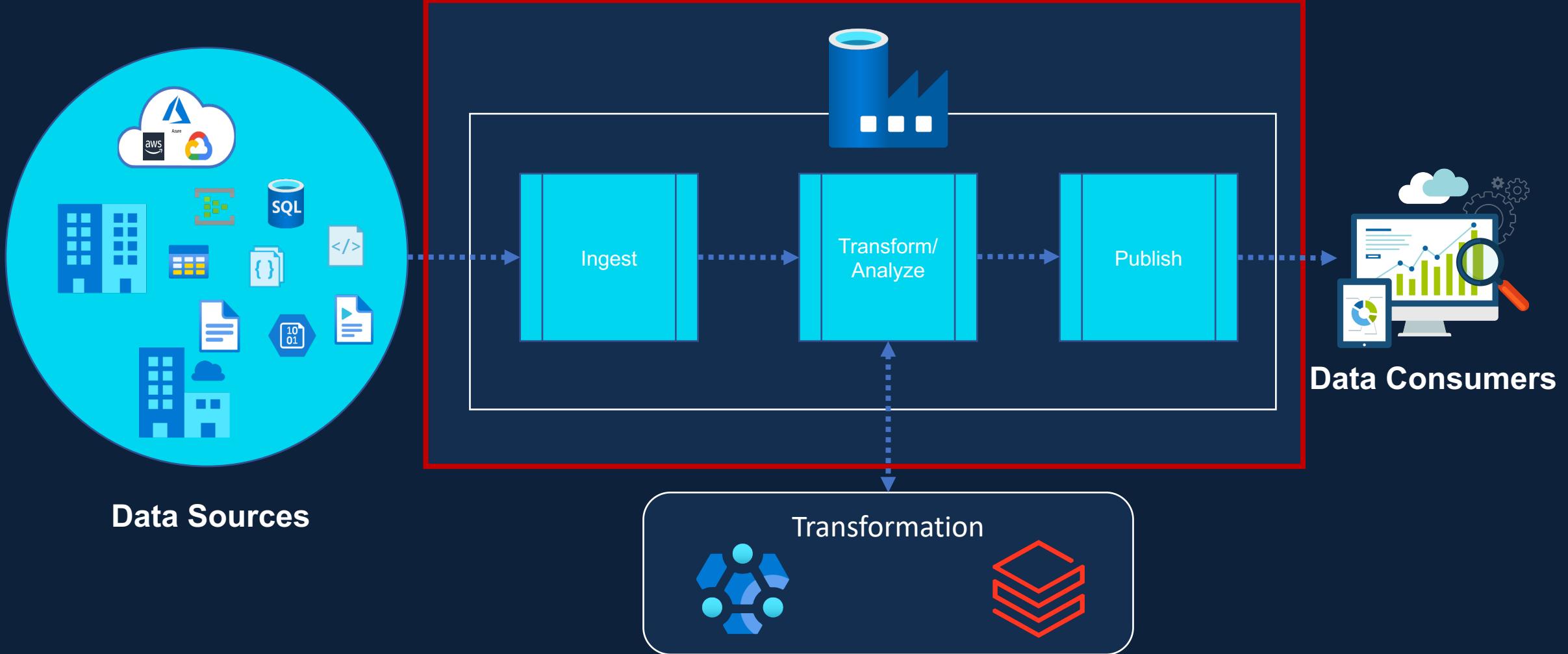
The Data Problem



The Data Problem



The Data Problem



What is Azure Data Factory



Fully Managed Service

Serverless

Data Integration Service

Data Transformation Service

Data Orchestration Service

A fully managed, serverless data integration solution for ingesting, preparing and transforming all of your data at scale.

What Azure Data Factory Is Not



Data Migration Tool

Data Streaming Service

Suitable for Complex Data Transformations

Data Storage Service

Create Data Factory Service

Data Factory Components

Trigger

Pipeline

Activity

Activity

Dataset

Linked Service

Linked Service

Storage

ADLS

SQL
Database

Compute

Azure
Databricks

Azure
HDInsight

Connecting from Power BI

Unity Catalog - Introduction



Unity Catalog Overview

Enable Unity Catalog Metastore

Cluster Configurations

Unity Catalog Object Model

Access External Data Lake

Unity Catalog



Unity Catalog



Unity Catalog is a Databricks offered unified solution for implementing data governance in the Data Lakehouse

Unity Catalog



Unity Catalog

Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Data Governance

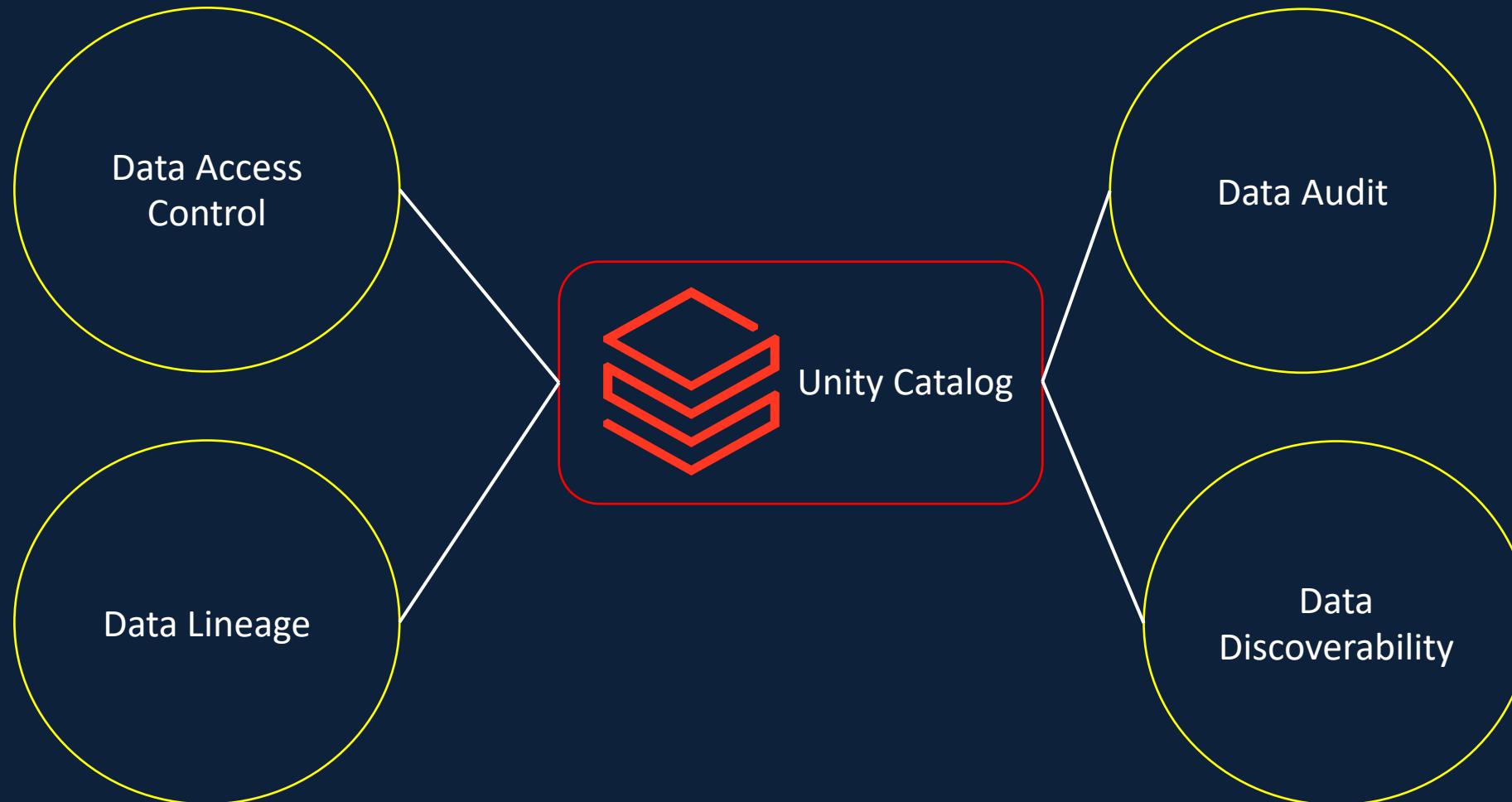
Is the process of managing the availability, usability, integrity and security of the data present in an enterprise

Controls access to the data for the users

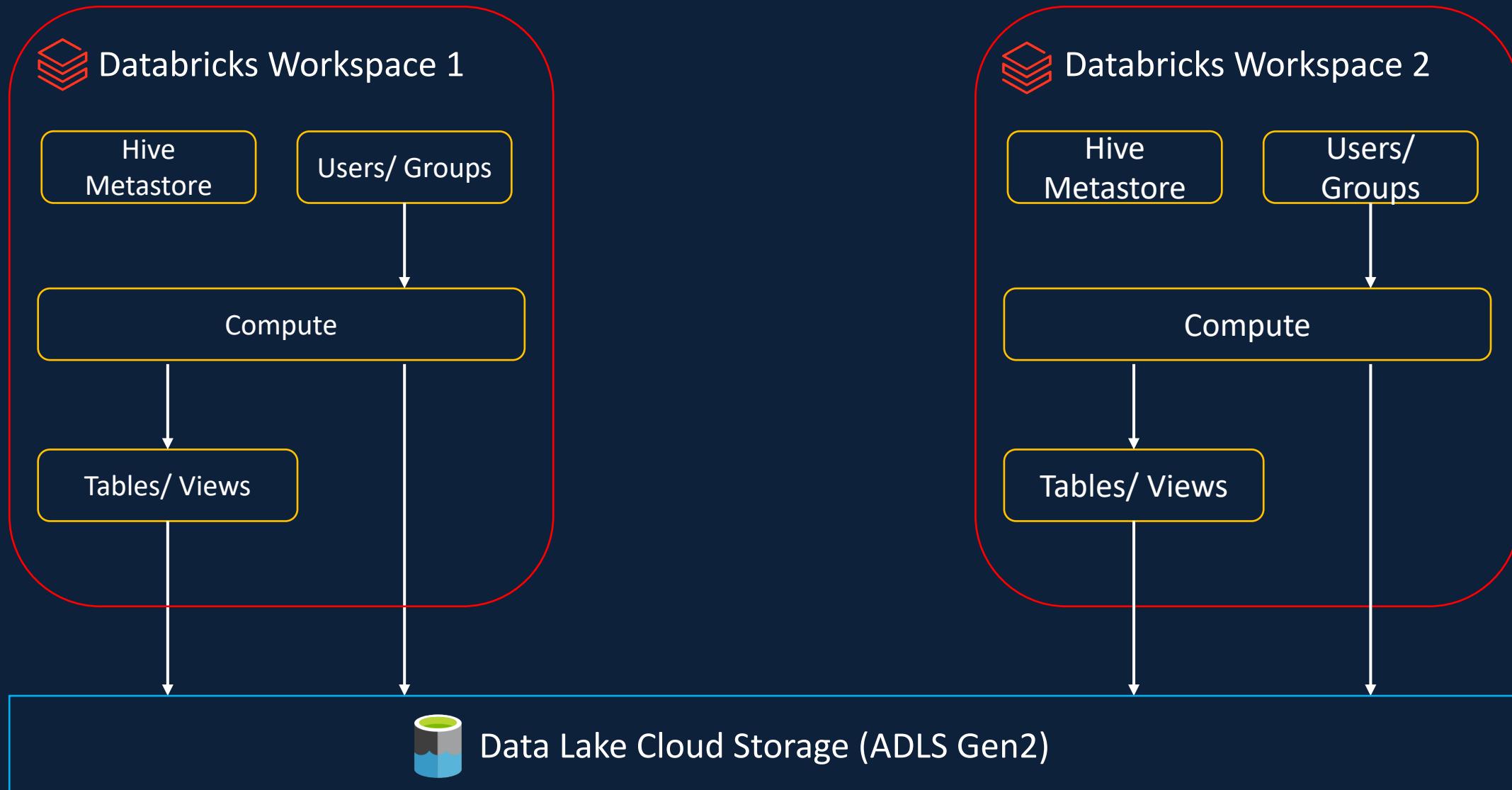
Ensures that the data is trust worthy and not misused

Helps implement privacy regulations such as GDPR, CCPA etc

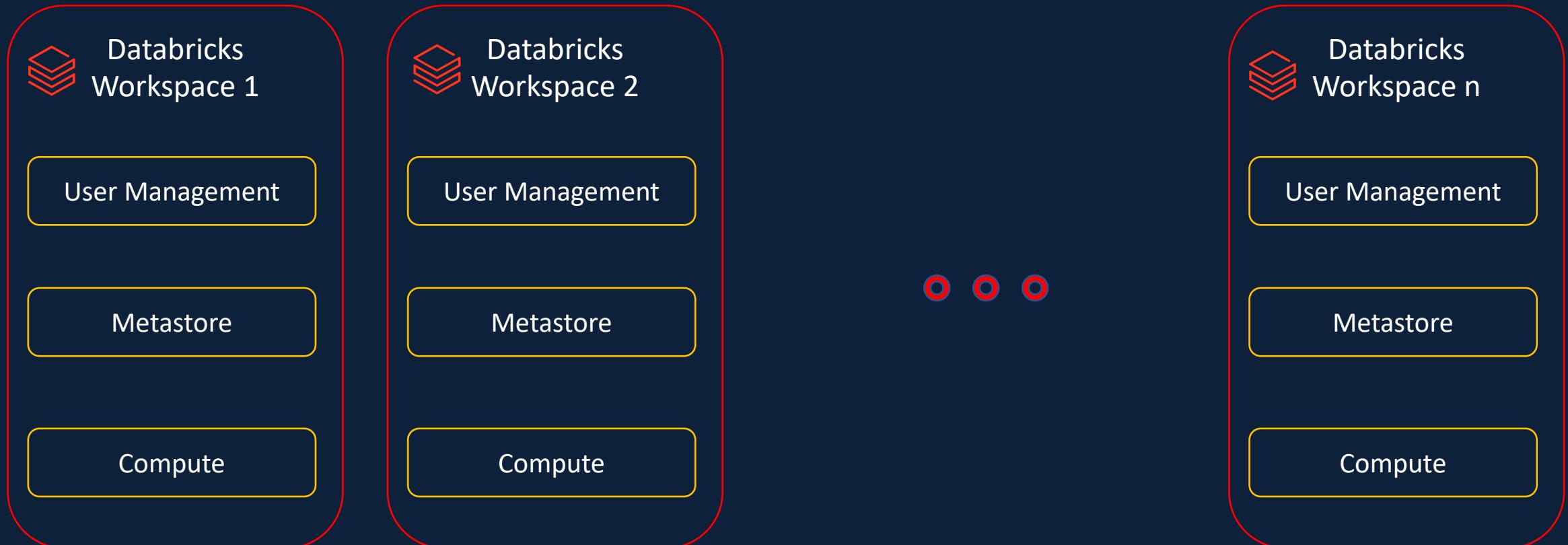
Data Governance



Without Unity Catalog



Without Unity Catalog



With Unity Catalog

Without Unity Catalog

Databricks
Workspace 1

User Management

Metastore

Compute

Databricks
Workspace 2

User Management

Metastore

Compute

Databricks Account

With Unity Catalog

Databricks Unity Catalog

User Management

Metastore

Databricks
Workspace 1

Compute

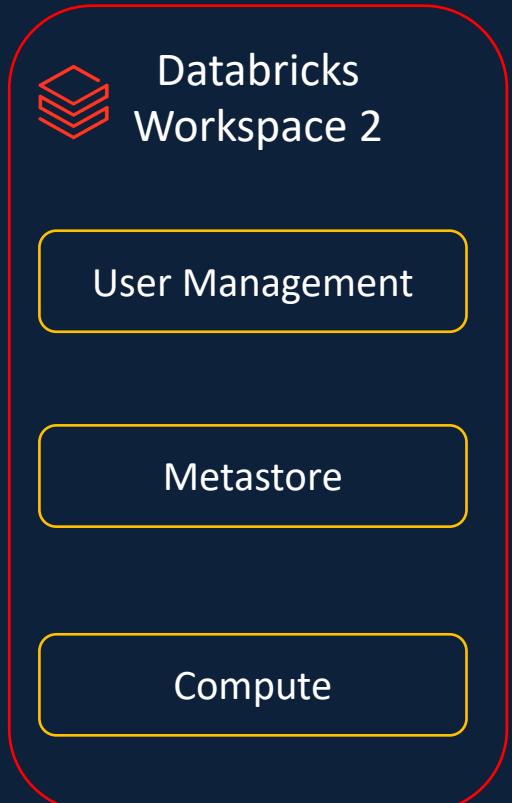
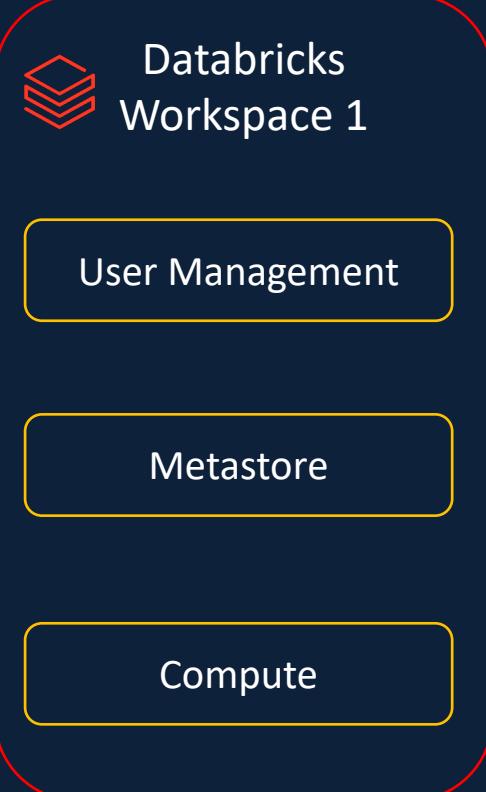
Databricks
Workspace 2

Compute

Databricks Account

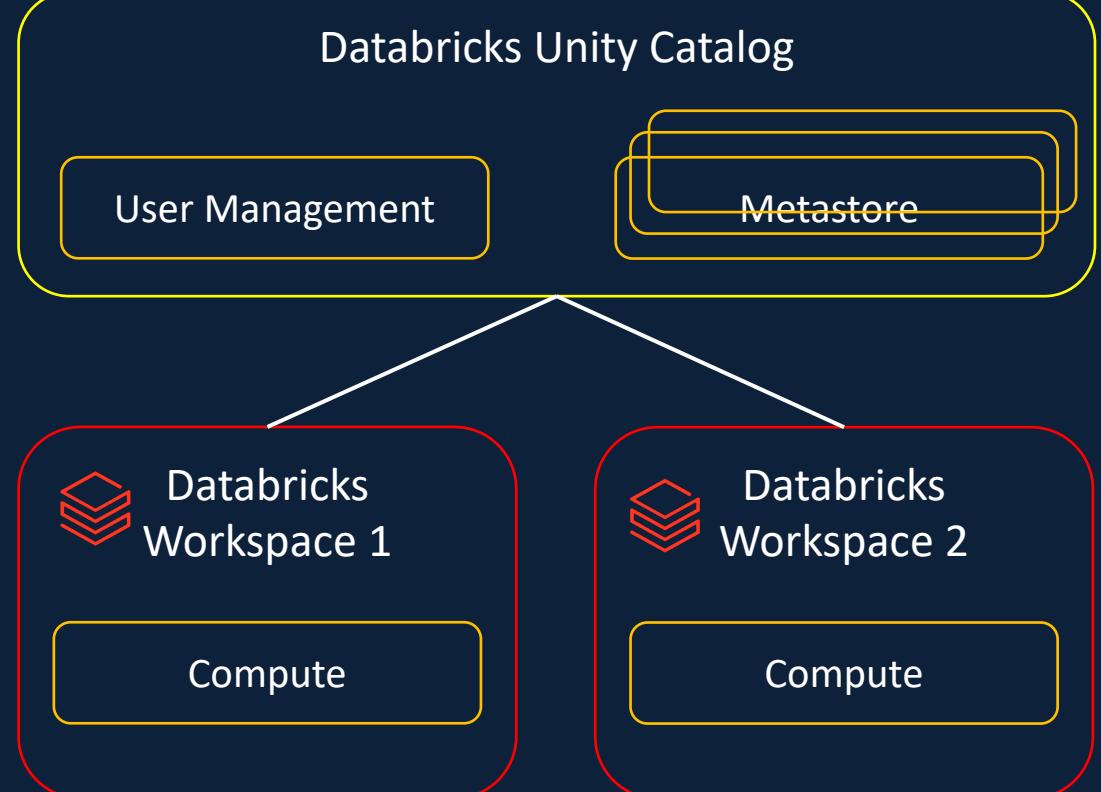
With Unity Catalog

Without Unity Catalog



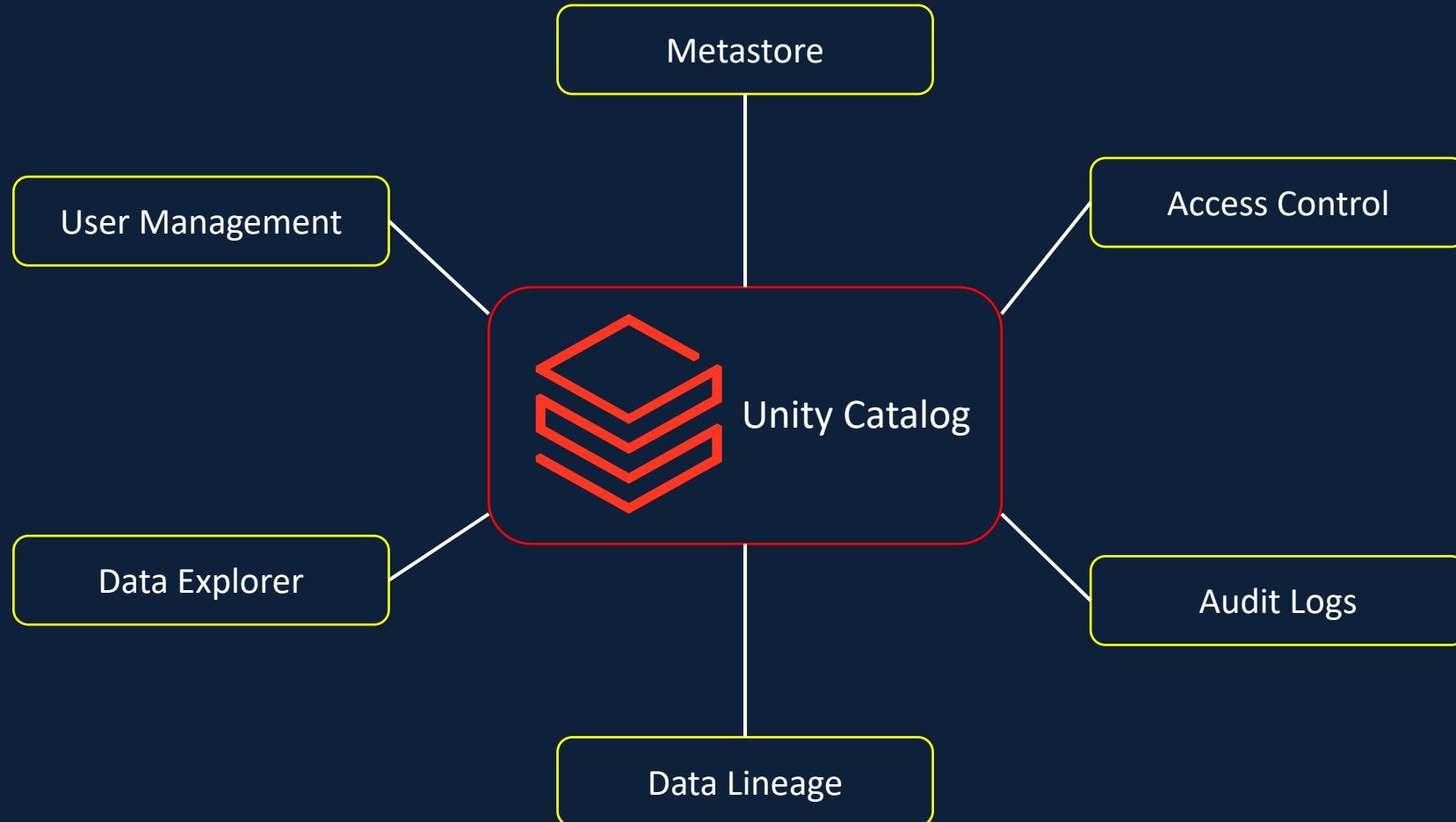
Databricks Account

With Unity Catalog

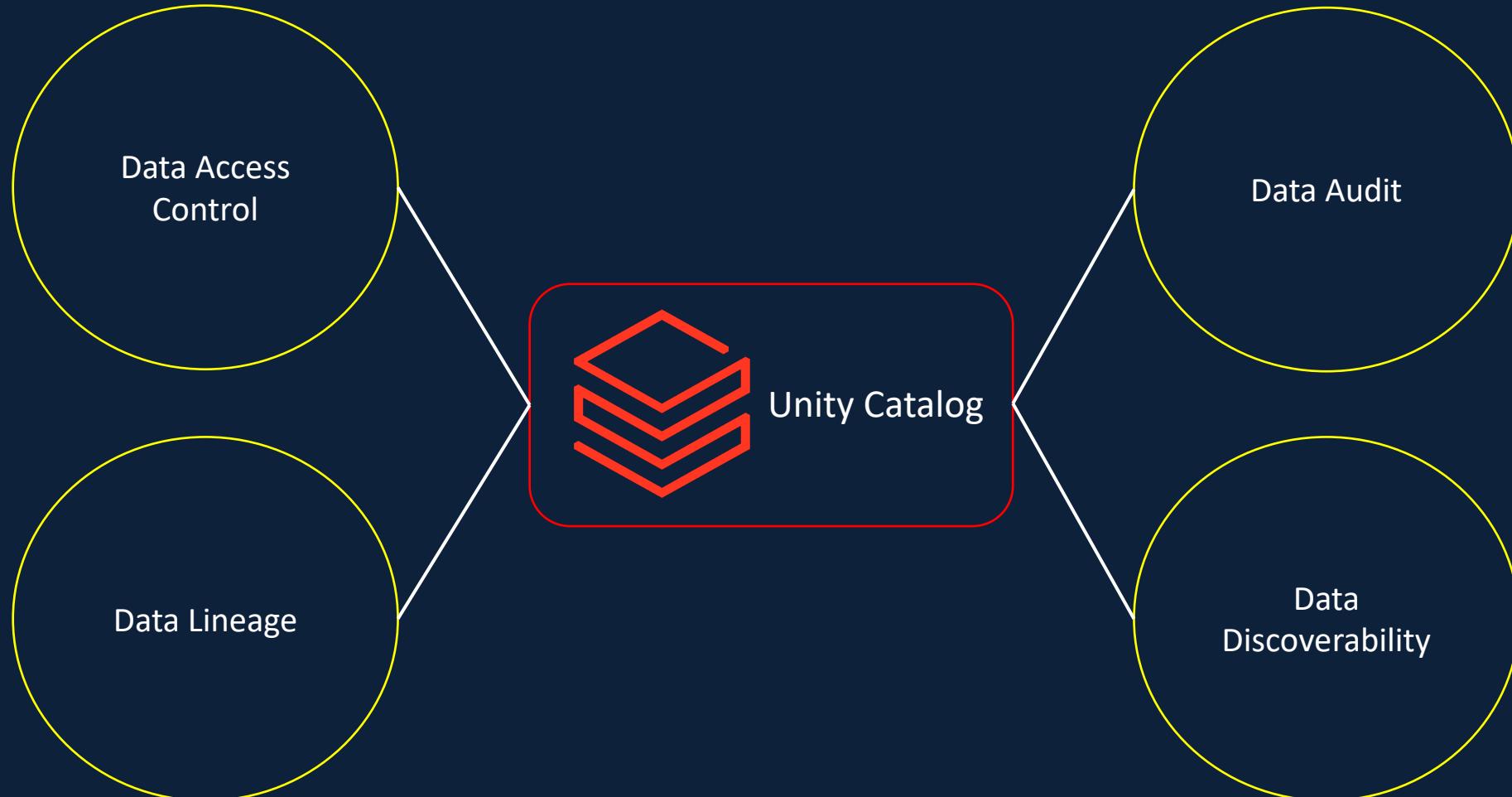


Databricks Account

Unity Catalog

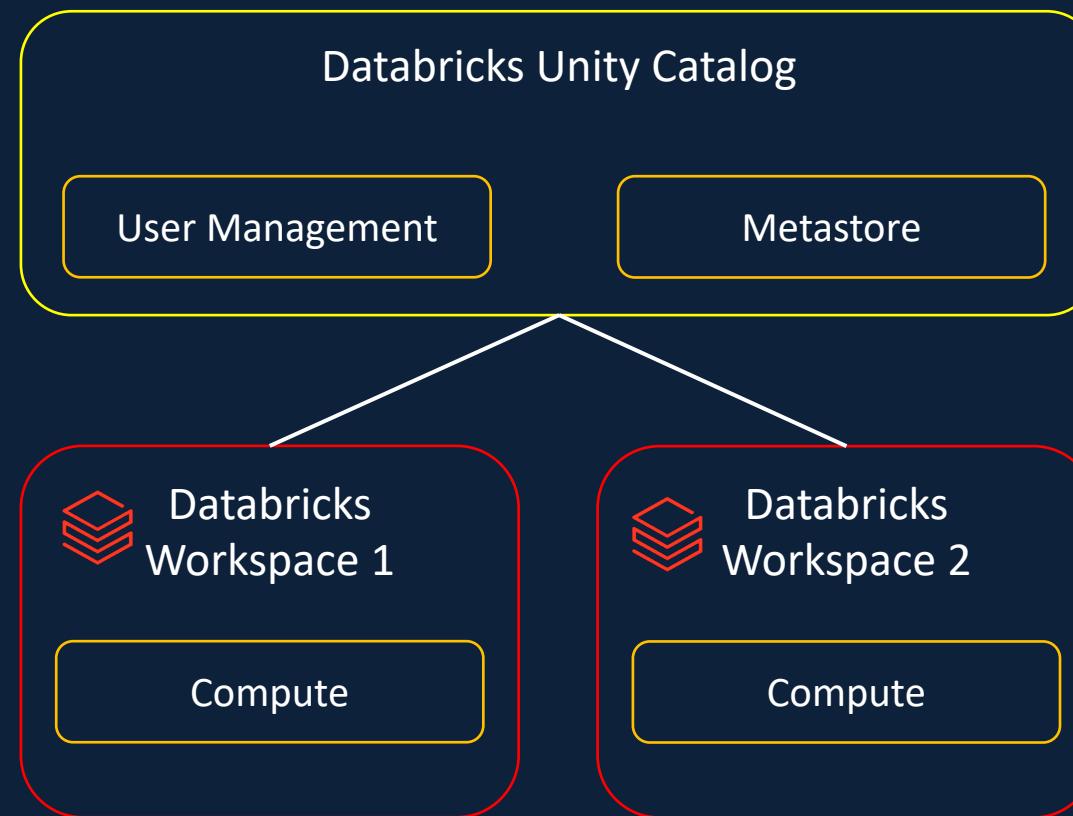


Unity Catalog

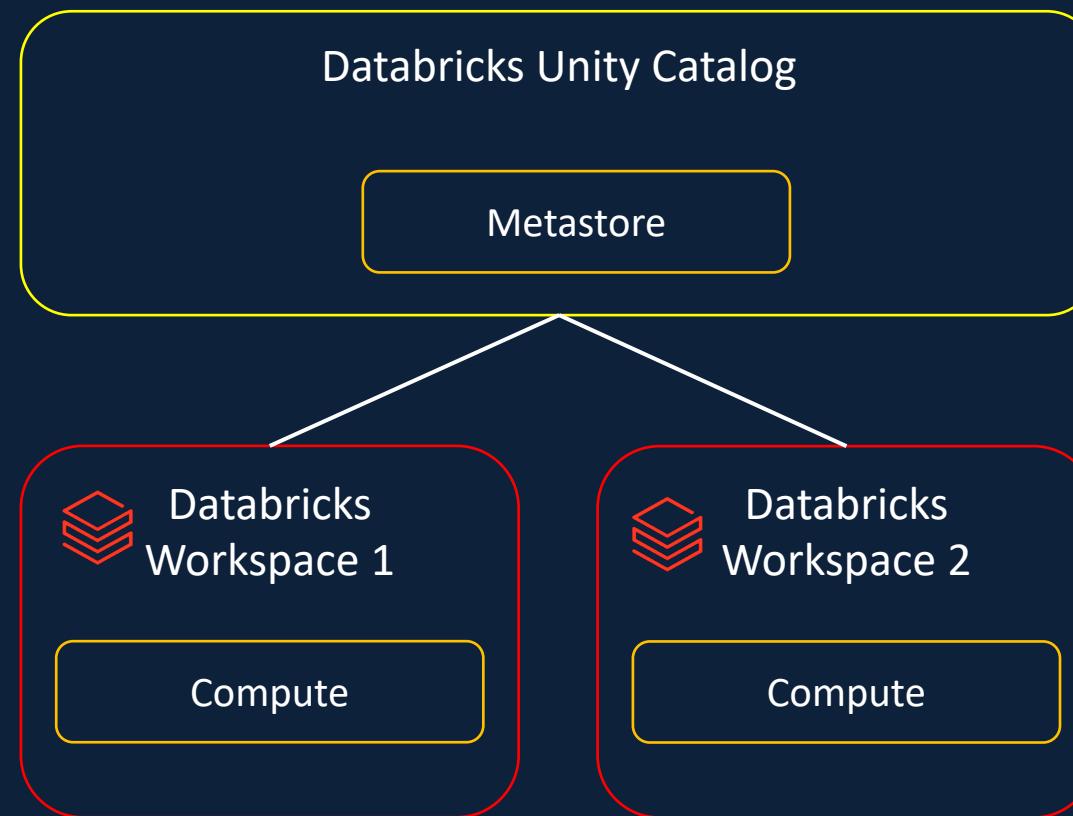


Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

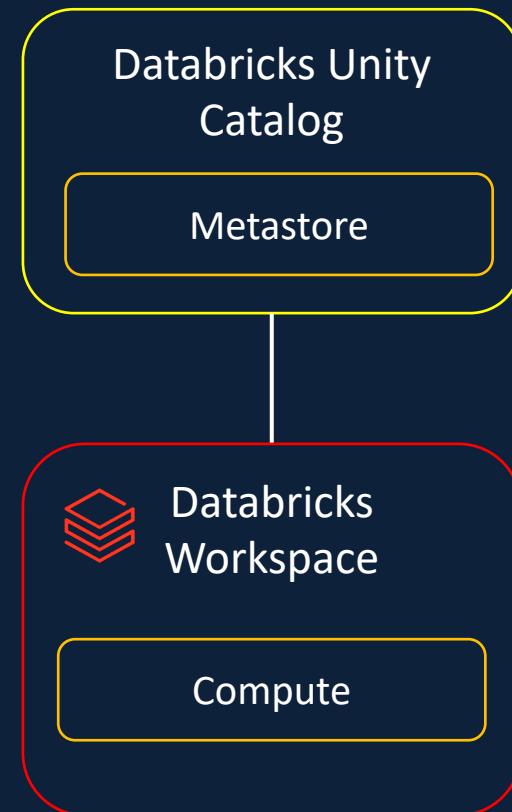
Unity Catalog Set-up



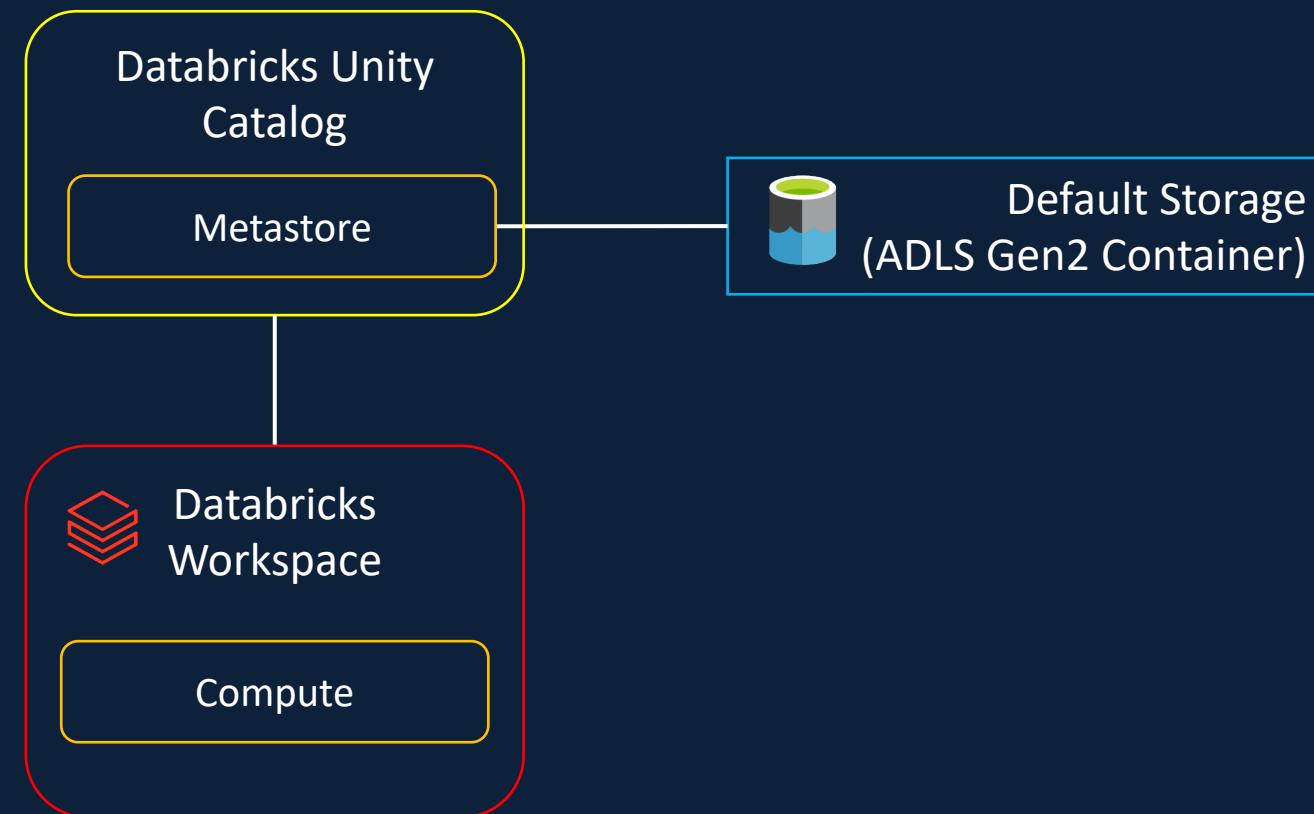
Unity Catalog Set-up



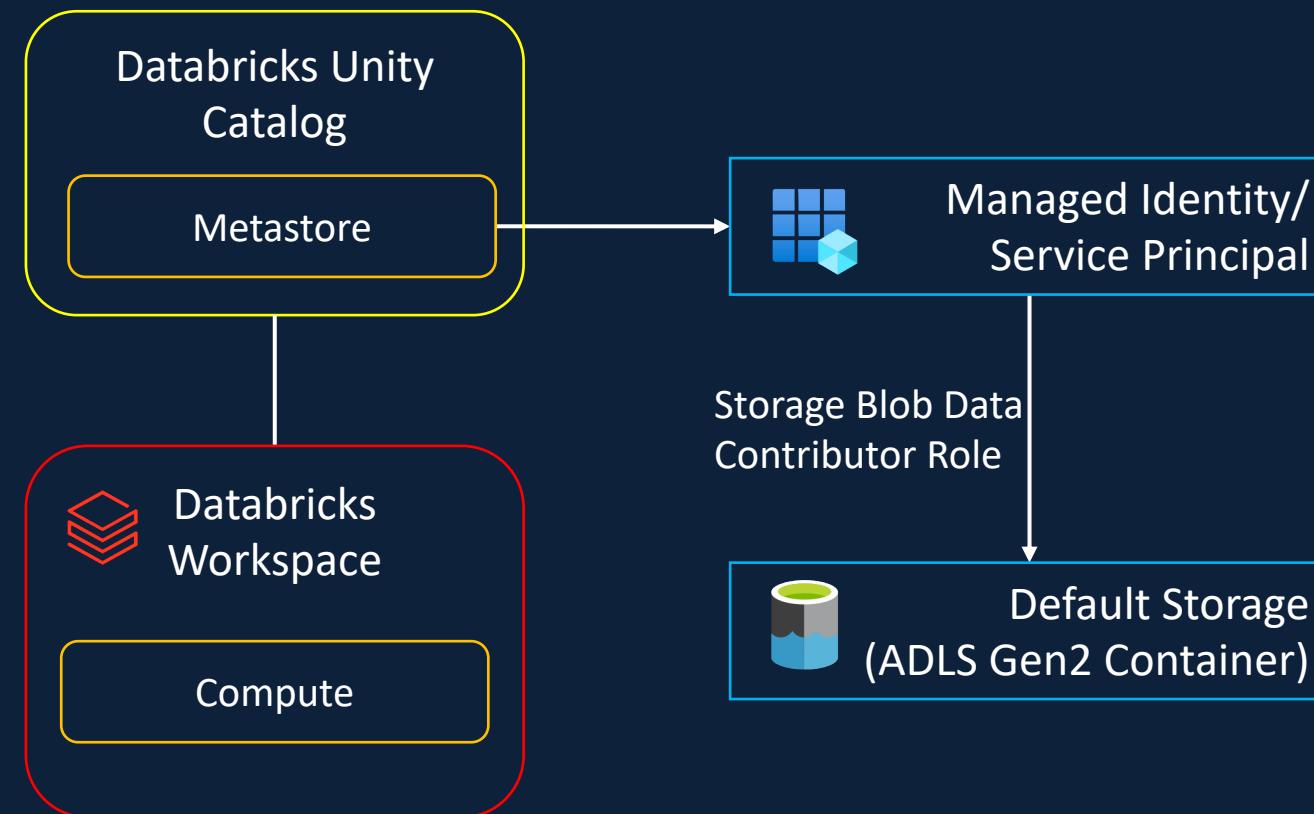
Unity Catalog Set-up



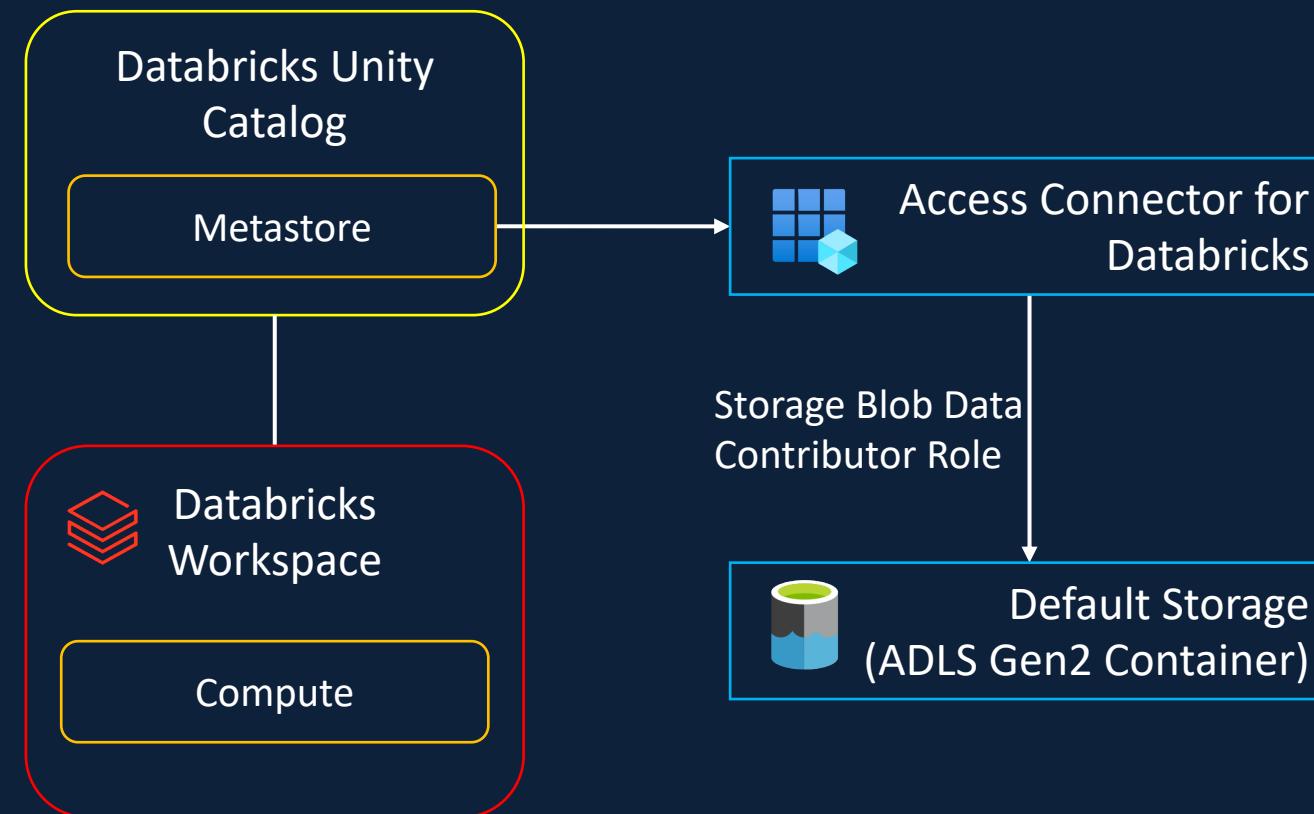
Unity Catalog Set-up



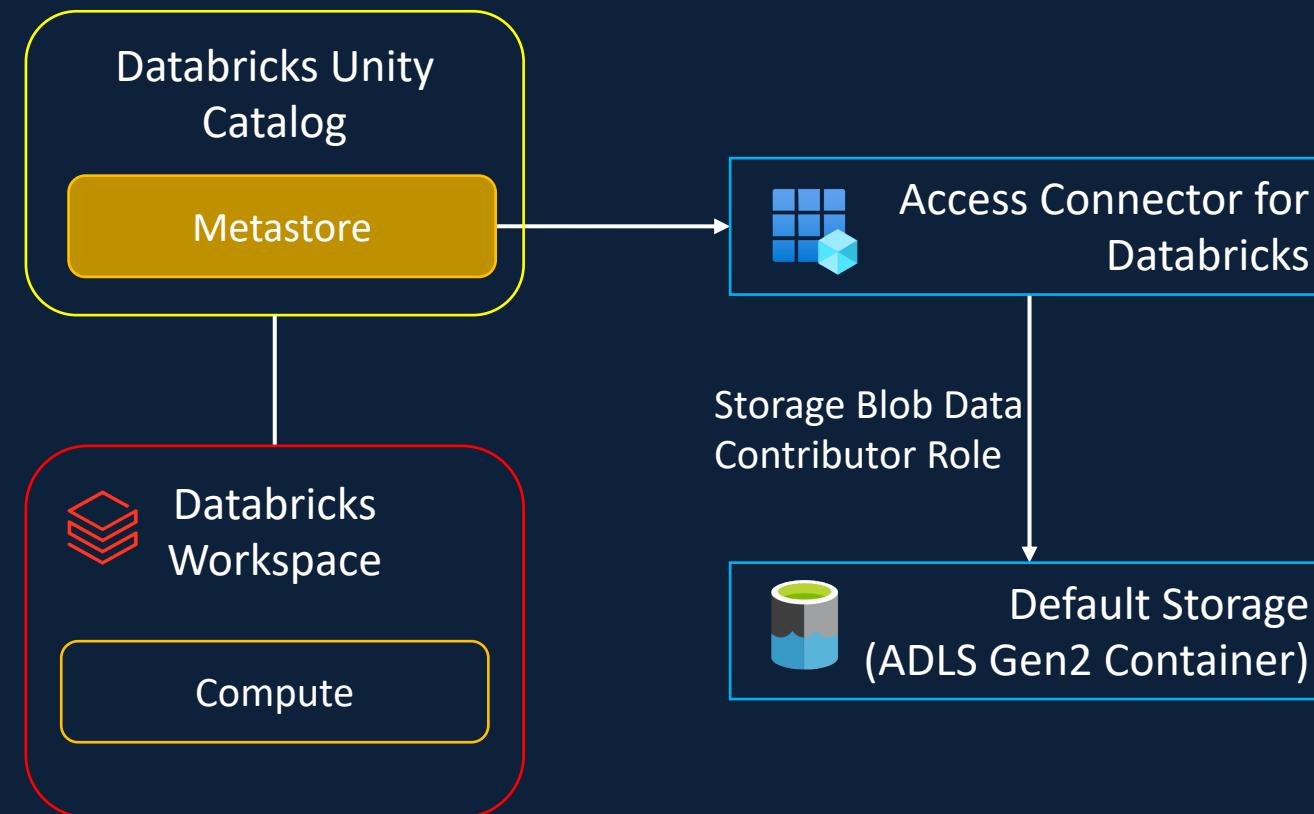
Unity Catalog Set-up



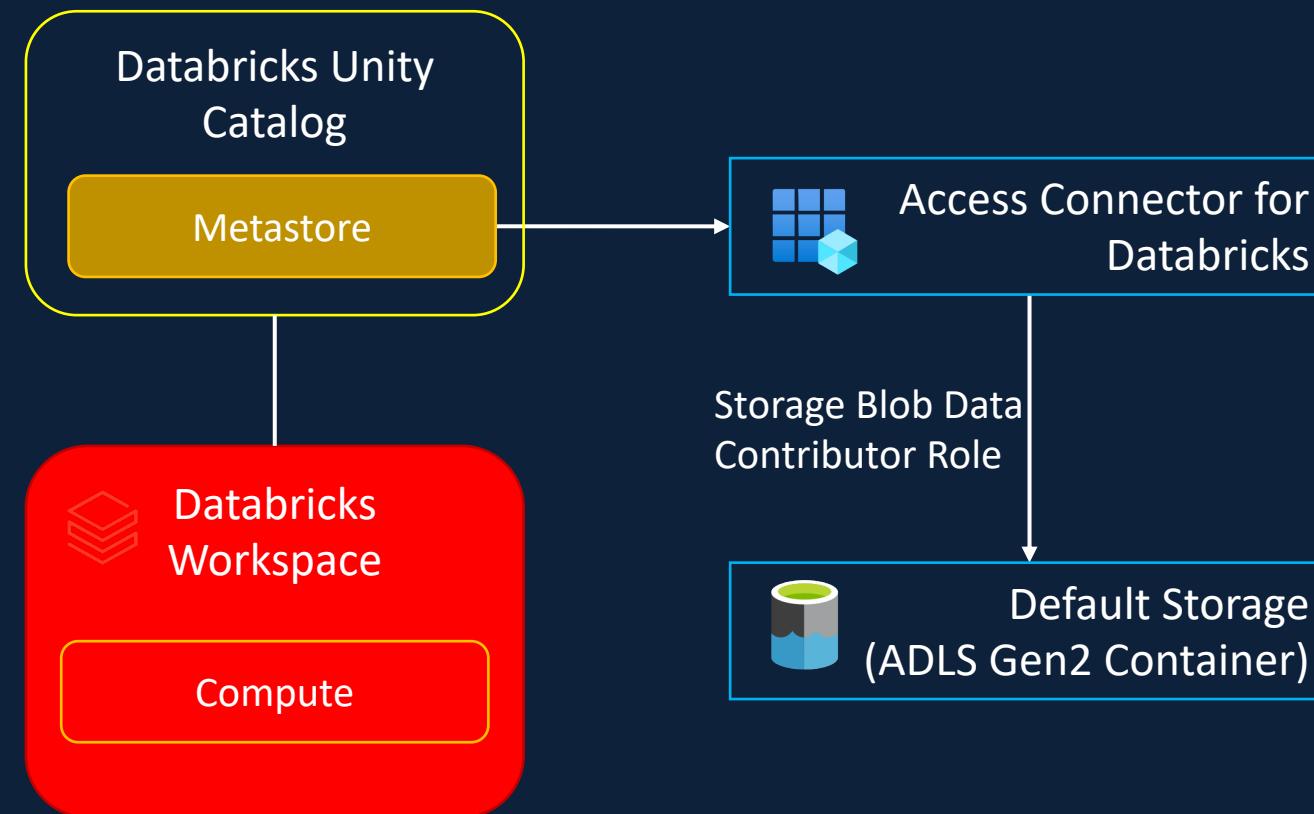
Unity Catalog Set-up



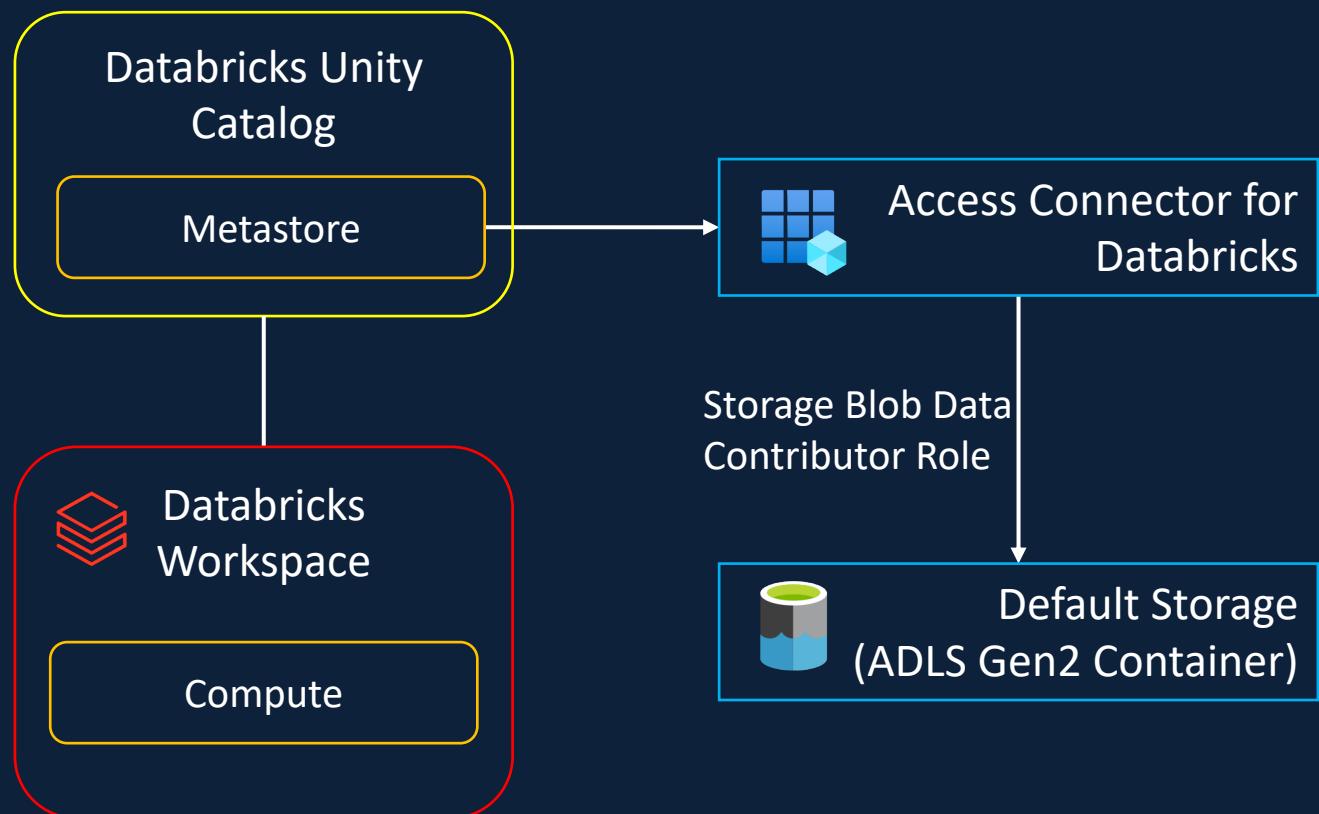
Unity Catalog Set-up



Unity Catalog Set-up



Unity Catalog Set-up



Create Azure Databricks Workspace

Create Azure Data Lake Gen2

Create Access Connector

Add role Storage Blob Data Contributor

Create Unit Catalog Metastore

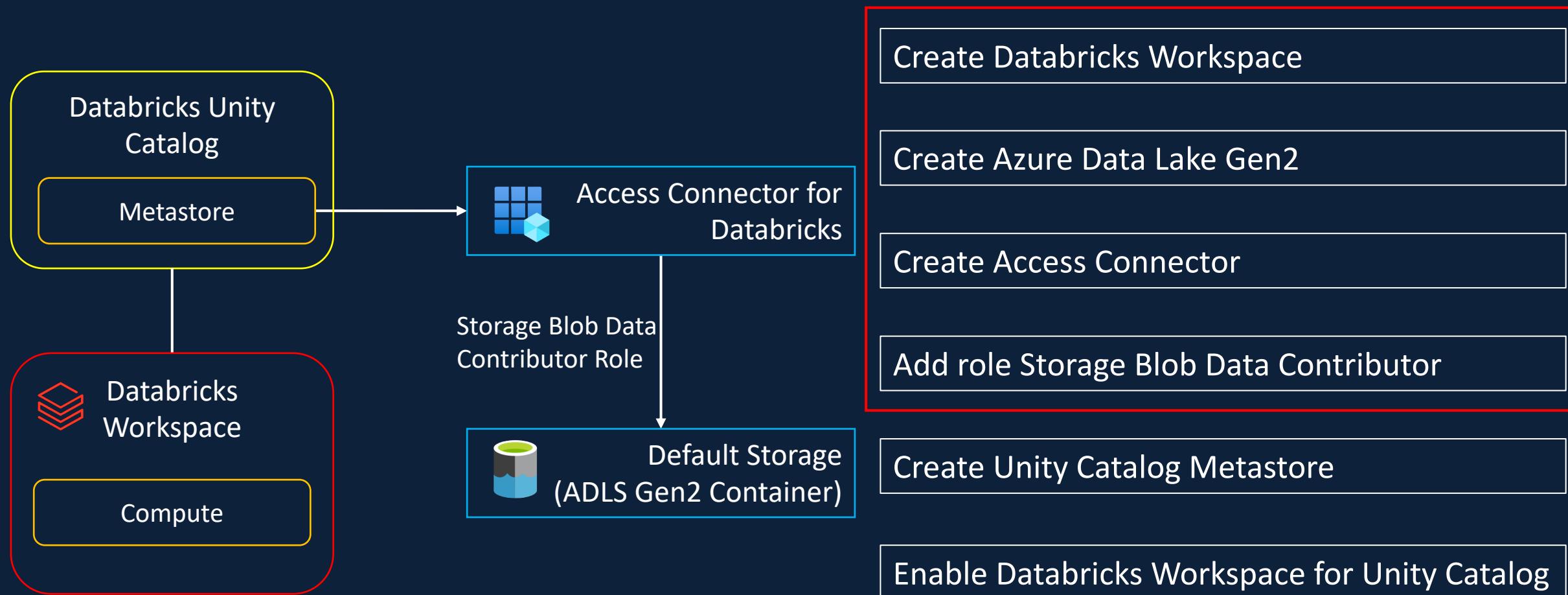
Enable Databricks Workspace for Unity Catalog

Unity Catalog Set-up

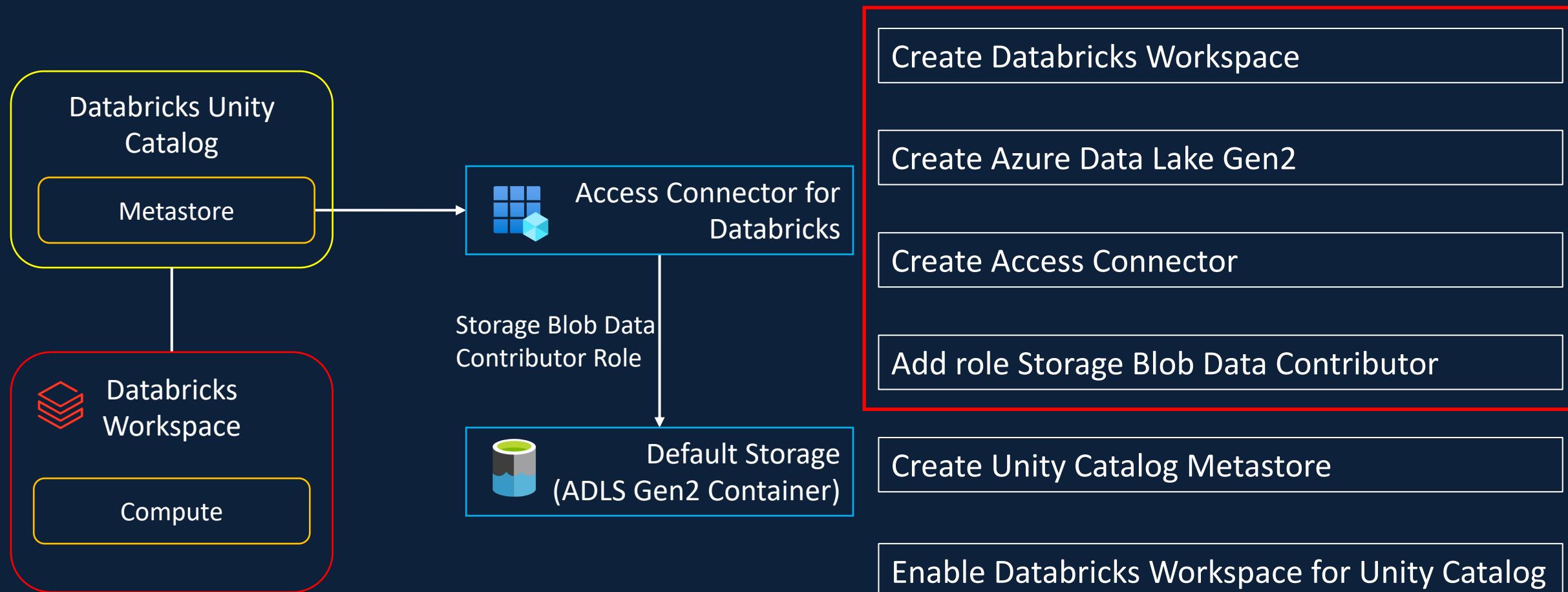
Pre-requisites



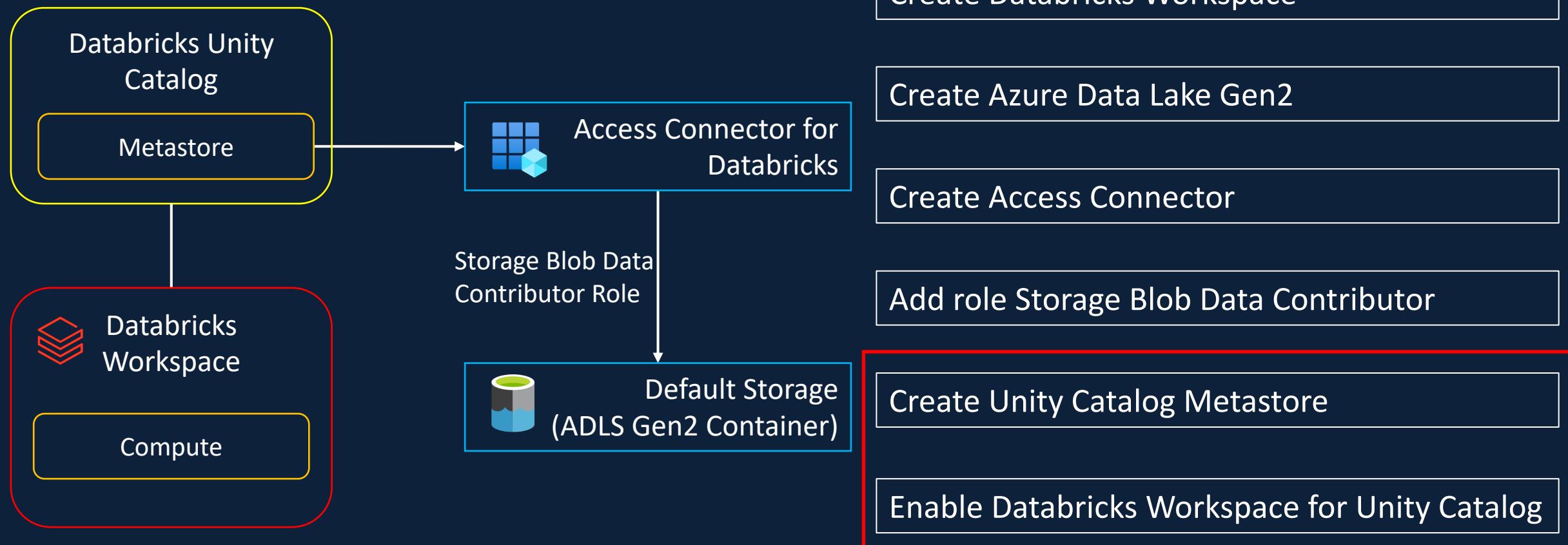
Unity Catalog Set-up



Unity Catalog Set-up



Unity Catalog Set-up



Cluster Configuration



Unity Catalog Object Model



Unity Catalog Object Model

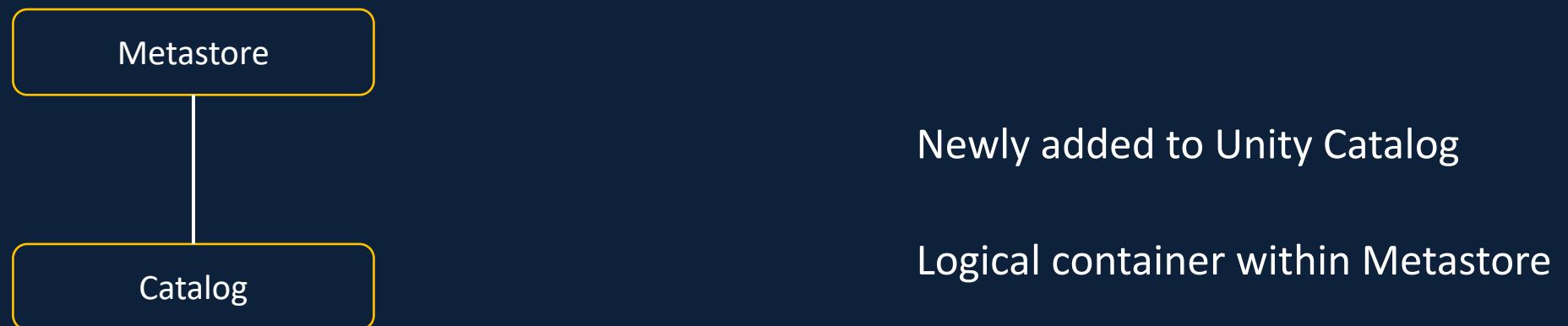
Metastore

Top level container

Only one per region

Paired with Default ADLS Storage

Unity Catalog Object Model



Unity Catalog Object Model

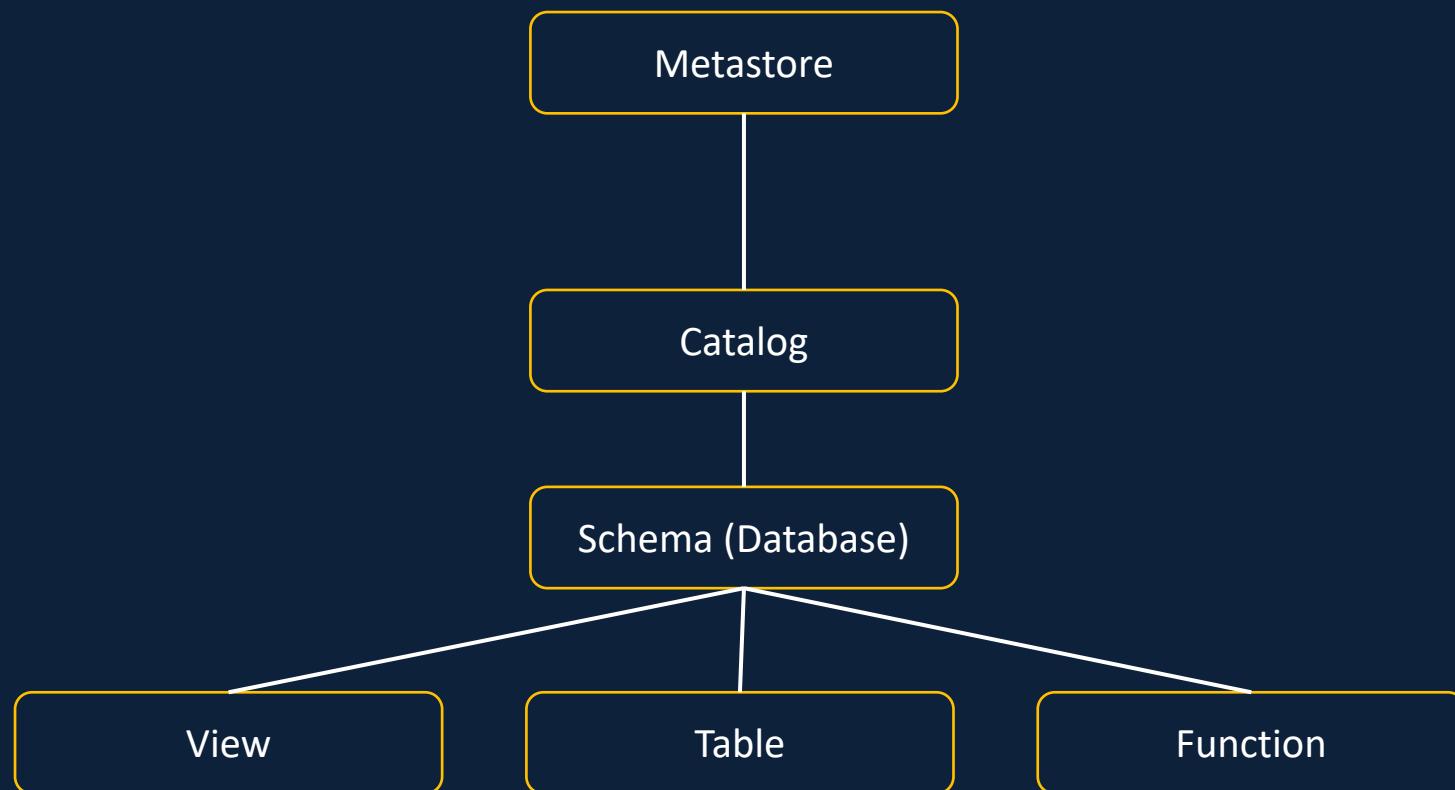


Next level container within Catalogs

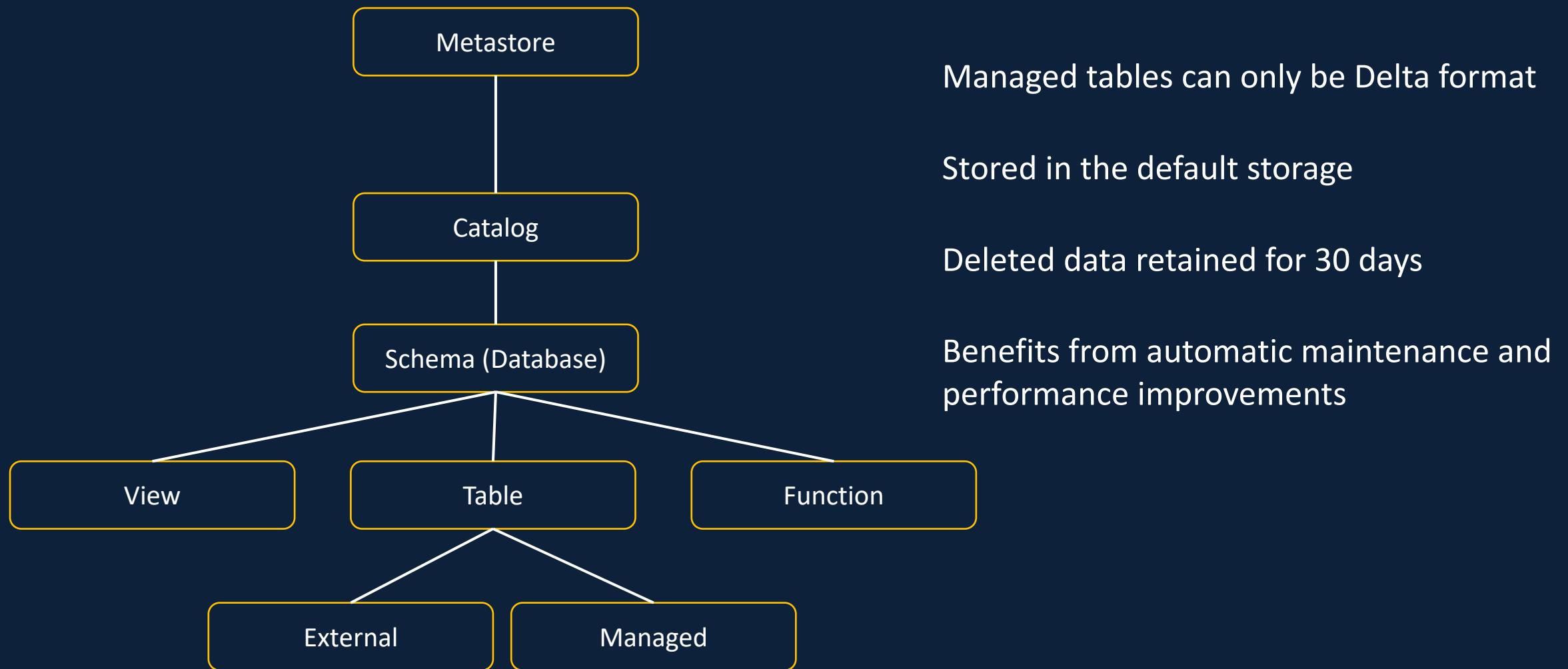
Schemas and Databases are the same

Use Schema instead of Database

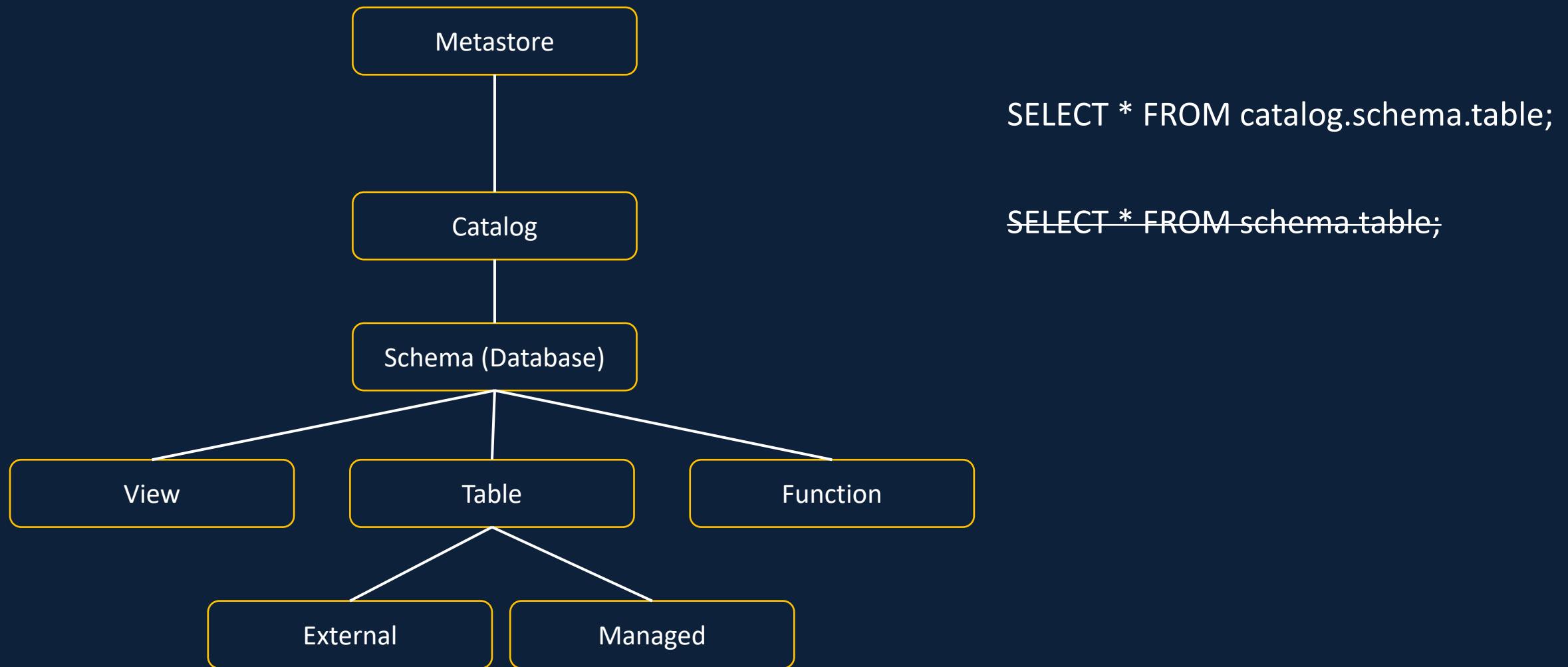
Unity Catalog Object Model



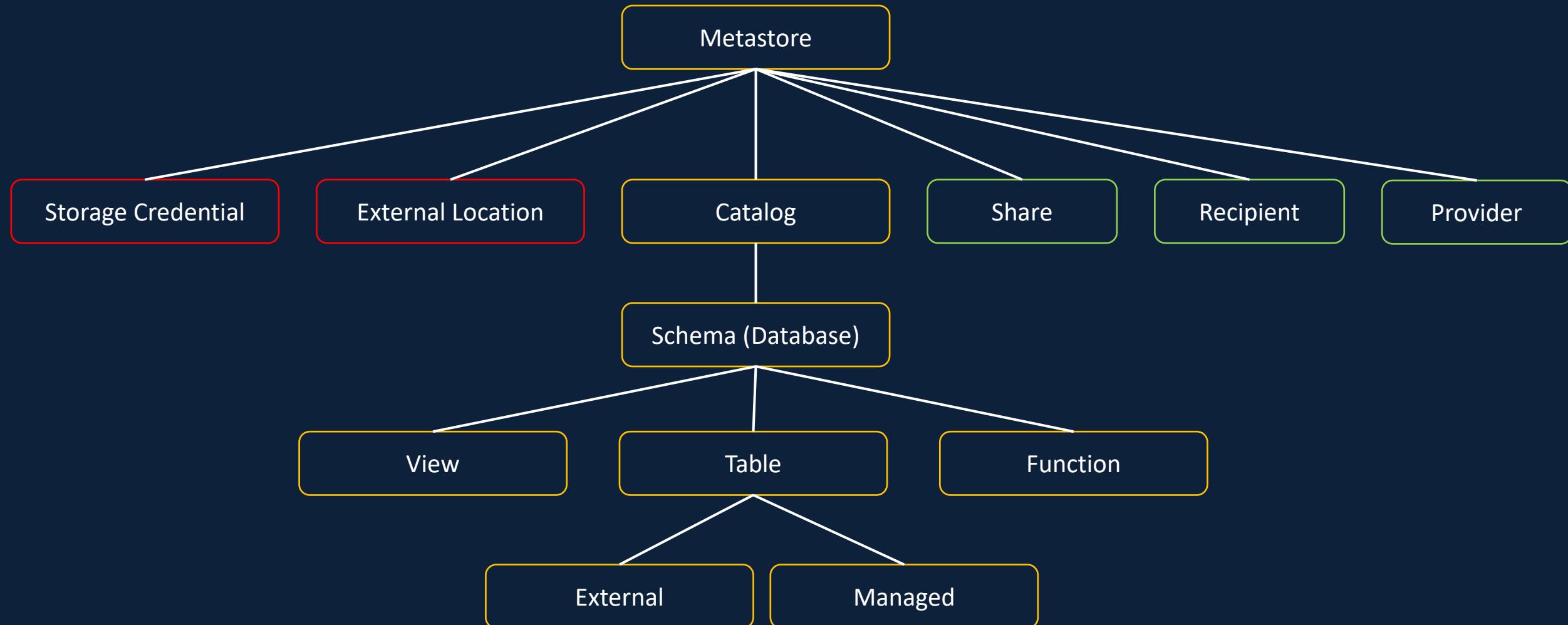
Unity Catalog Object Model



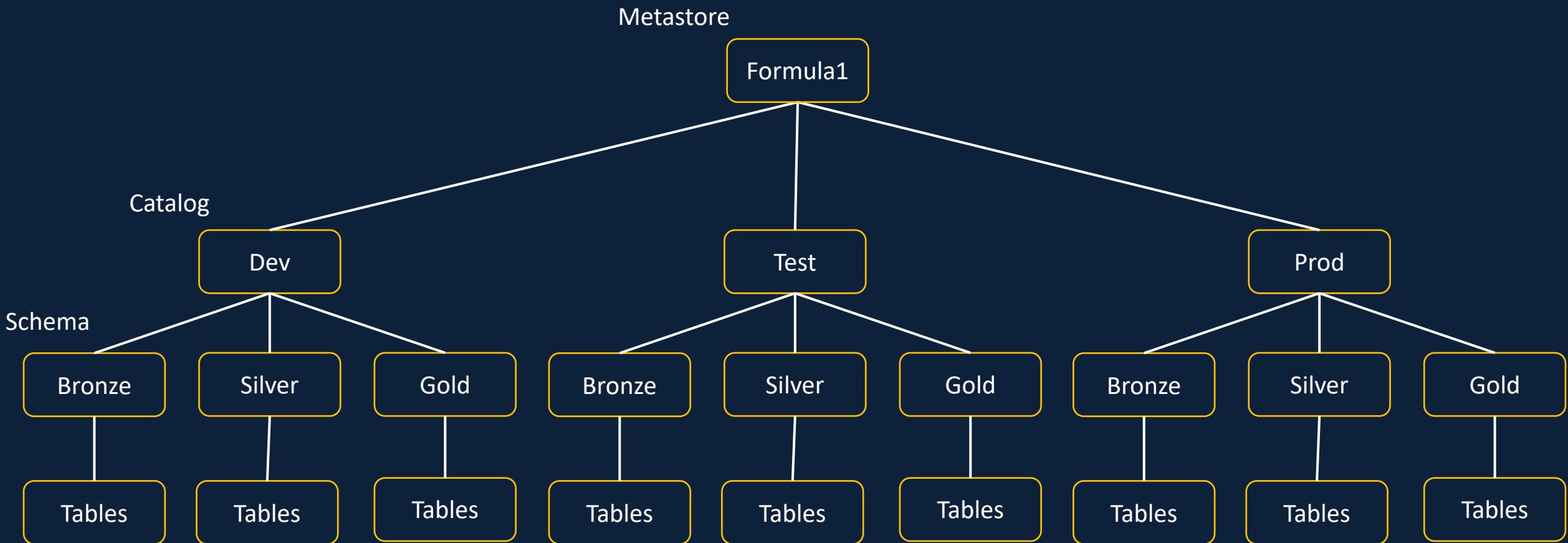
Unity Catalog Object Model



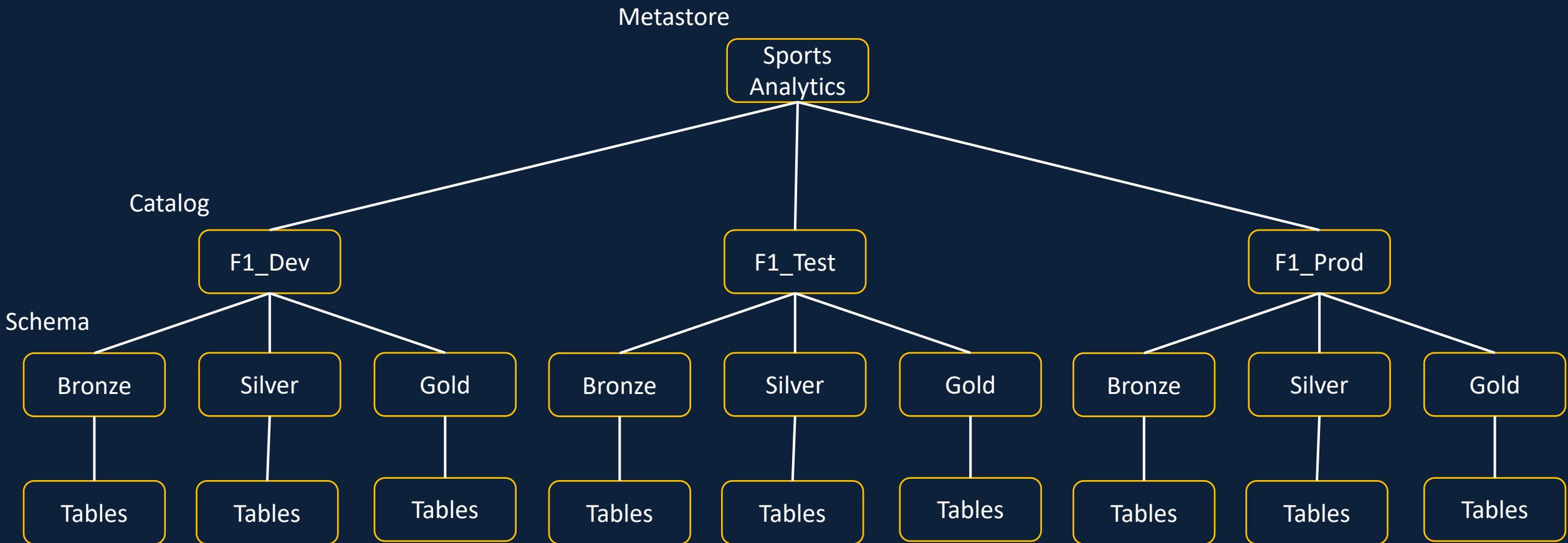
Unity Catalog Object Model



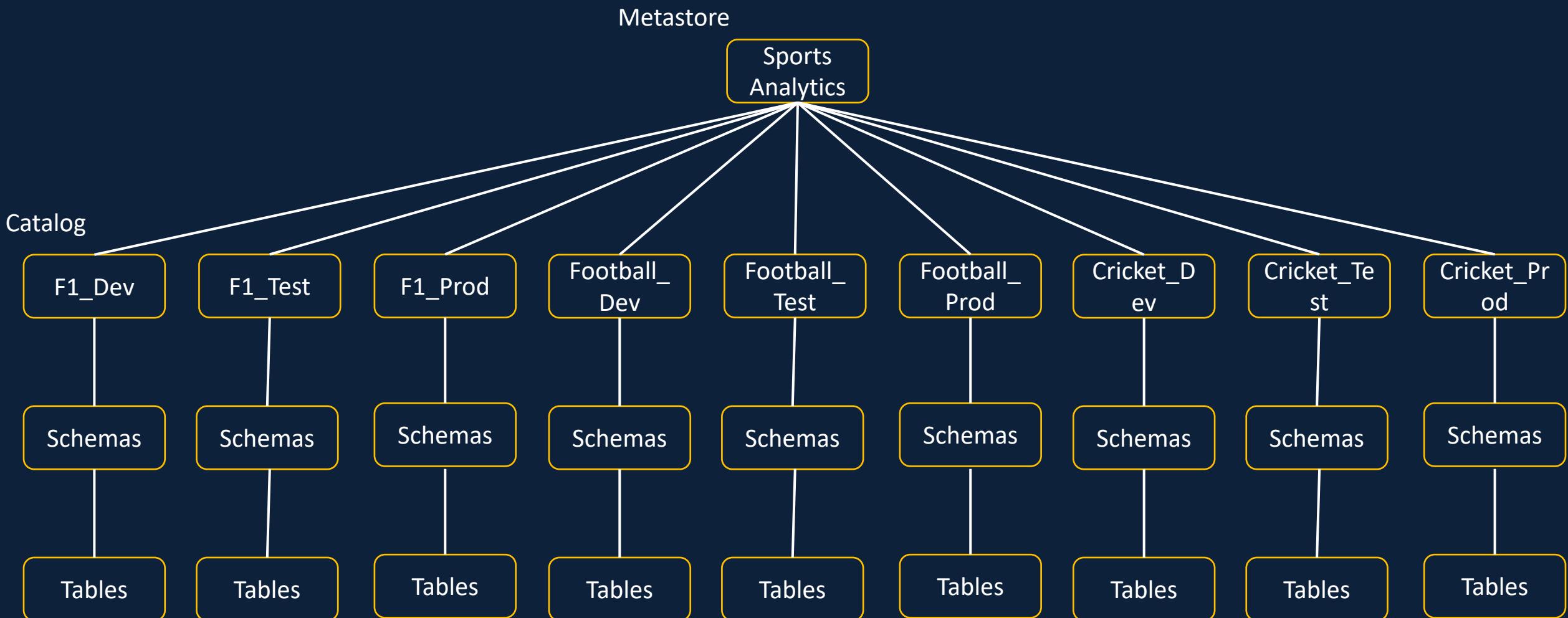
Unity Catalog Object Model



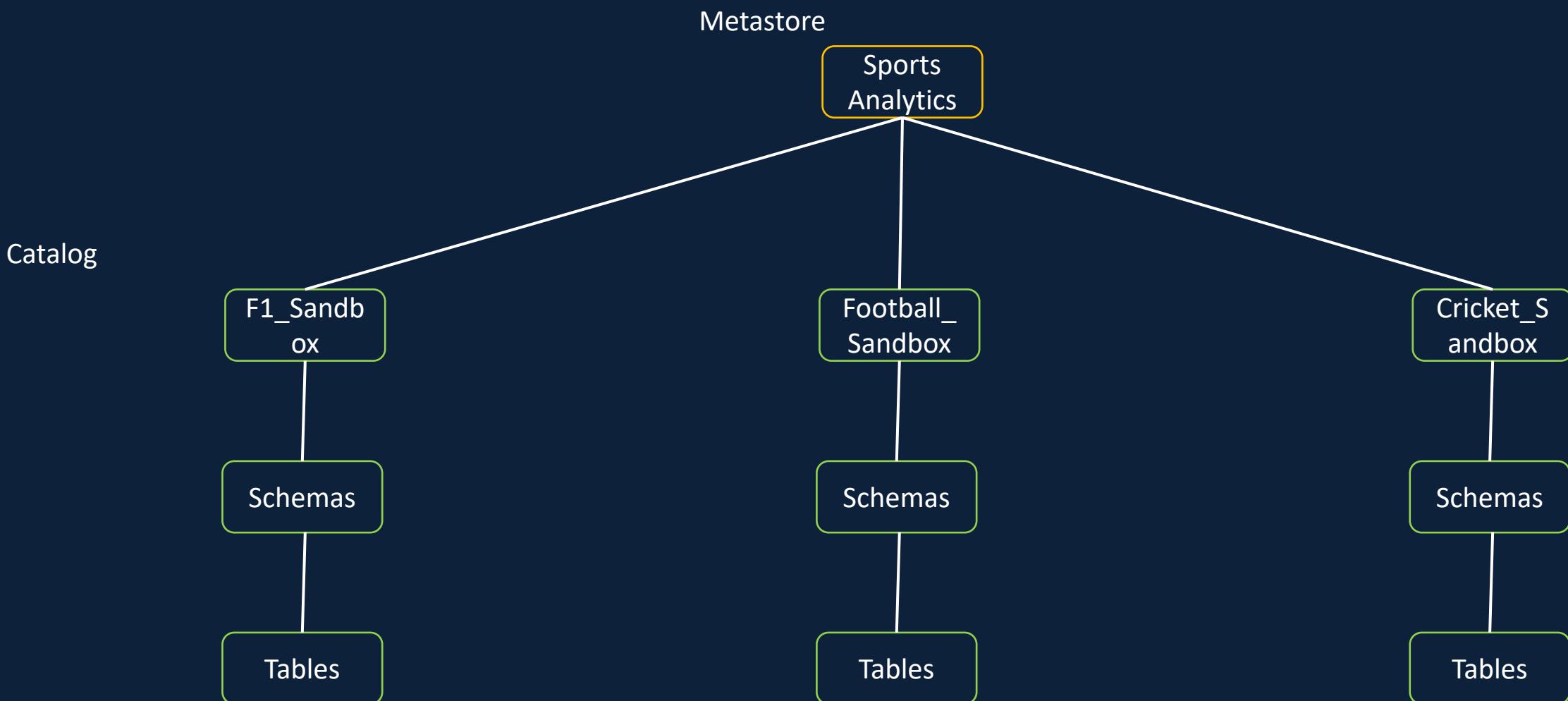
Unity Catalog Object Model



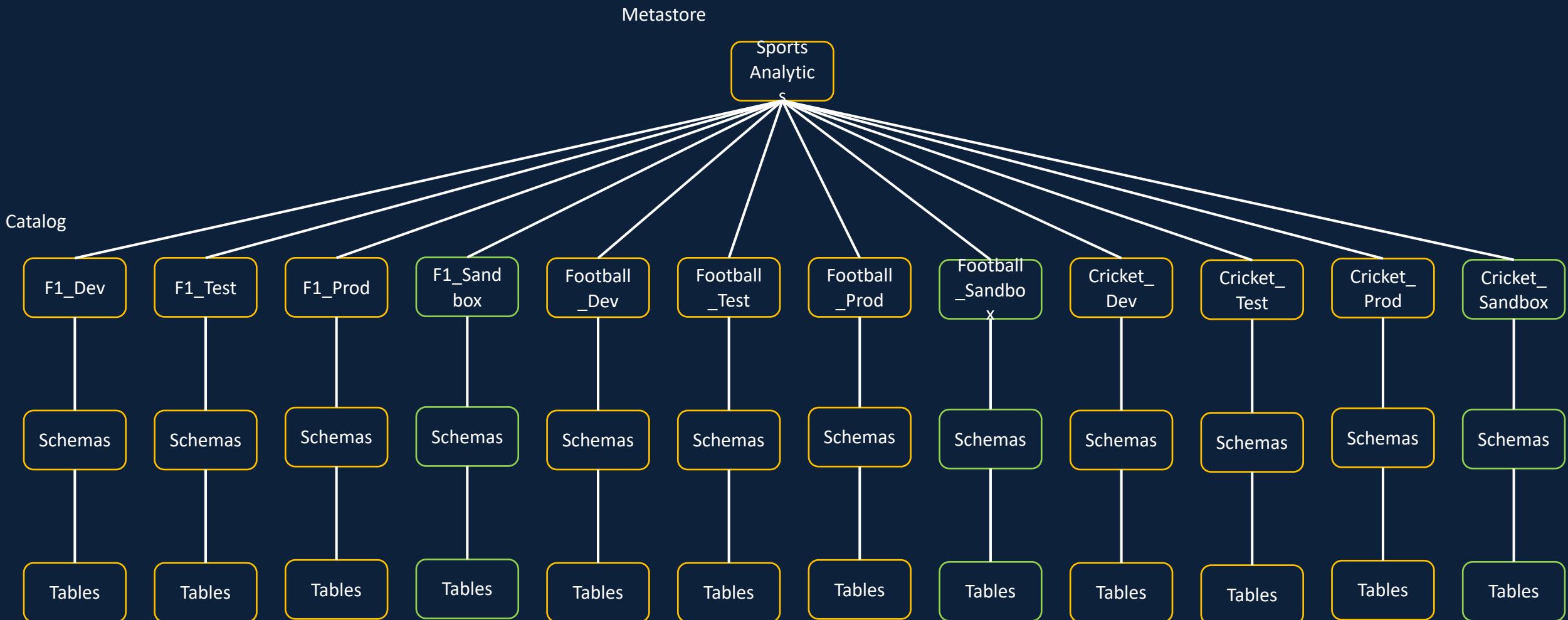
Unity Catalog Object Model



Unity Catalog Object Model



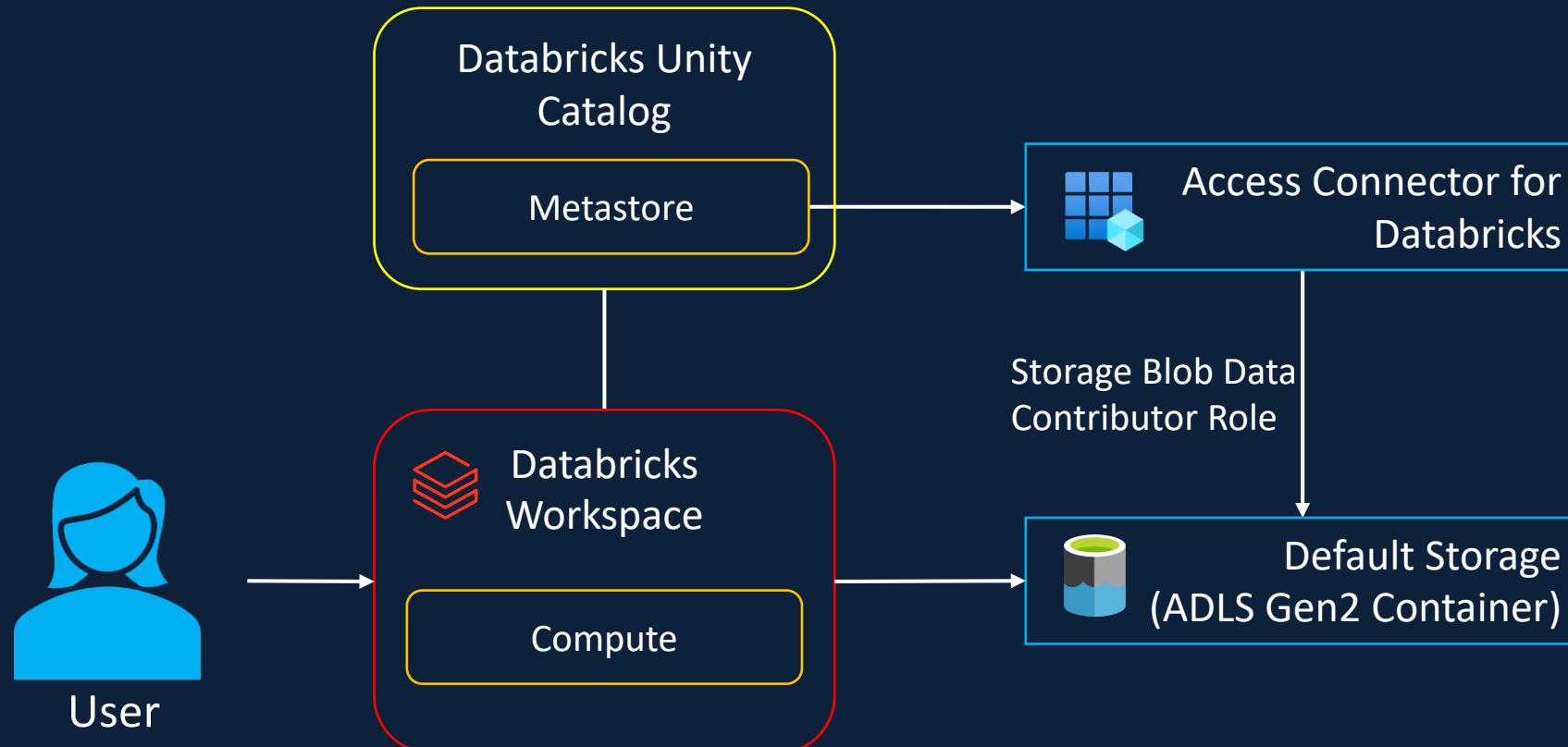
Unity Catalog Object Model



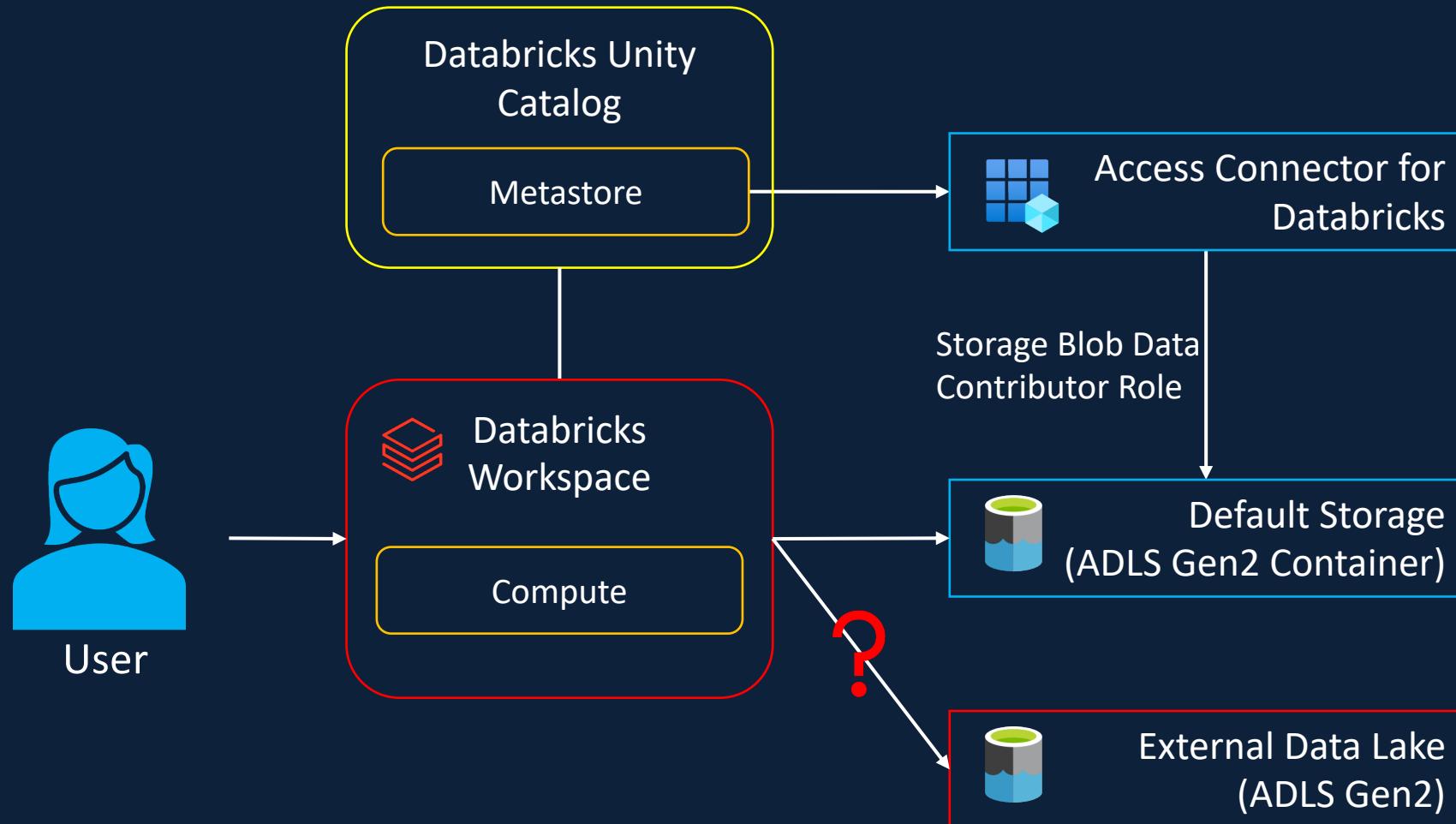
Accessing External Locations



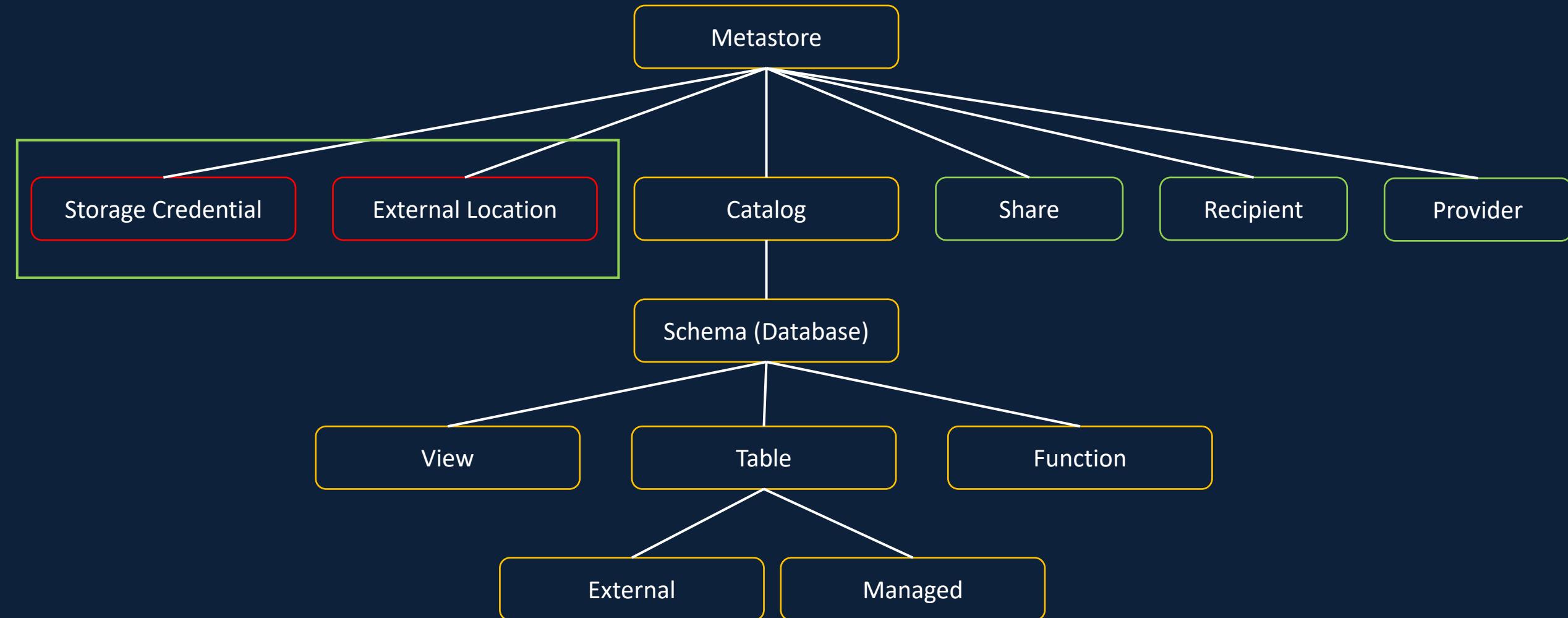
Unity Catalog Configuration



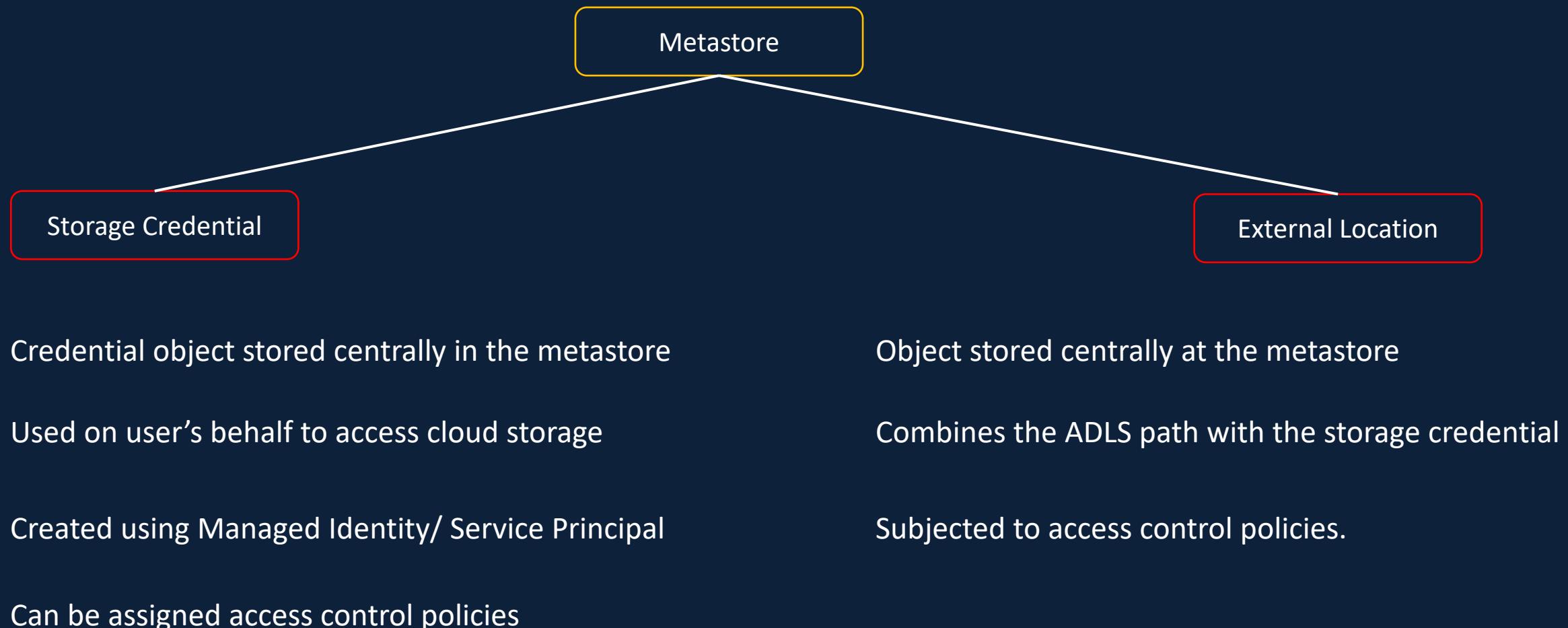
Accessing External Locations



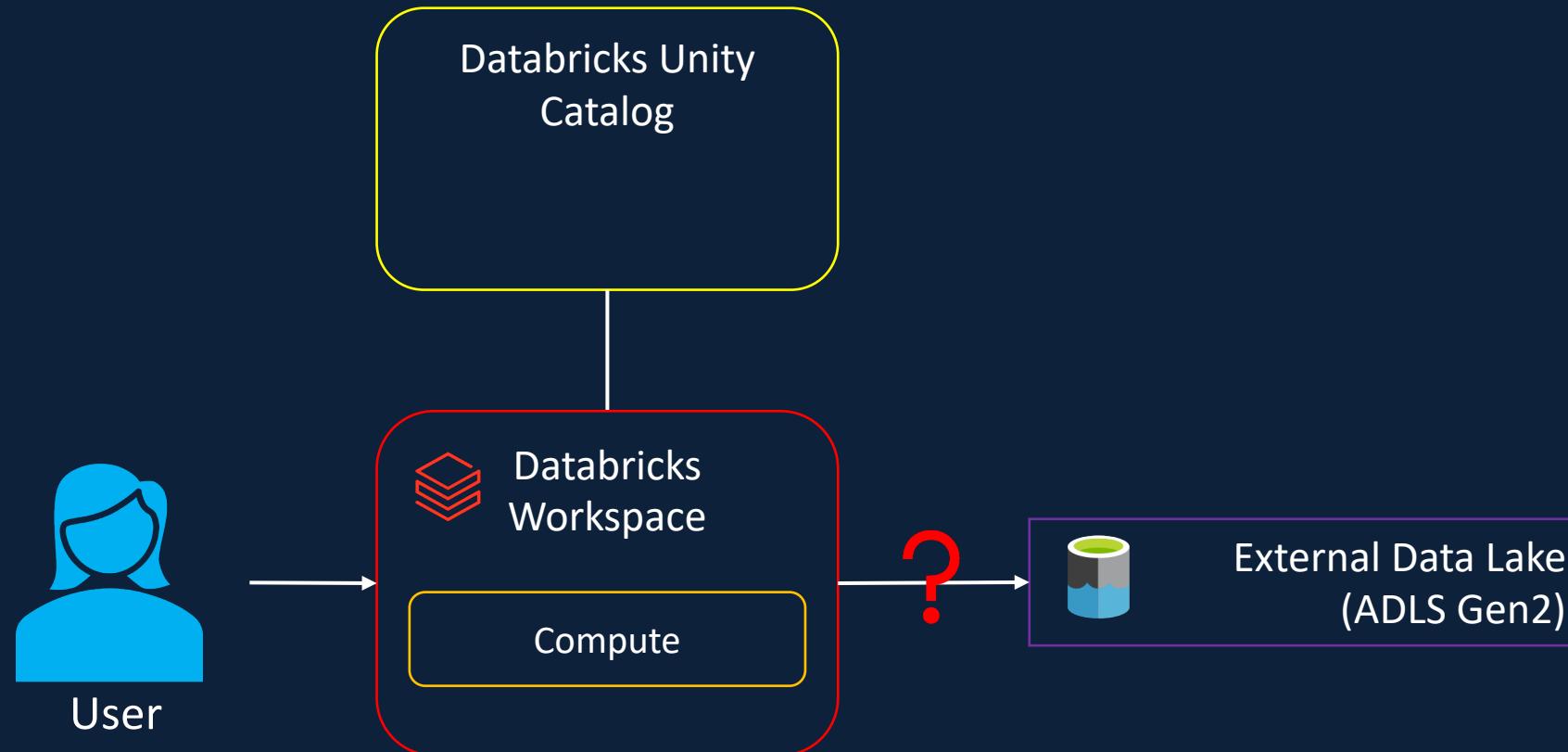
Unity Catalog Object Model



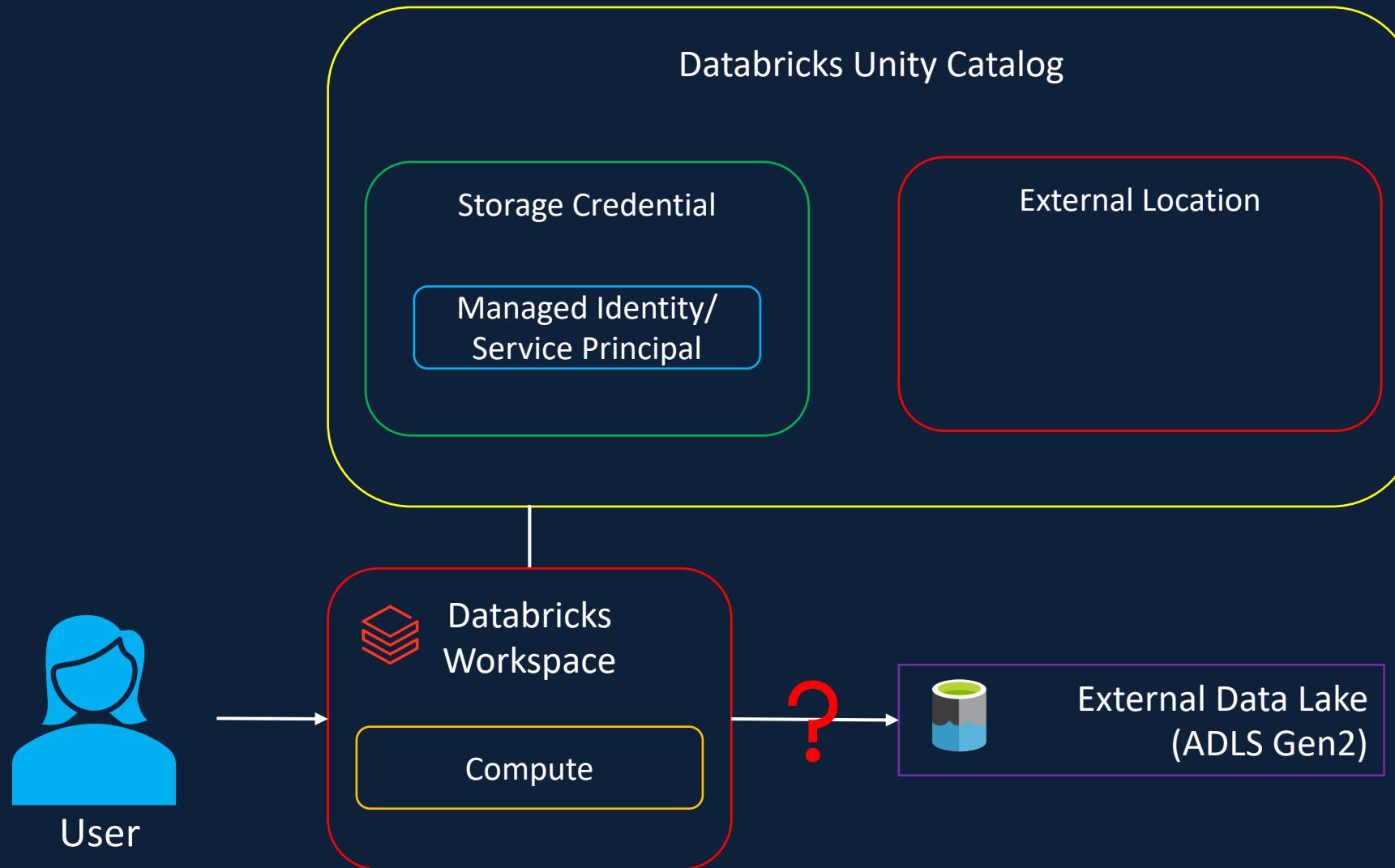
Storage Credential/ External Location



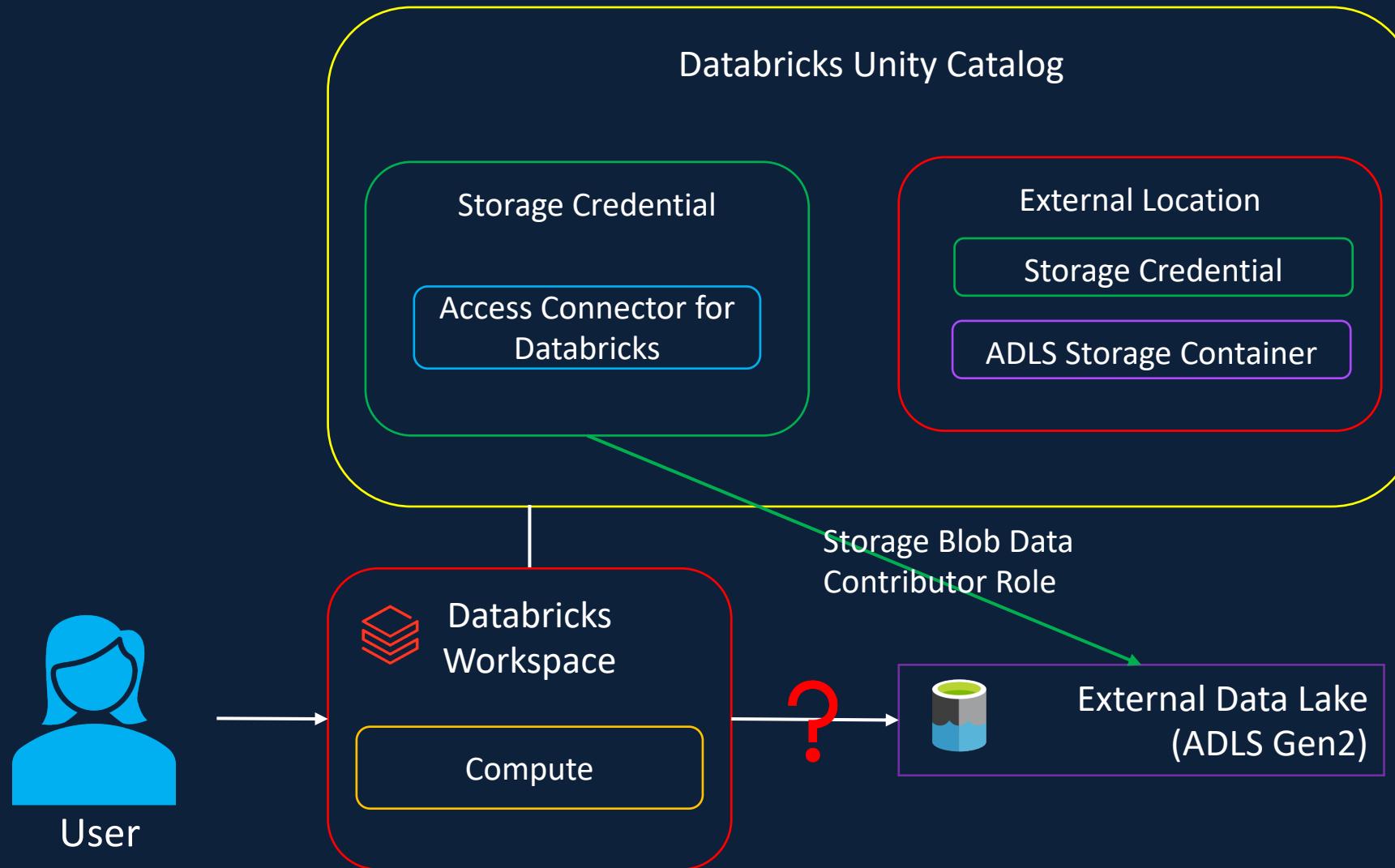
Accessing External Locations



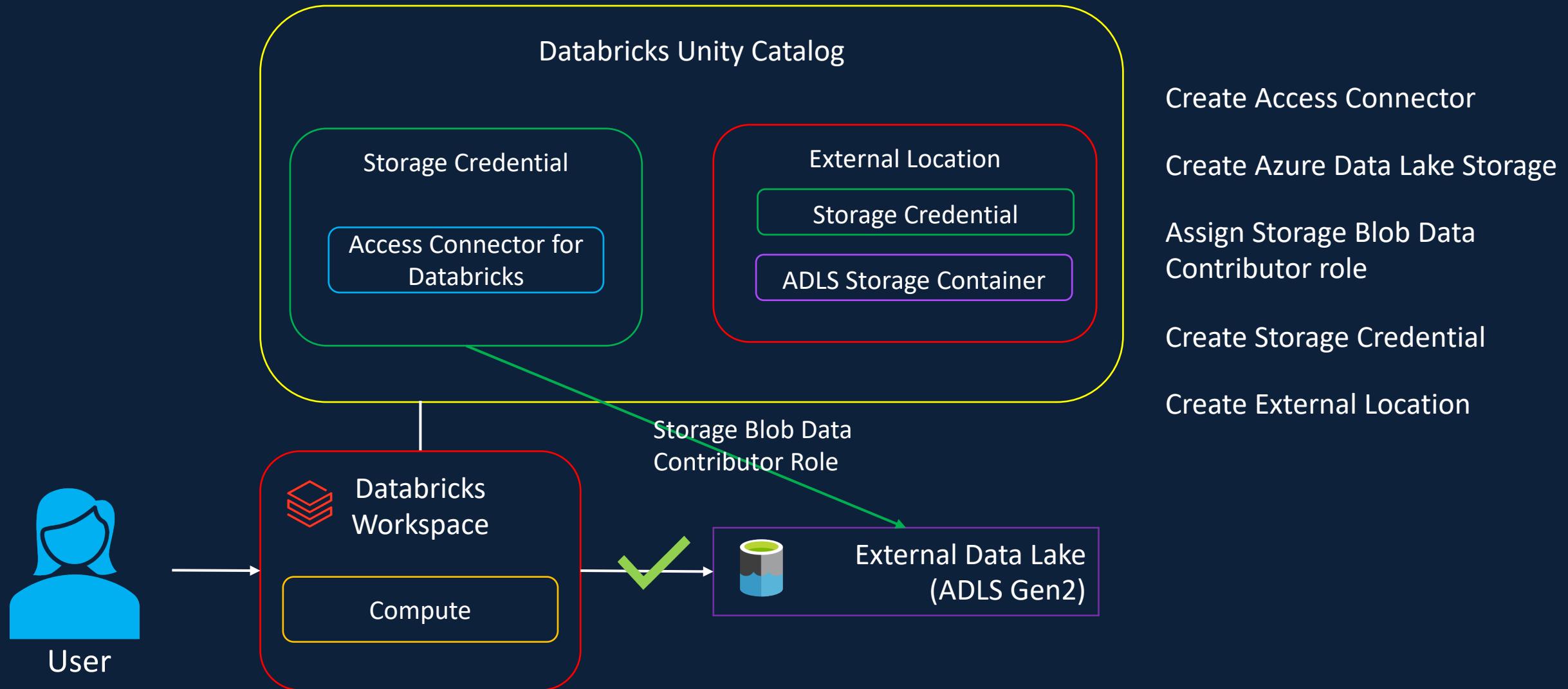
Accessing External Locations



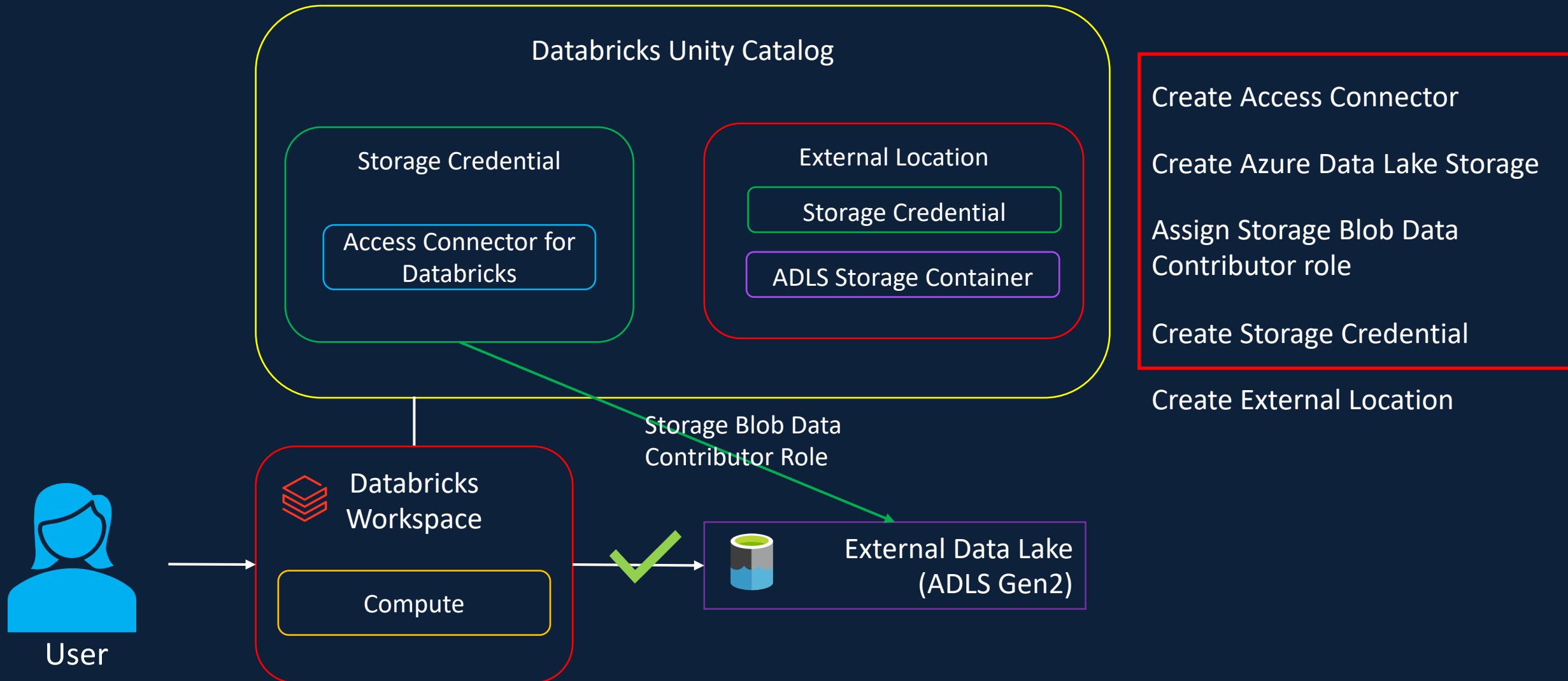
Accessing External Locations



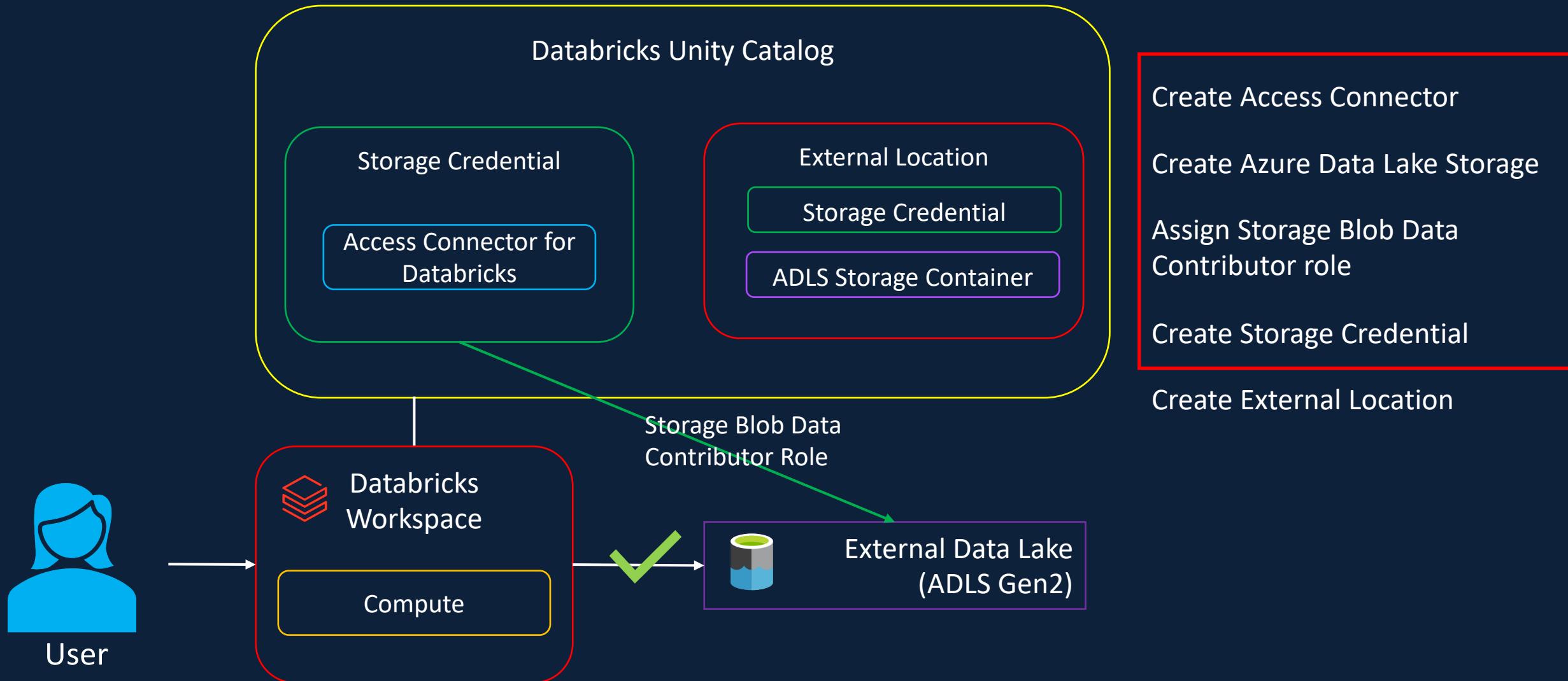
Accessing External Locations



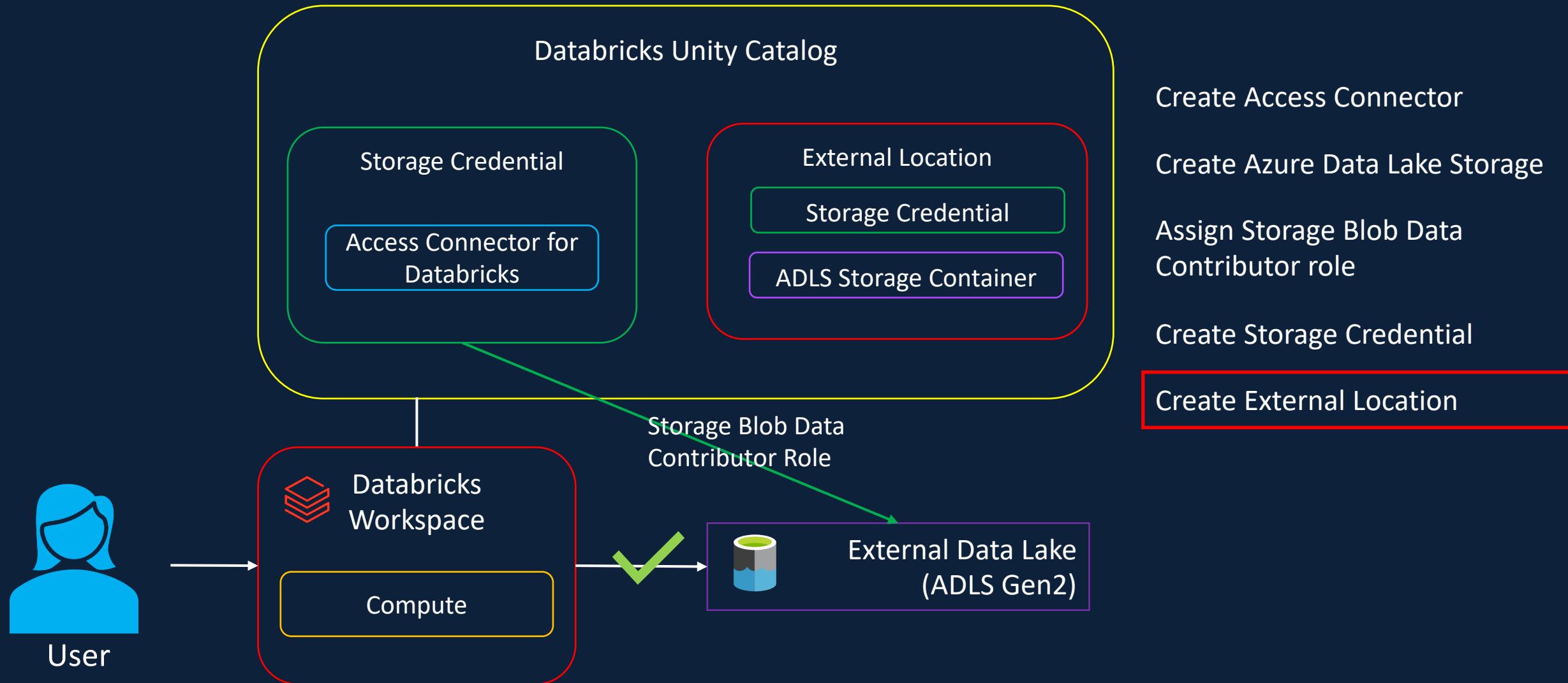
Creating Storage Credential



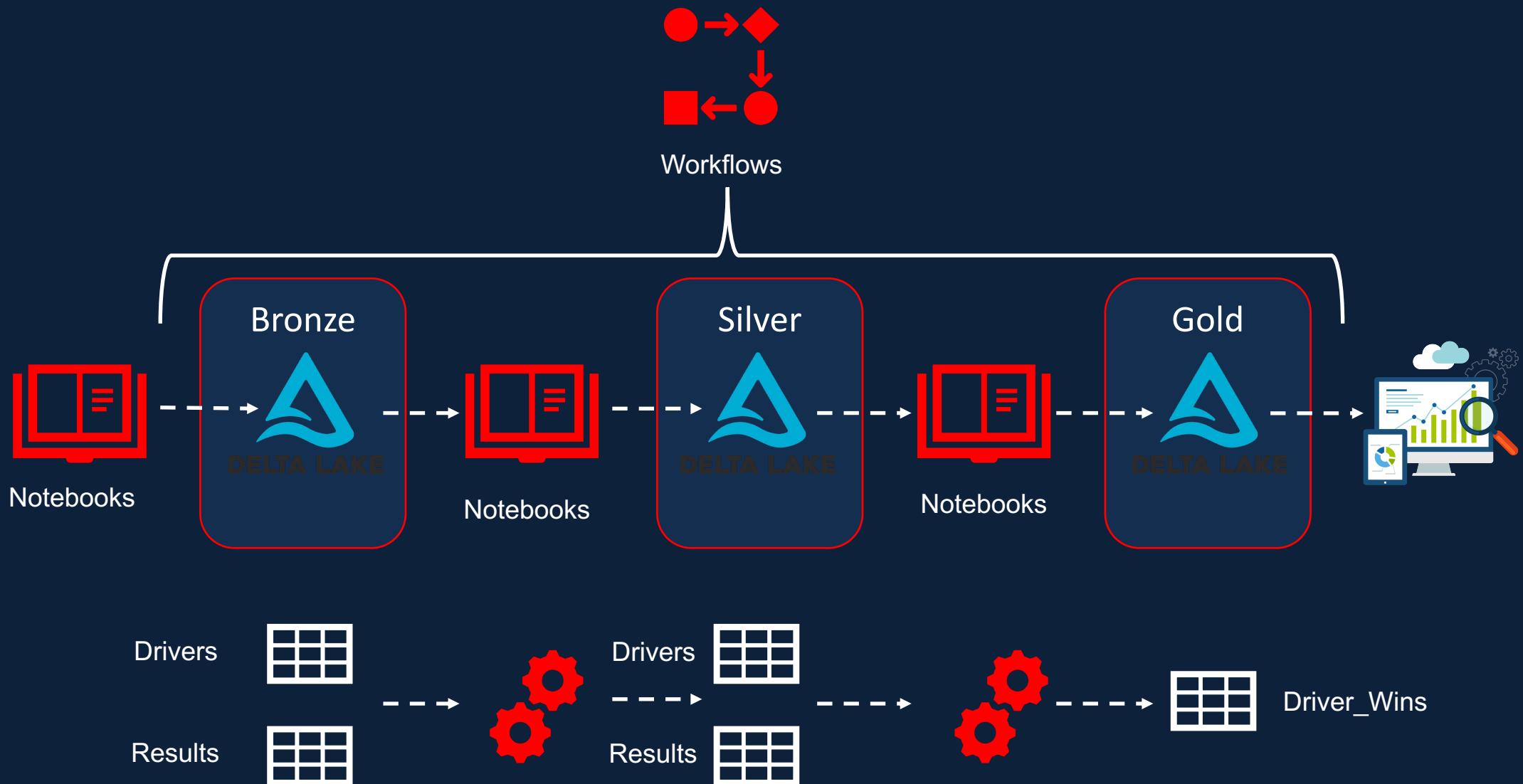
Creating Storage Credential



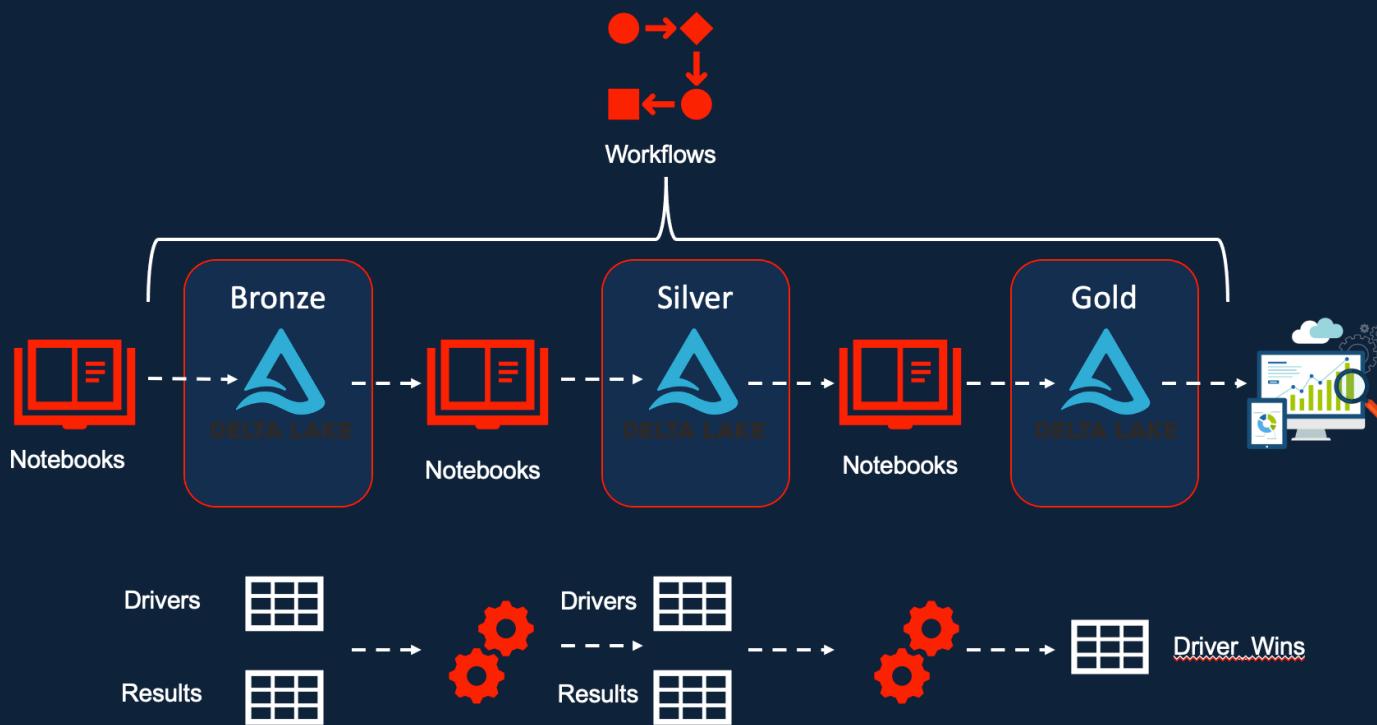
Creating External Location



Mini Project - Overview



Mini Project - Overview



Create Storage Containers

Create External Locations

Create Catalog & Schemas

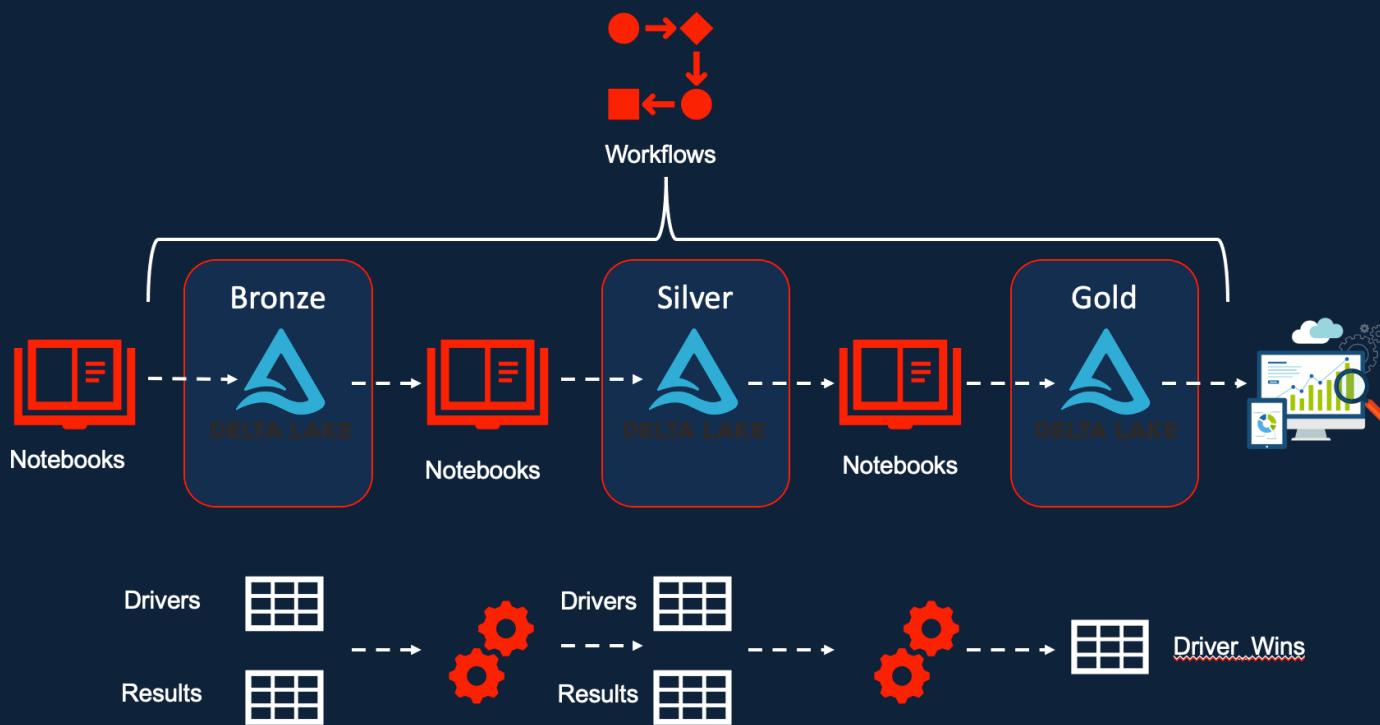
Create Bronze Tables (External)

Create Silver Tables (Managed)

Create Gold Table (Managed)

Create Databricks Workflow

Mini Project – Create External Locations



Create Storage Containers

Create External Locations

Create Catalog & Schemas

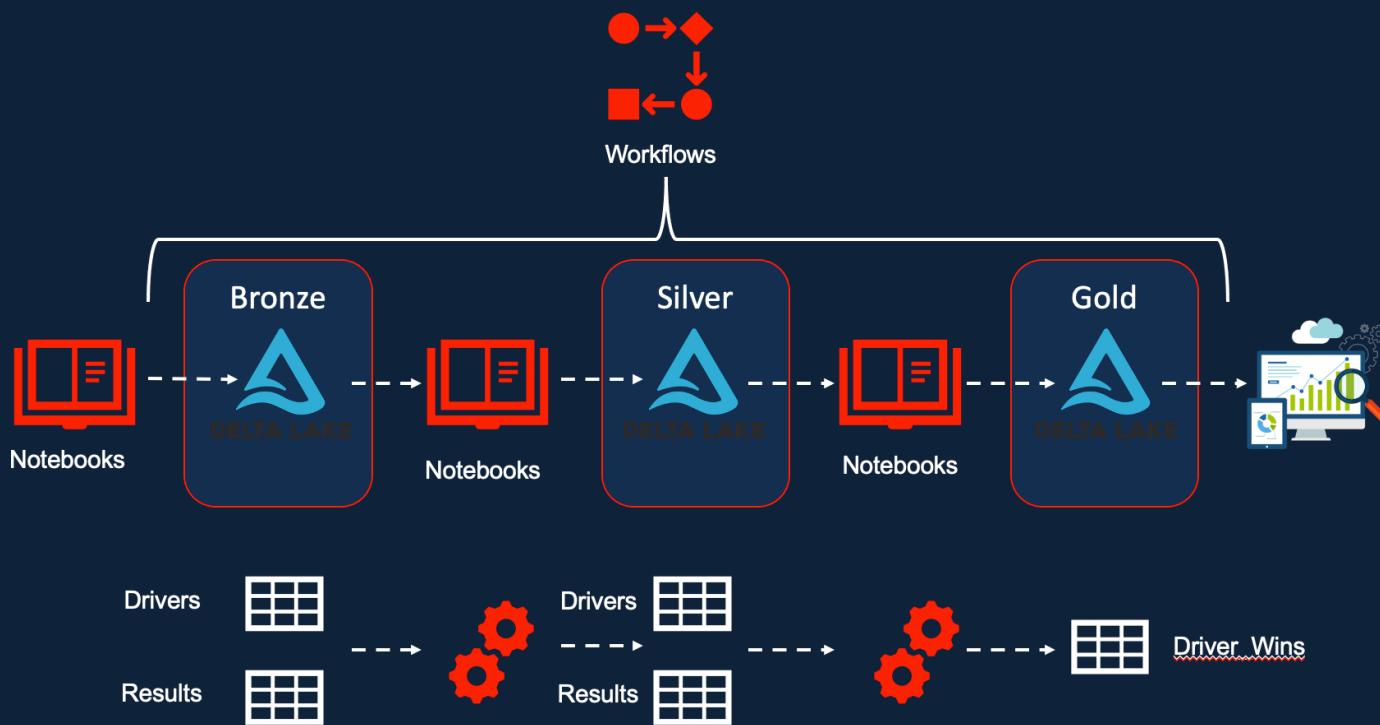
Create Bronze Tables (External)

Create Silver Tables (Managed)

Create Gold Table (Managed)

Create Databricks Workflow

Mini Project – Create Catalogs & Schemas



Create Storage Containers

Create External Locations

Create Catalog & Schemas

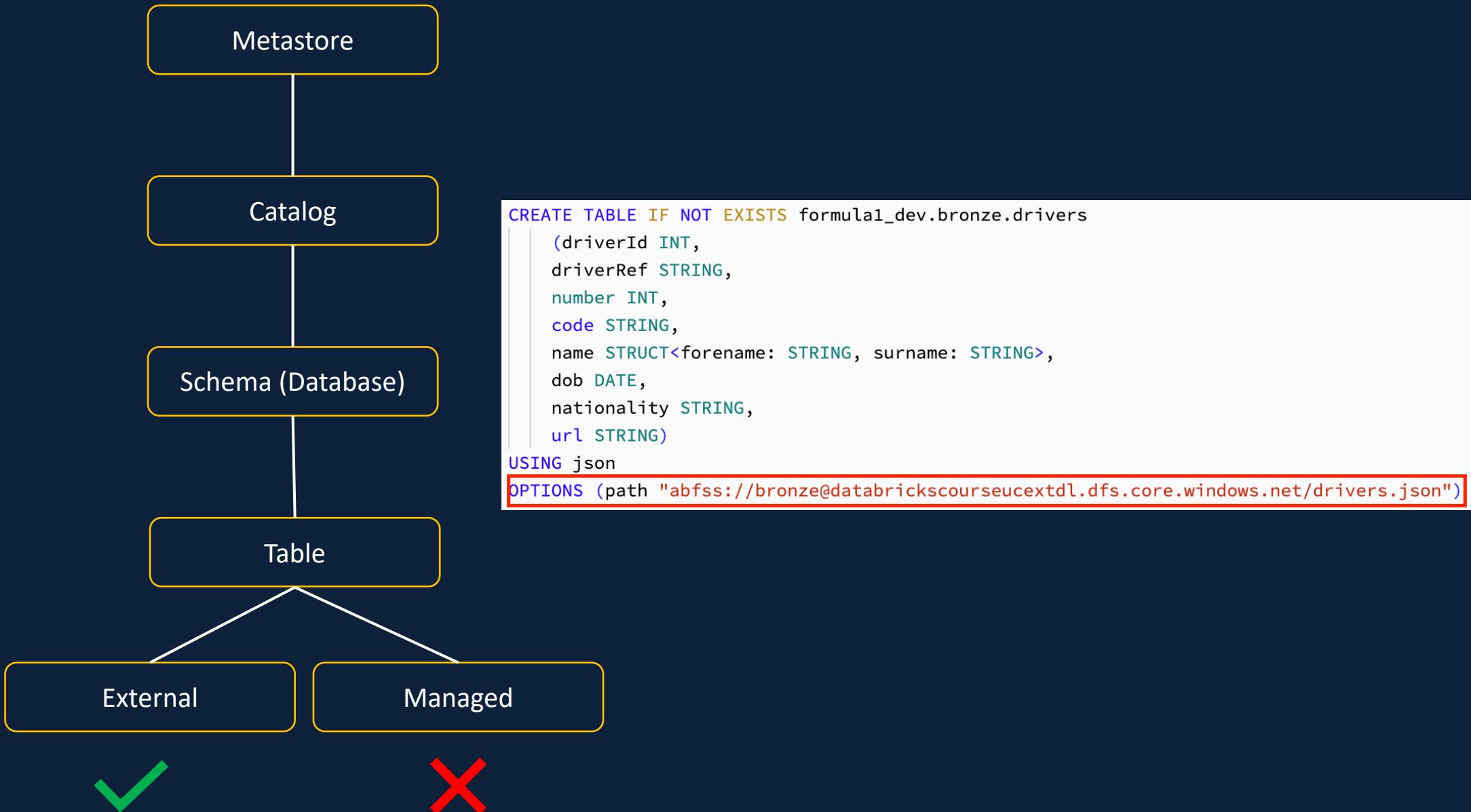
Create Bronze Tables (External)

Create Silver Tables (Managed)

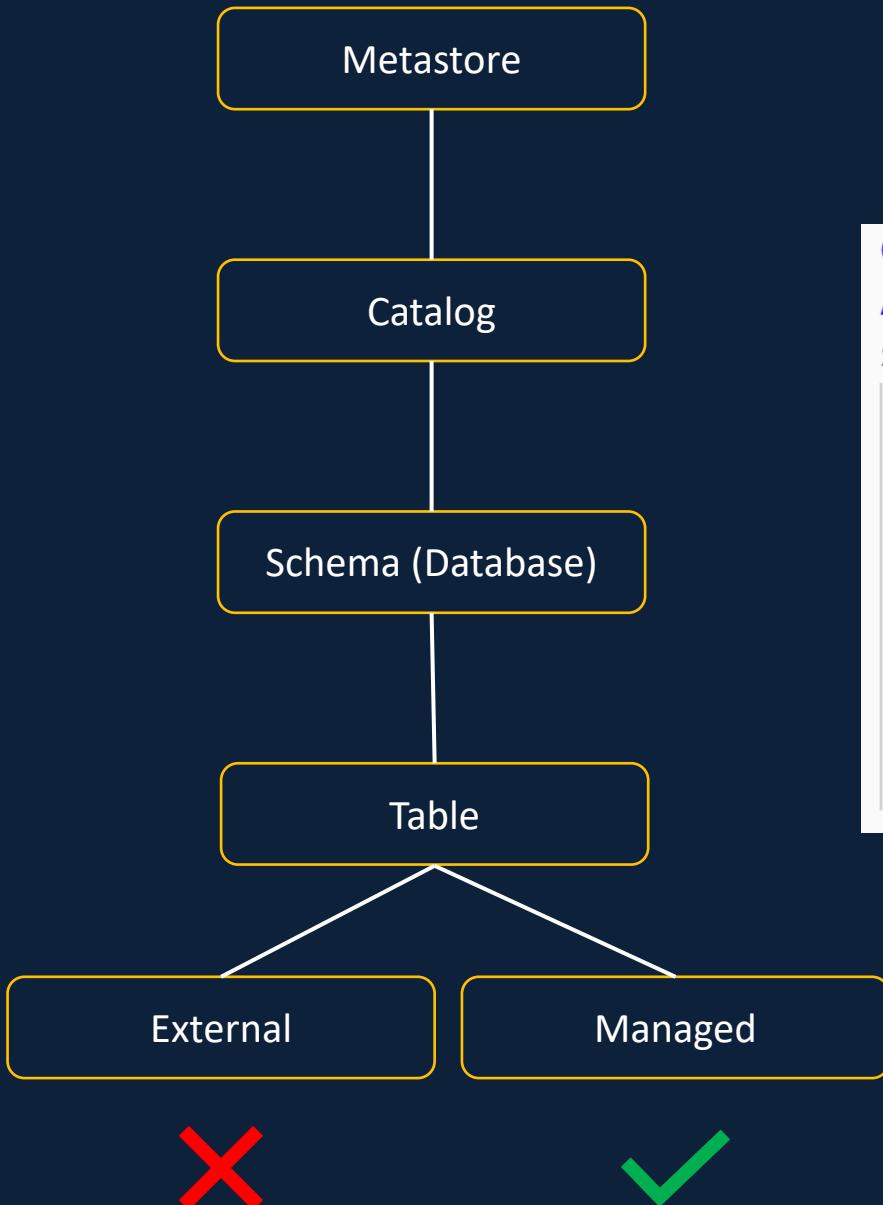
Create Gold Table (Managed)

Create Databricks Workflow

Mini Project – Create Catalogs & Schemas

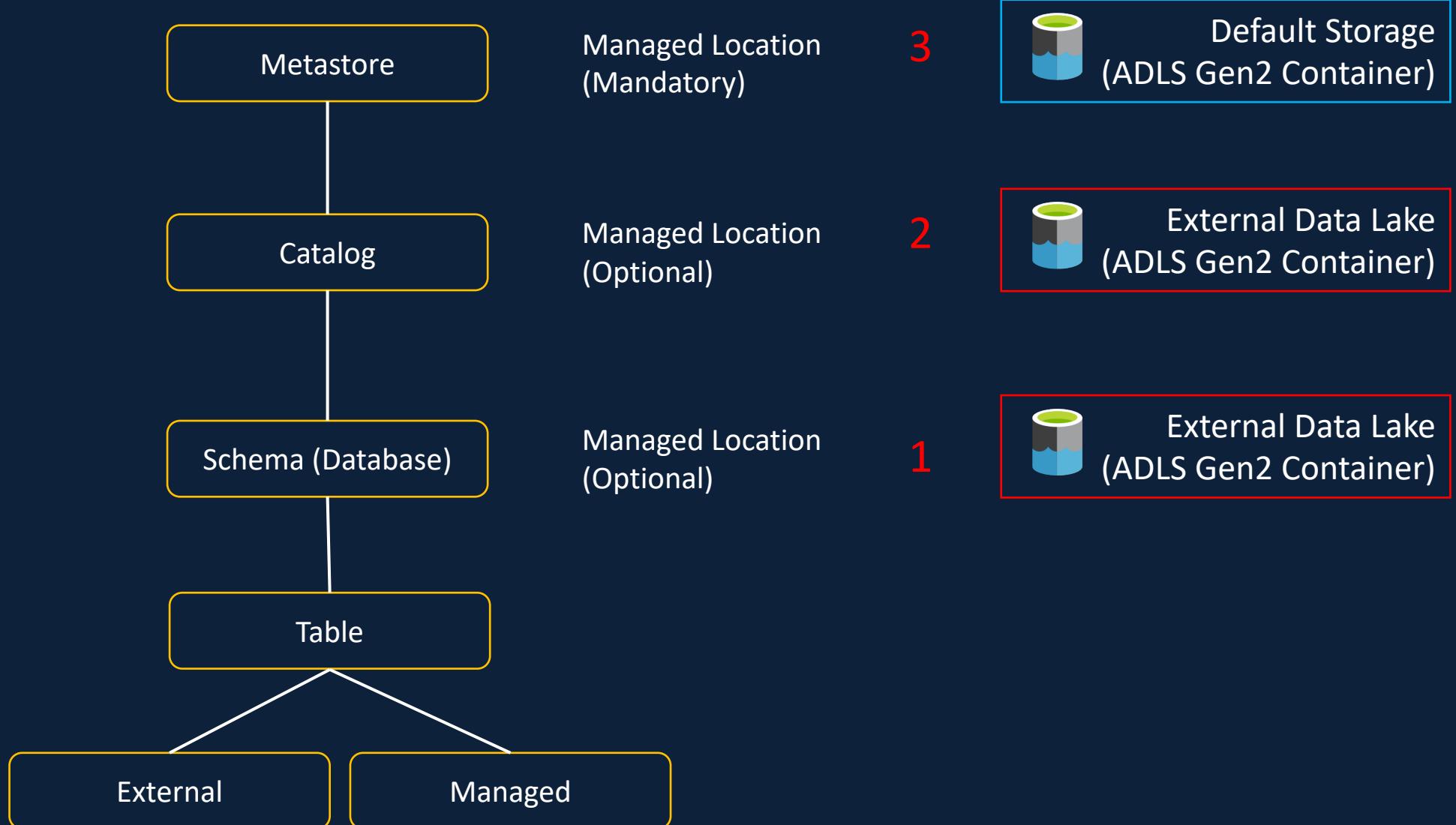


Mini Project – Create Catalogs & Schemas

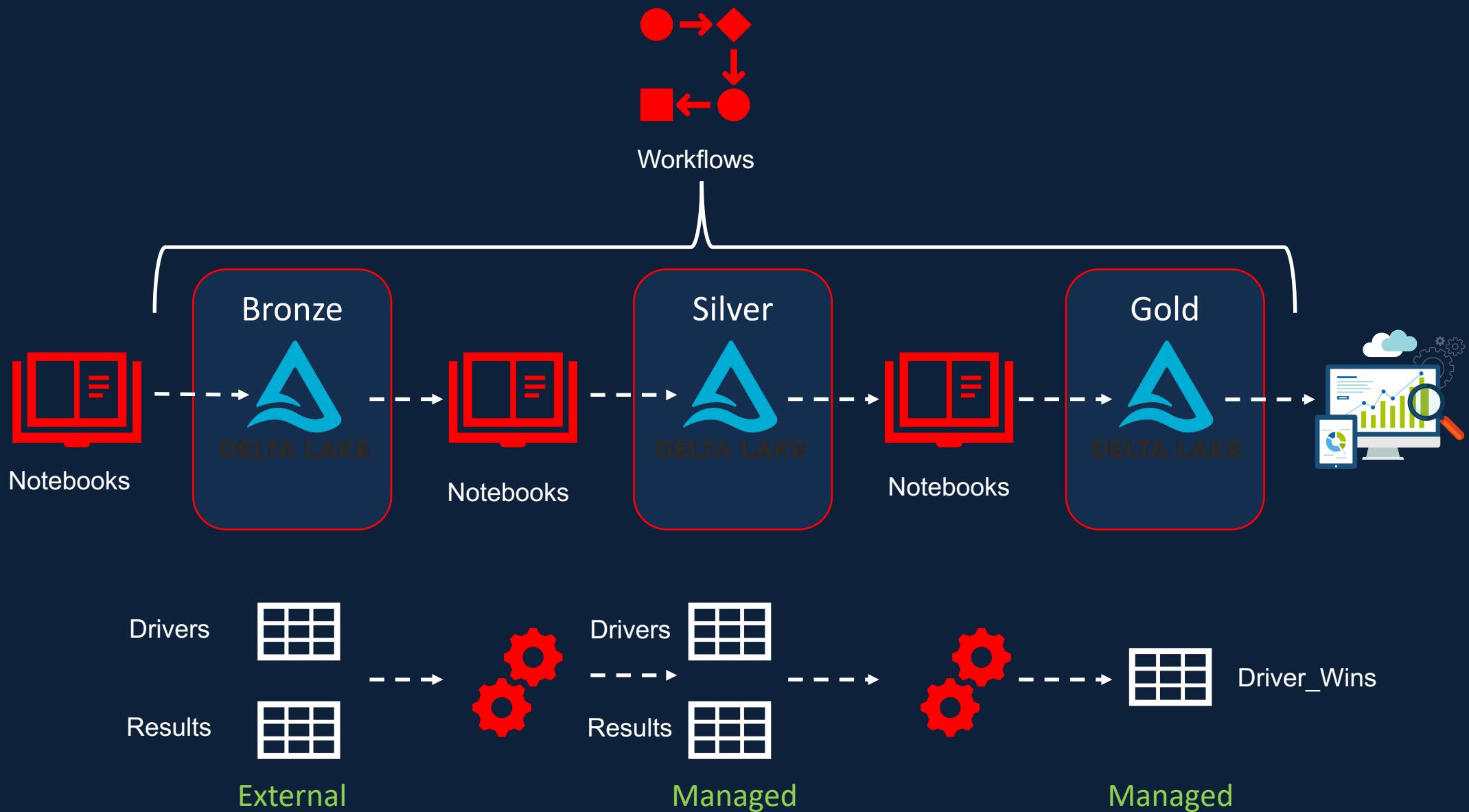


```
CREATE TABLE formula1_dev.silver.drivers  
AS  
SELECT driverId AS driver_id,  
driverRef AS driver_ref,  
number,  
code,  
concat(name.forename, ' ', name.surname) AS name,  
dob,  
nationality,  
current_timestamp() AS ingestion_date  
FROM formula1_dev.bronze.drivers;
```

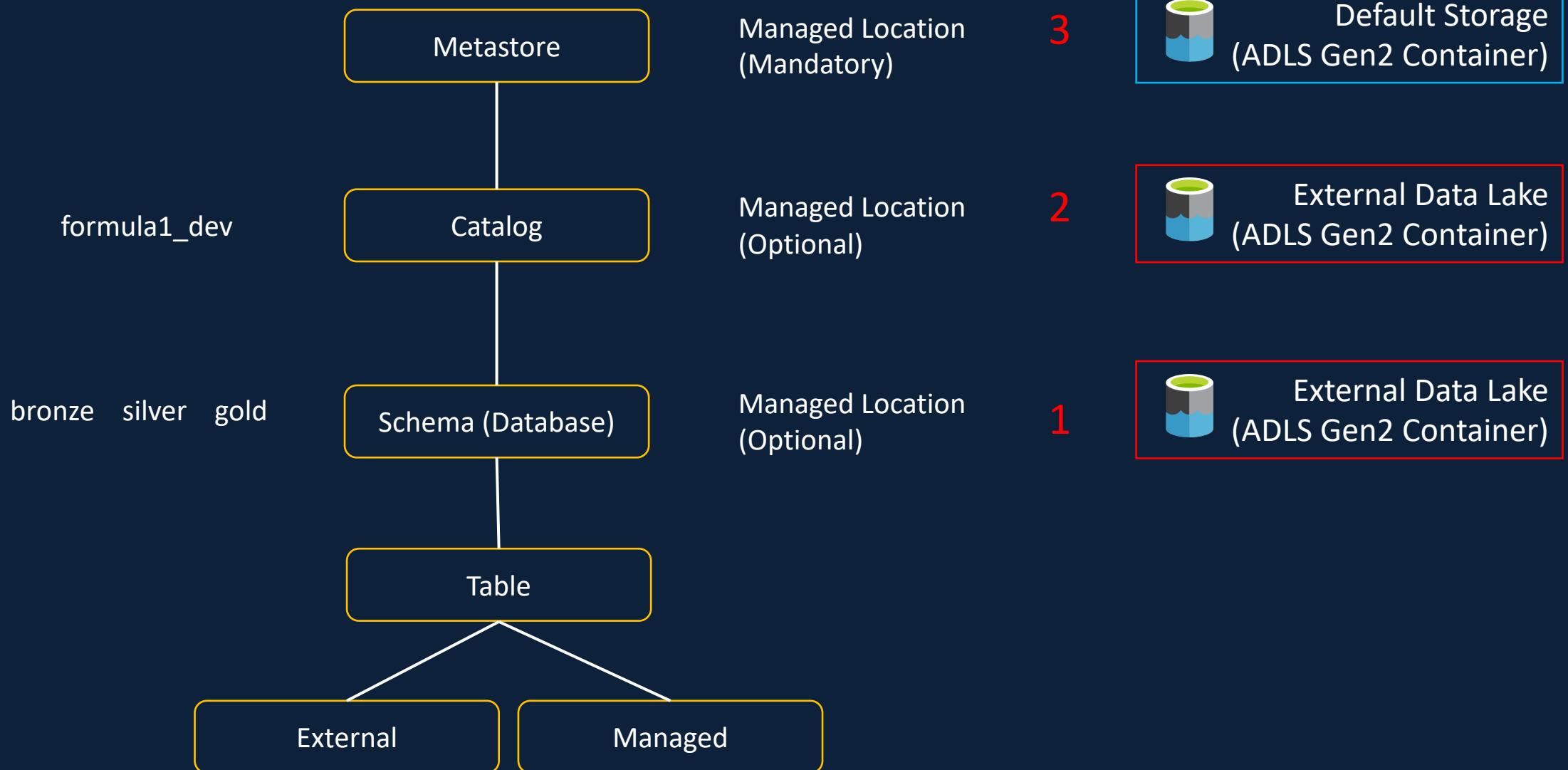
Mini Project – Create Catalogs & Schemas



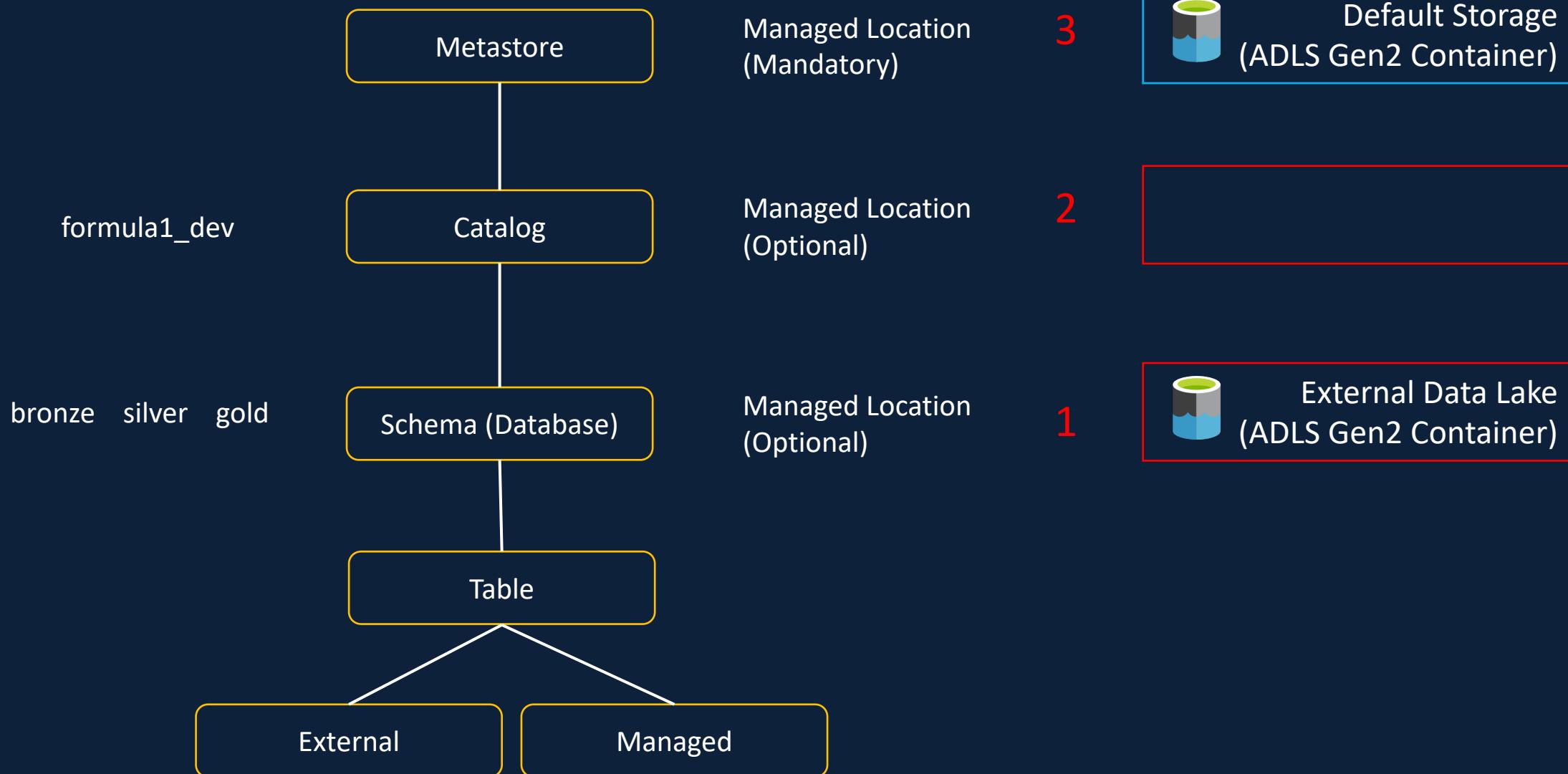
Mini Project – Create Catalogs & Schemas



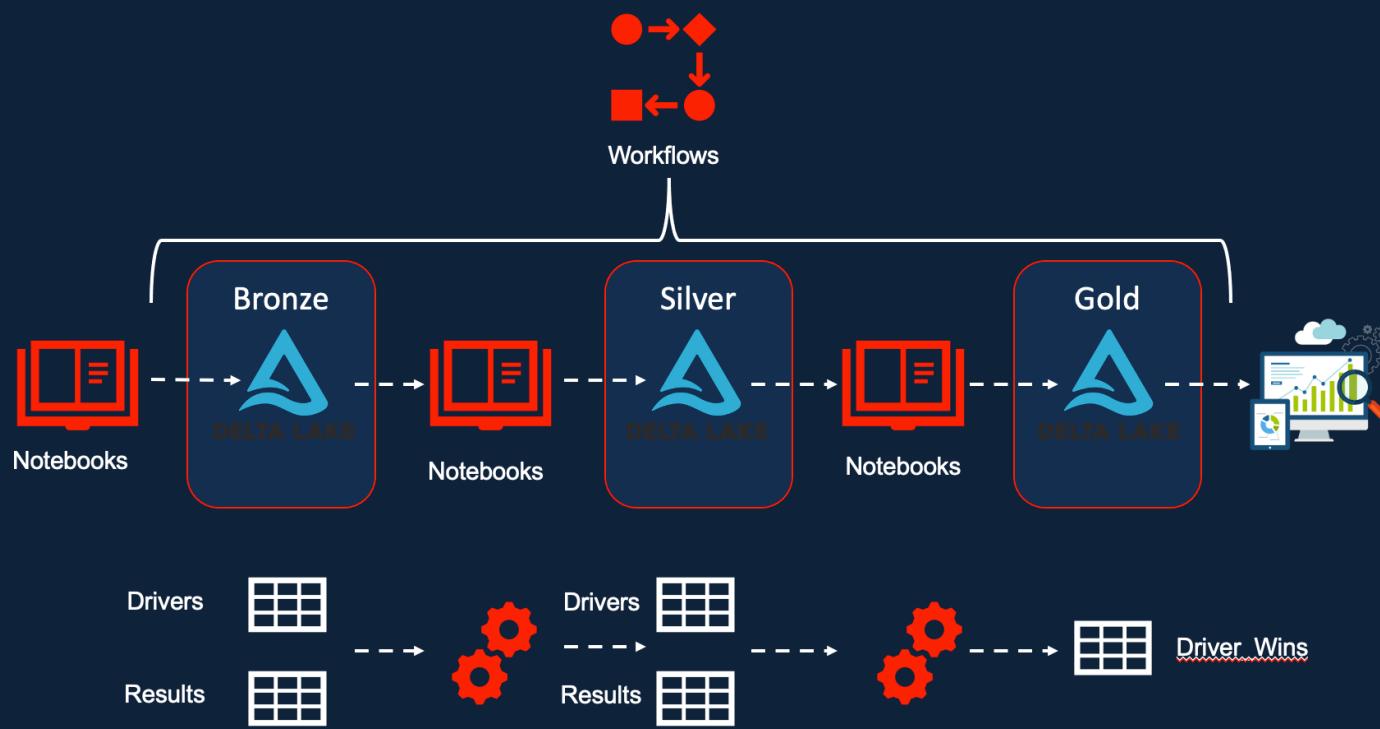
Mini Project – Create Catalogs & Schemas



Mini Project – Create Catalogs & Schemas



Mini Project – Create External Tables



Create Storage Containers

Create External Locations

Create Catalog & Schemas

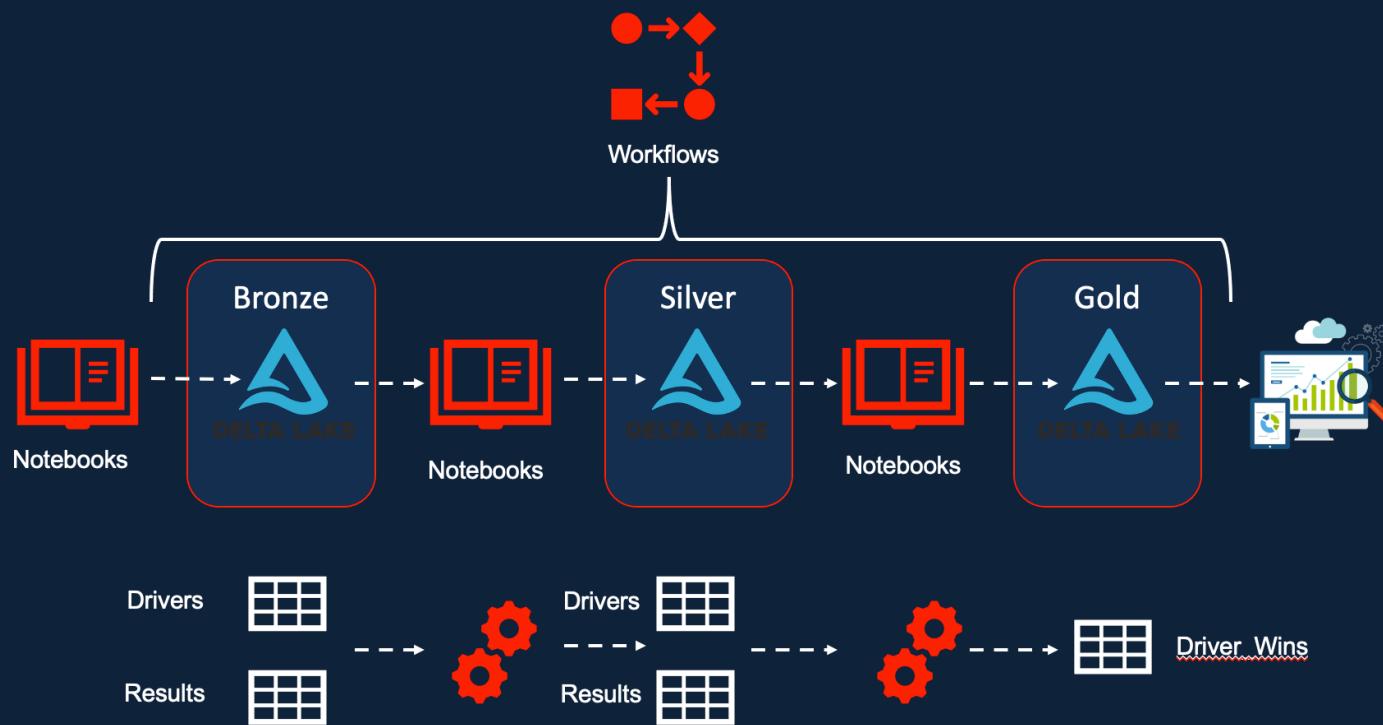
Create Bronze Tables (External)

Create Silver Tables (Managed)

Create Gold Table (Managed)

Create Databricks Workflow

Mini Project – Create Managed Tables



Create Storage Containers

Create External Locations

Create Catalog & Schemas

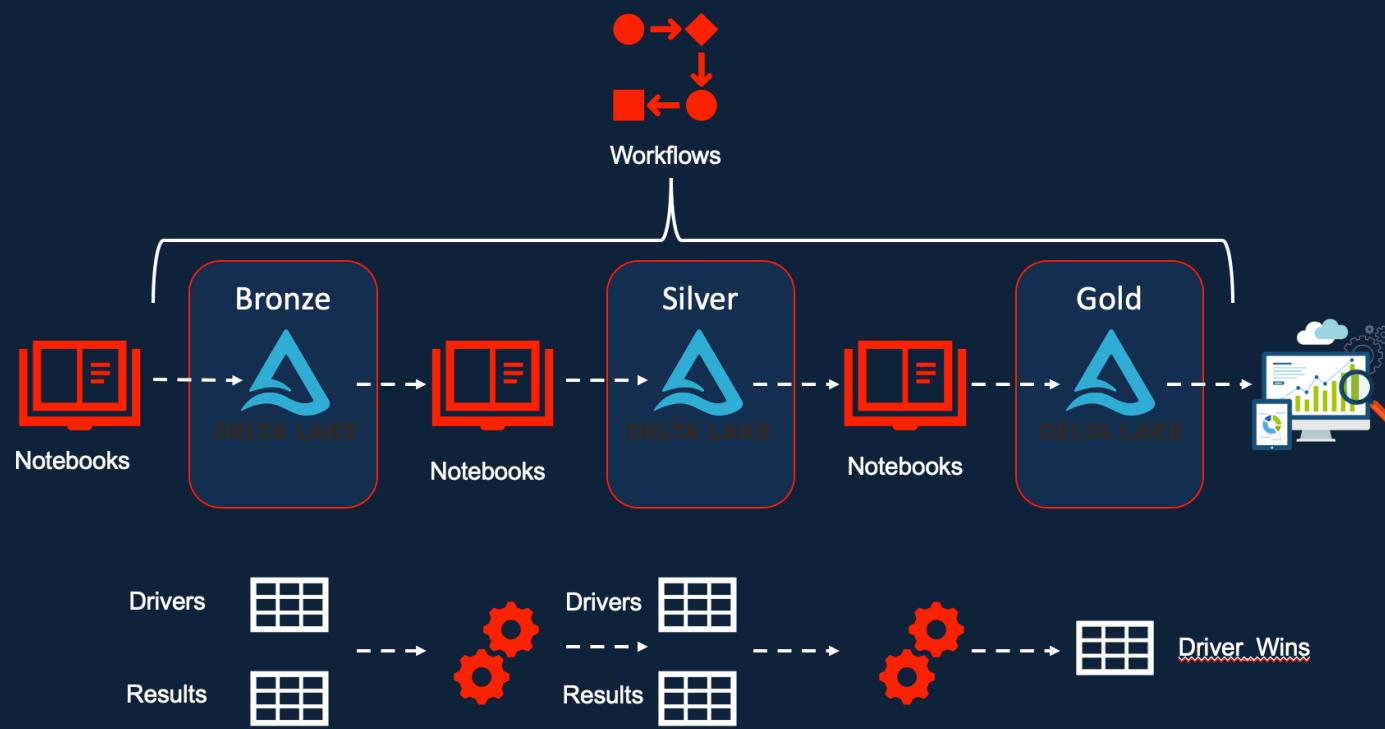
Create Bronze Tables (External)

Create Silver Tables (Managed)

Create Gold Table (Managed)

Create Databricks Workflow

Mini Project – Create Workflow



Create Storage Containers

Create External Locations

Create Catalog & Schemas

Create Bronze Tables (External)

Create Silver Tables (Managed)

Create Gold Table (Managed)

Create Databricks Workflow

Unity Catalog – Key capabilities



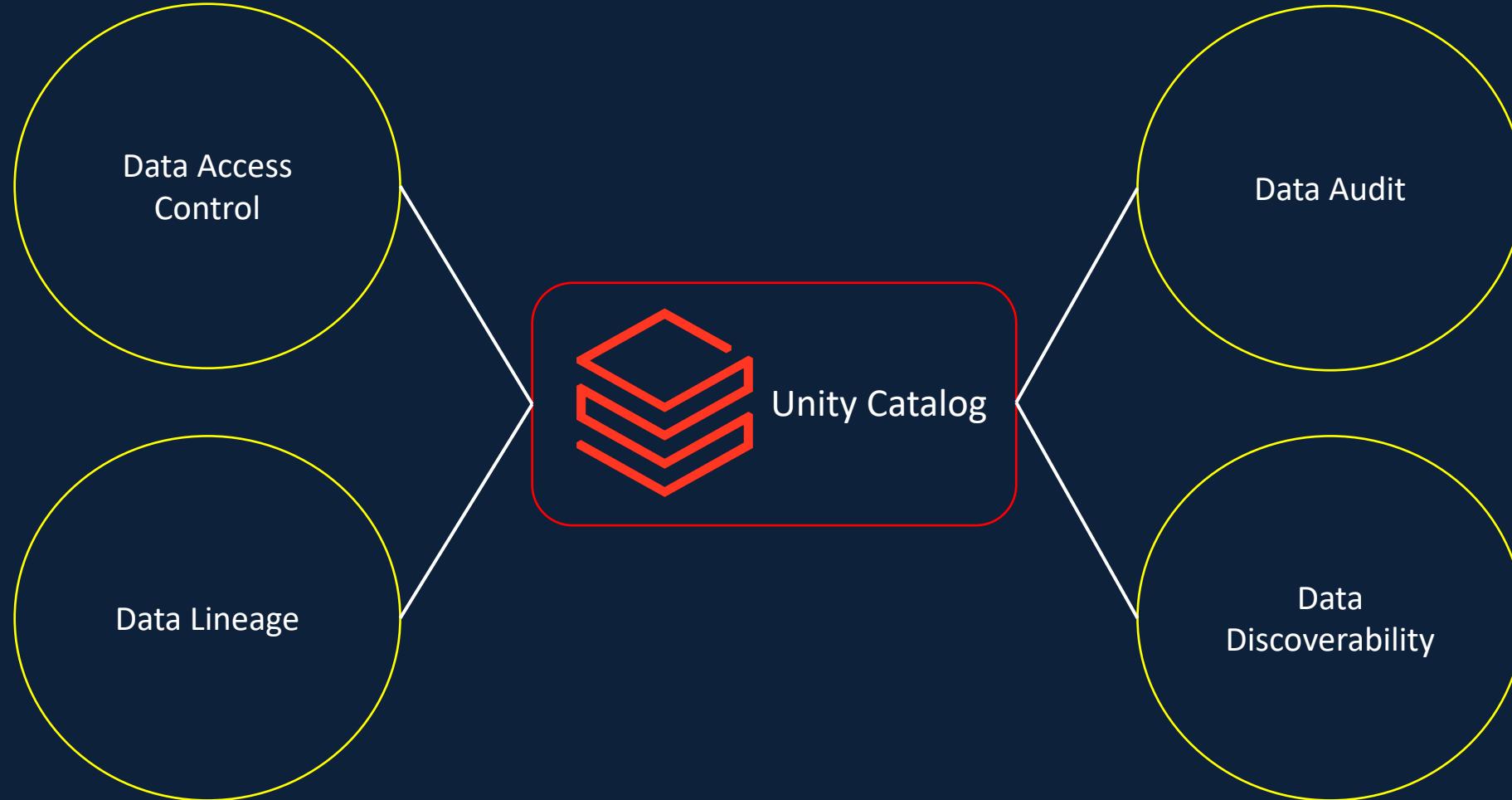
Data Discovery

Data Audit

Data Lineage

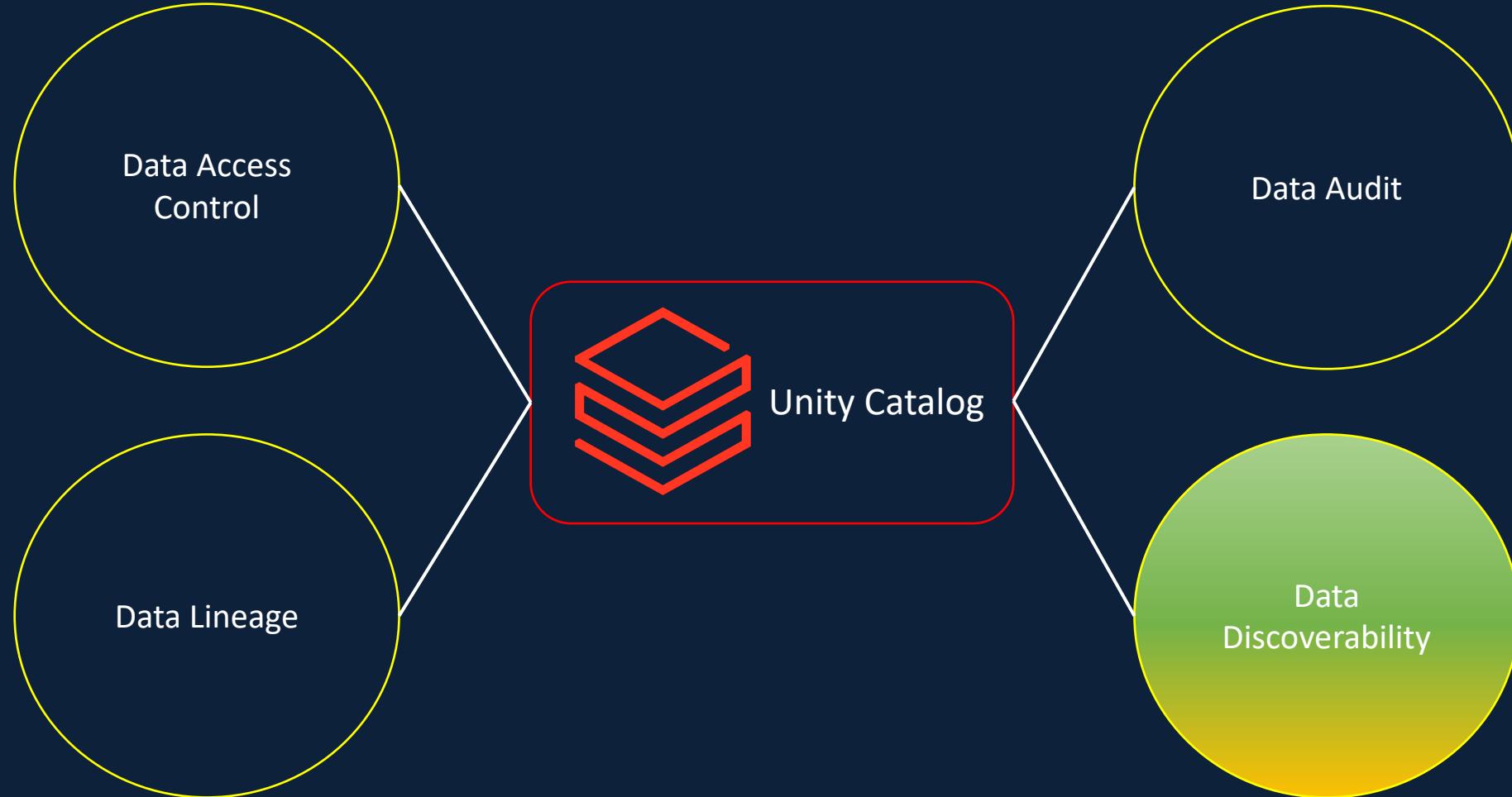
Data Access Control

Unity Catalog – Data Discoverability



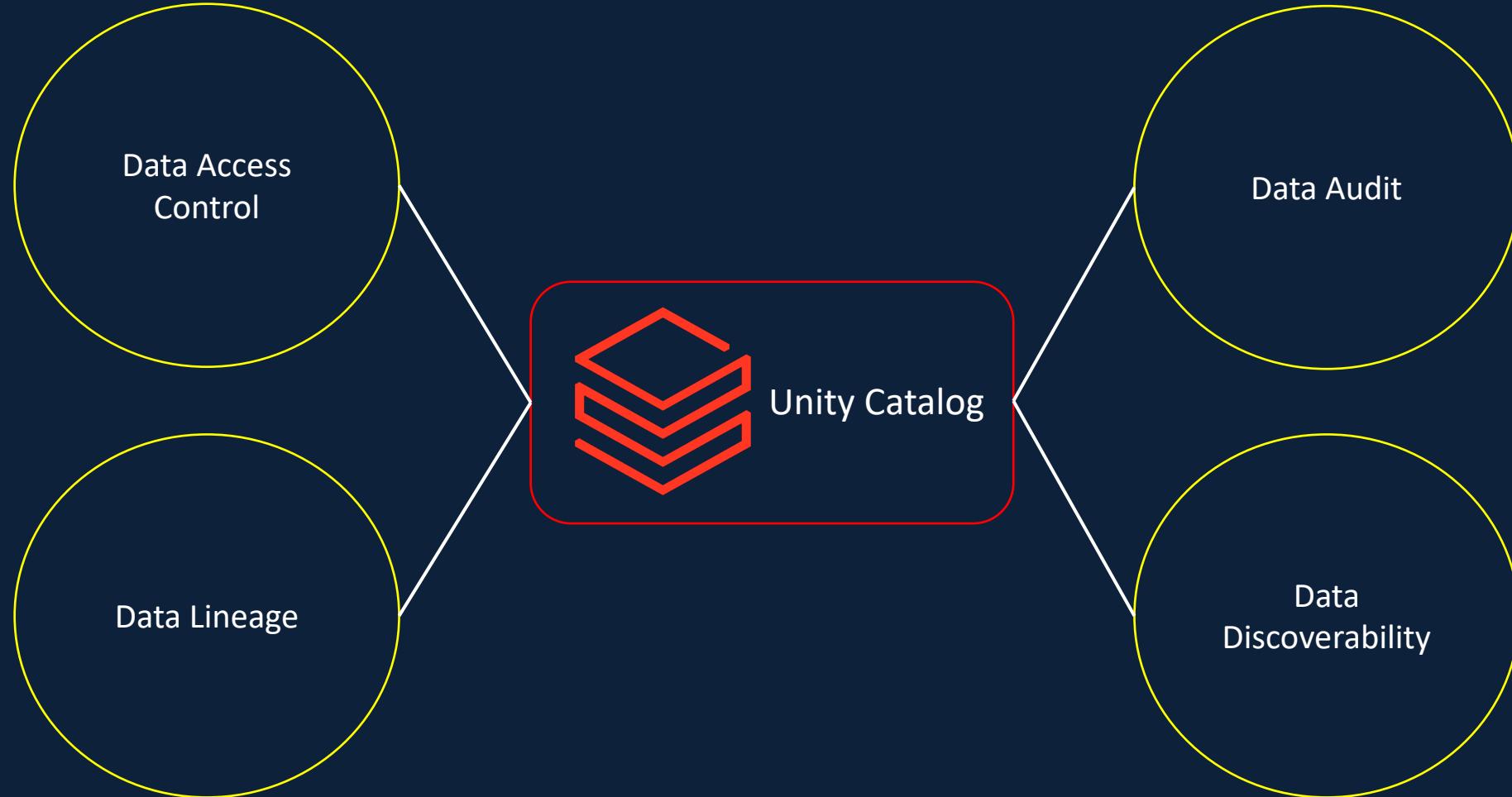
Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Discoverability



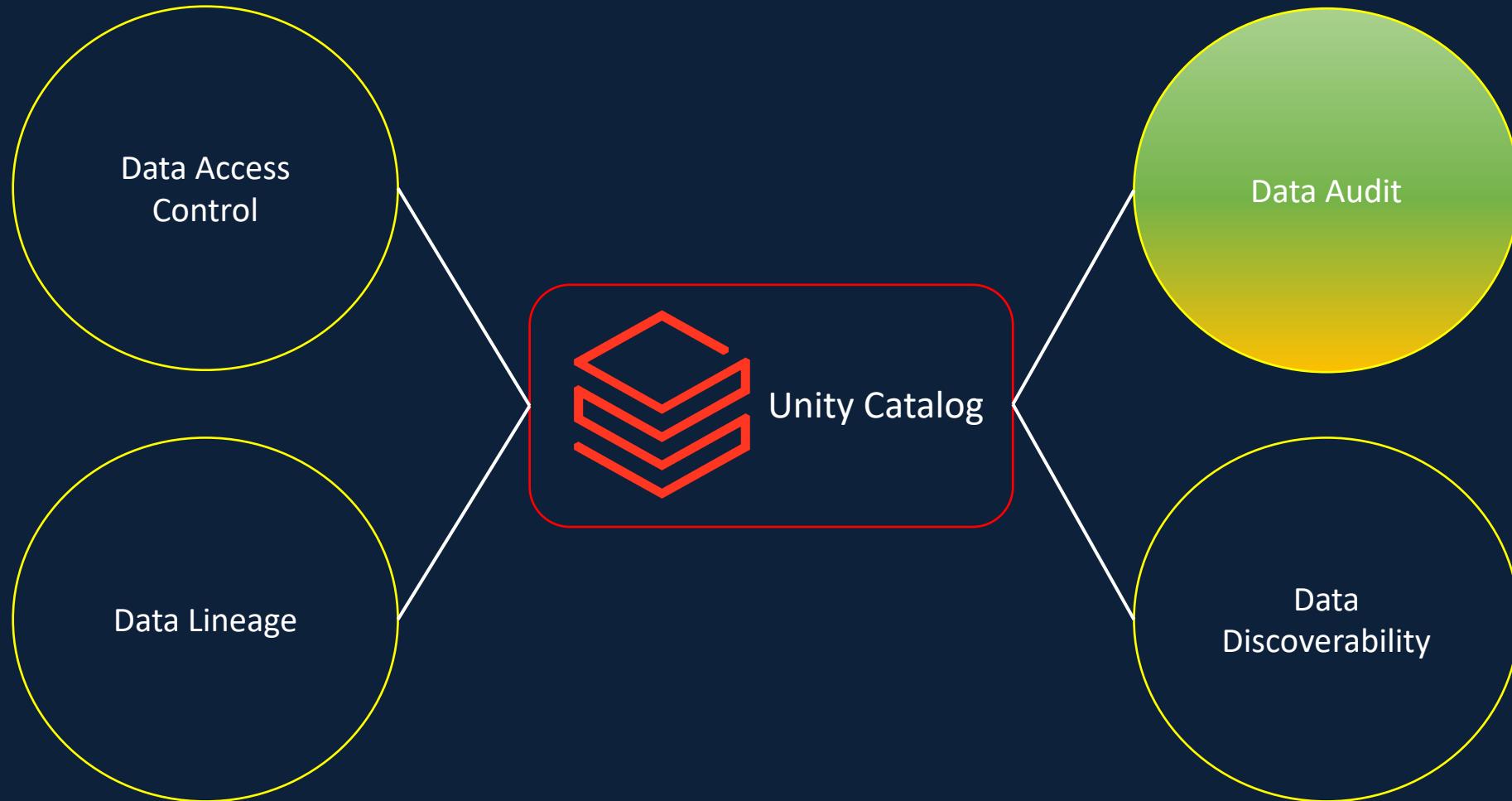
Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Audit



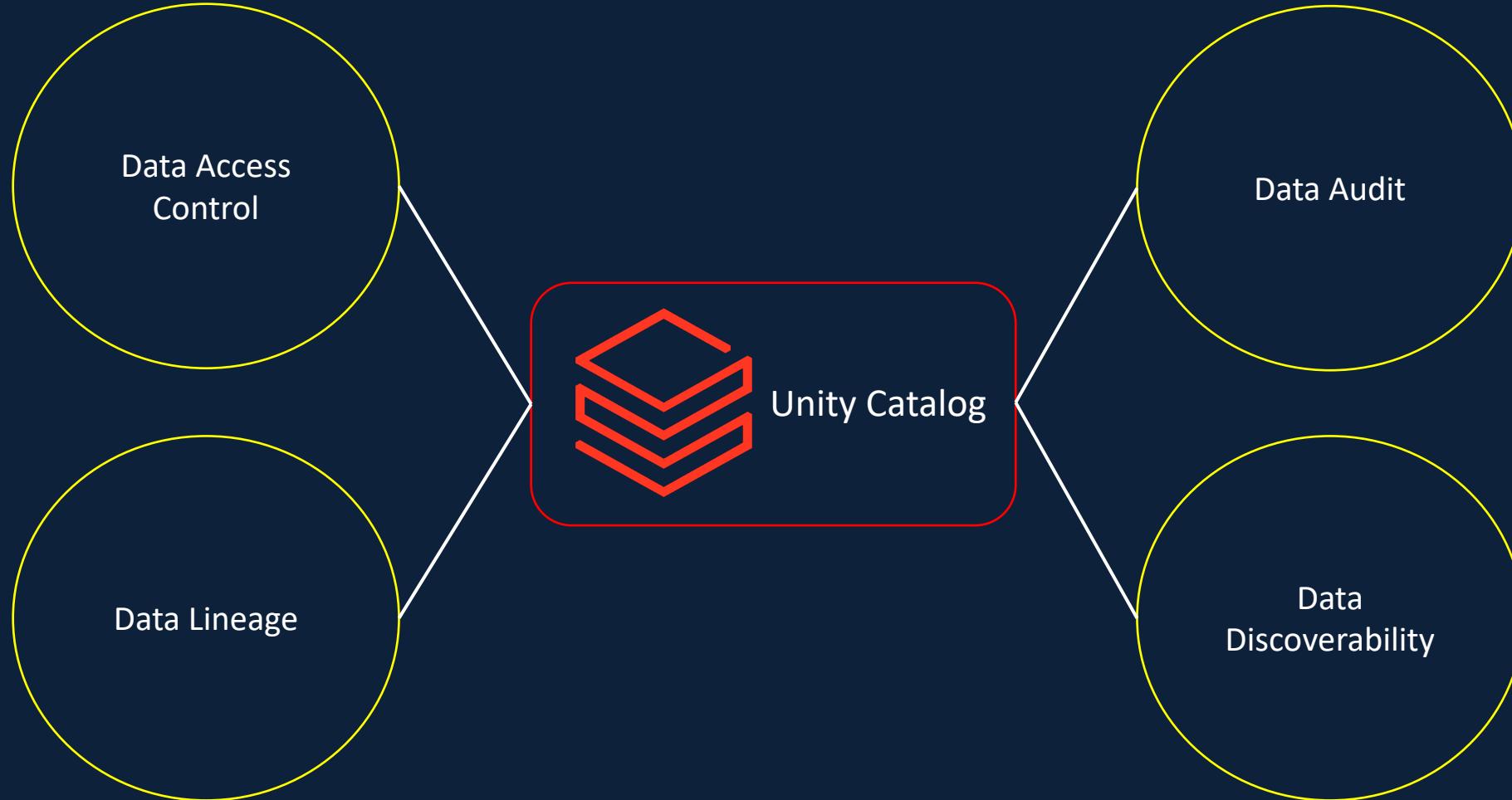
Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Discoverability



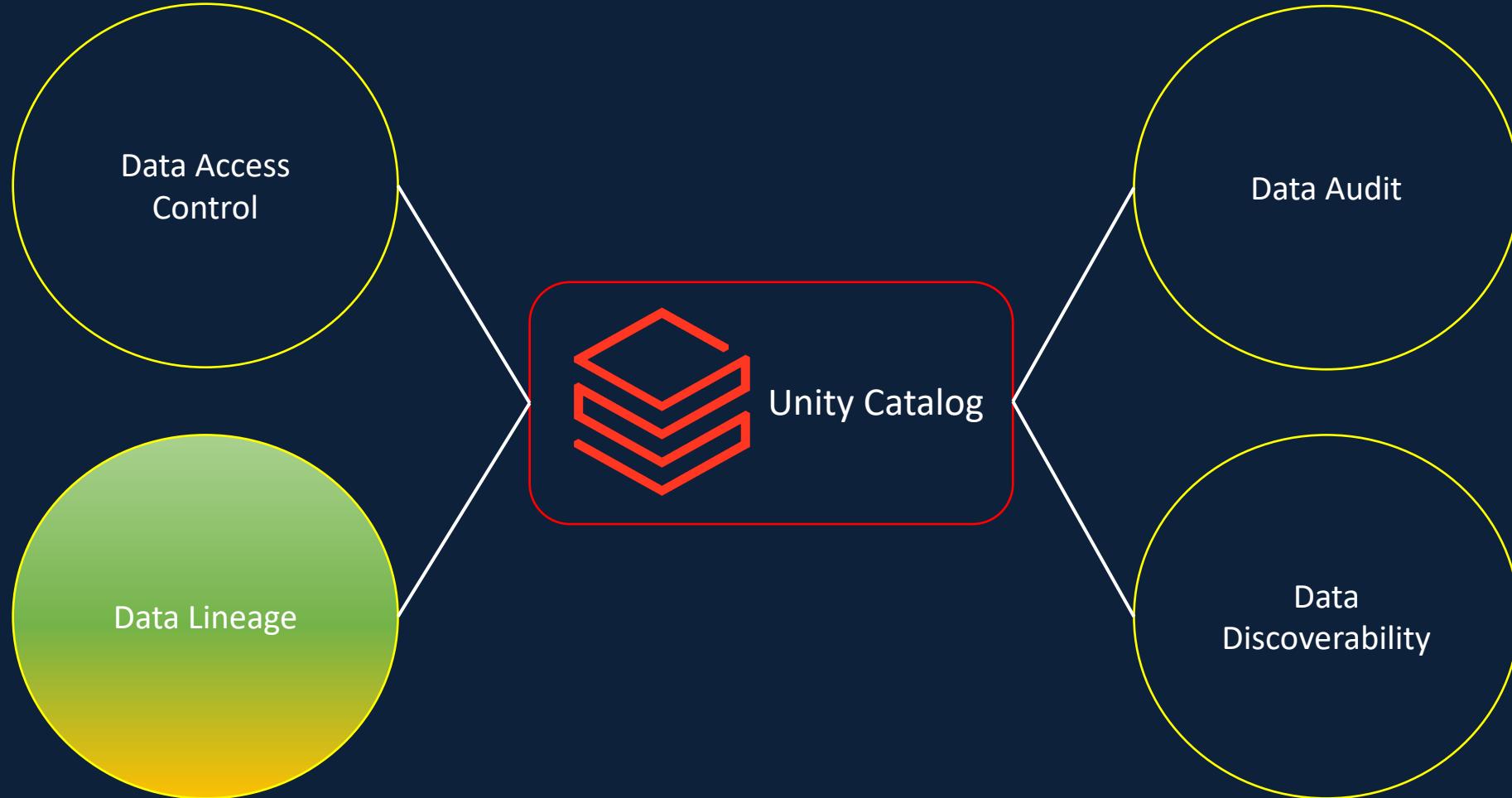
Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Lineage



Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Lineage



Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Data Lineage

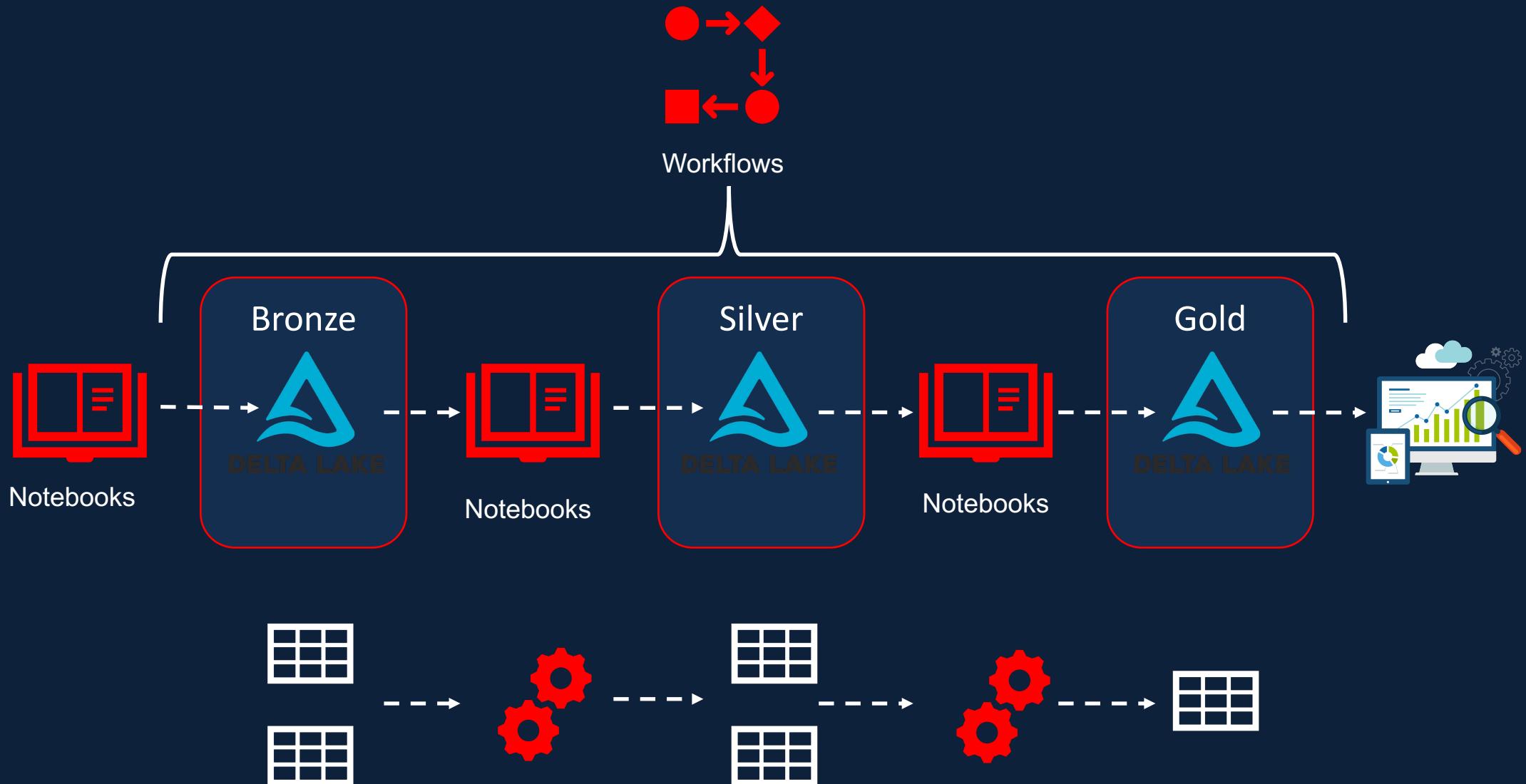
Data Lineage is the process of following/ tracking the journey of data within a pipeline.

What is the origin

How has it changed/ transformed

What is the destination

Data Lineage



Data Lineage - Benefits

Root cause analysis

Improved impact analysis

Trust & Reliability

Better compliance

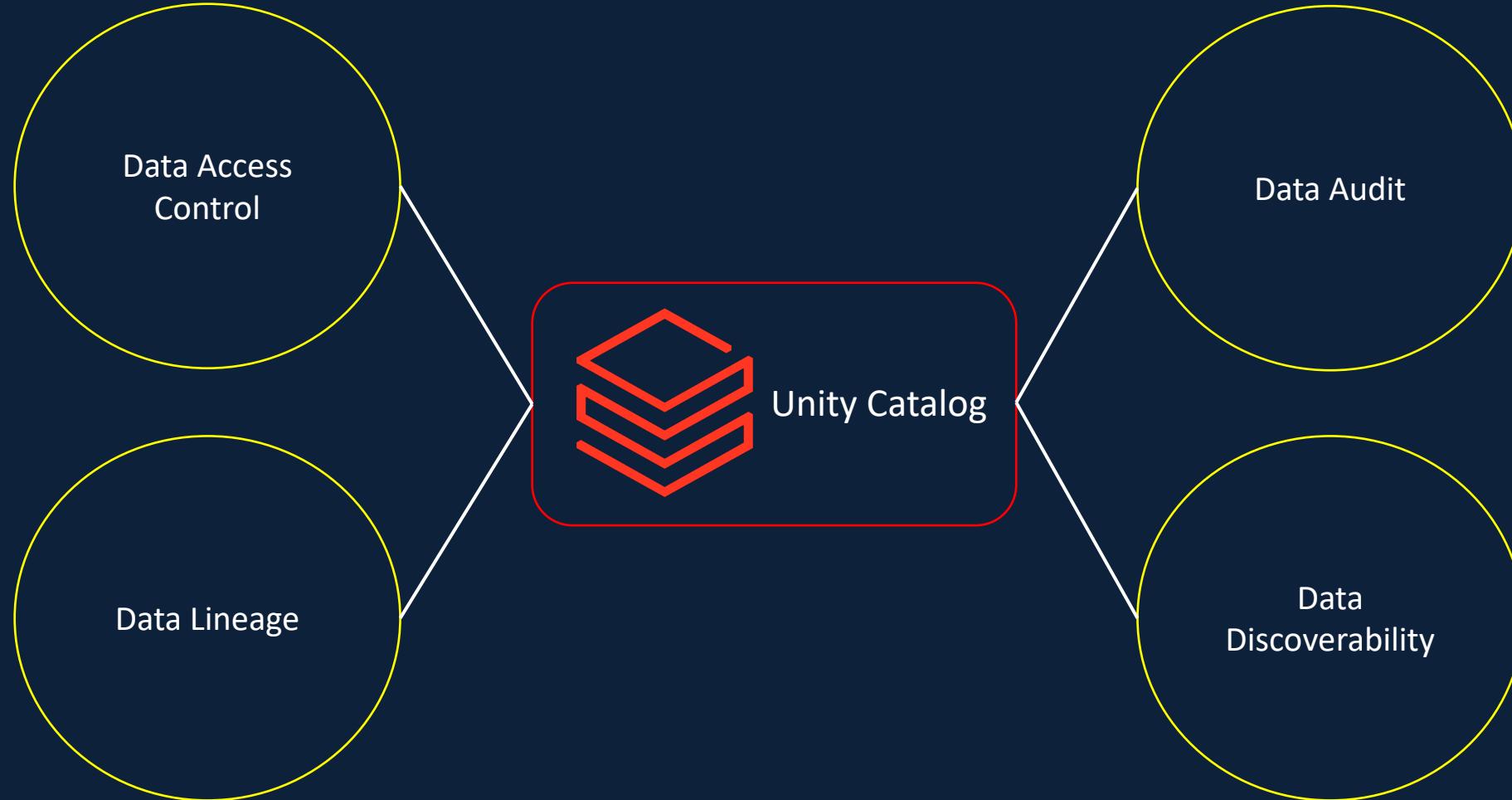
Data Lineage - Limitations

Only available for tables registered in Unity Catalog Metastore

Available only for the last 30 days

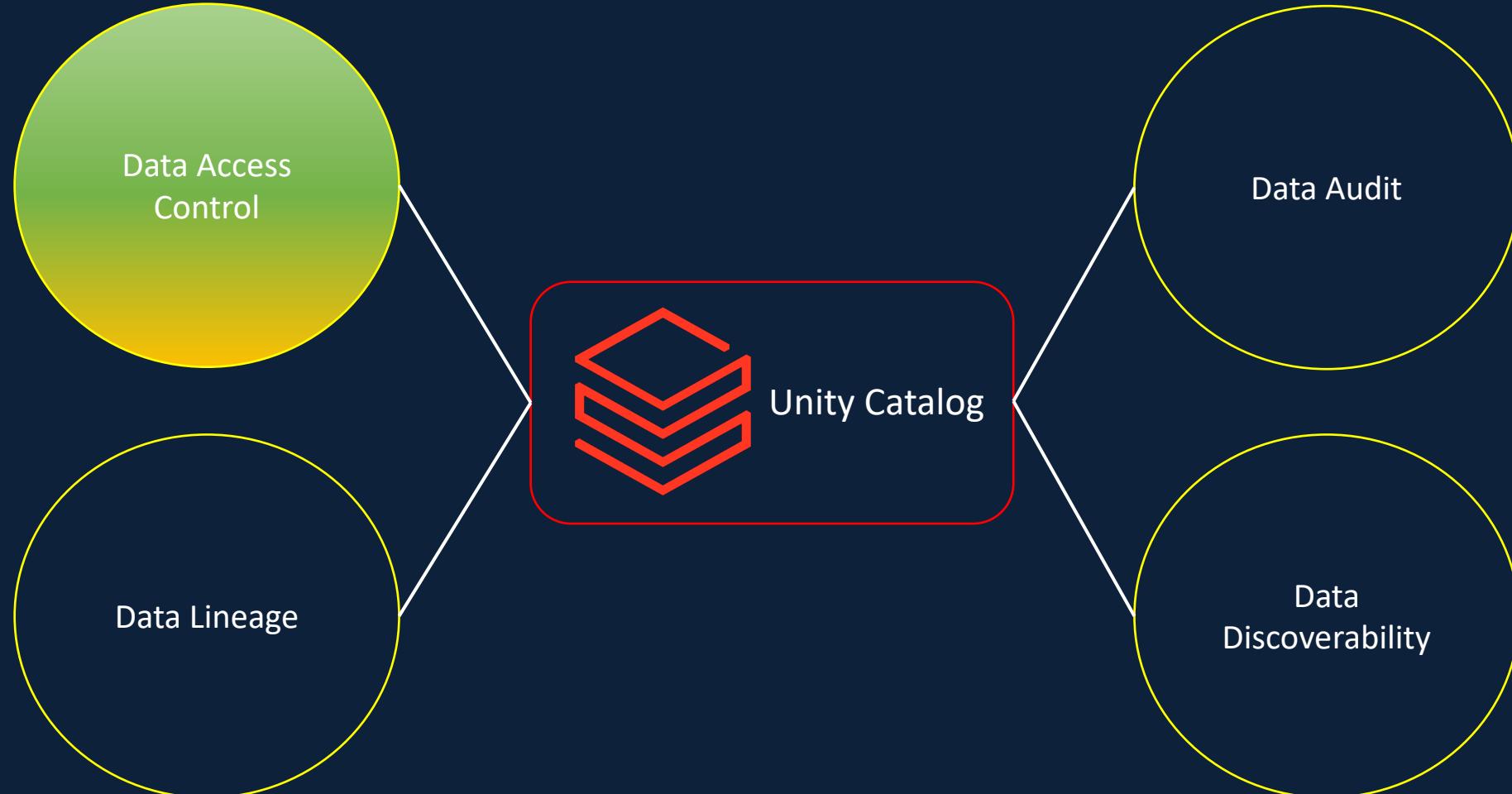
Limited column level lineage

Unity Catalog – Data Access Control



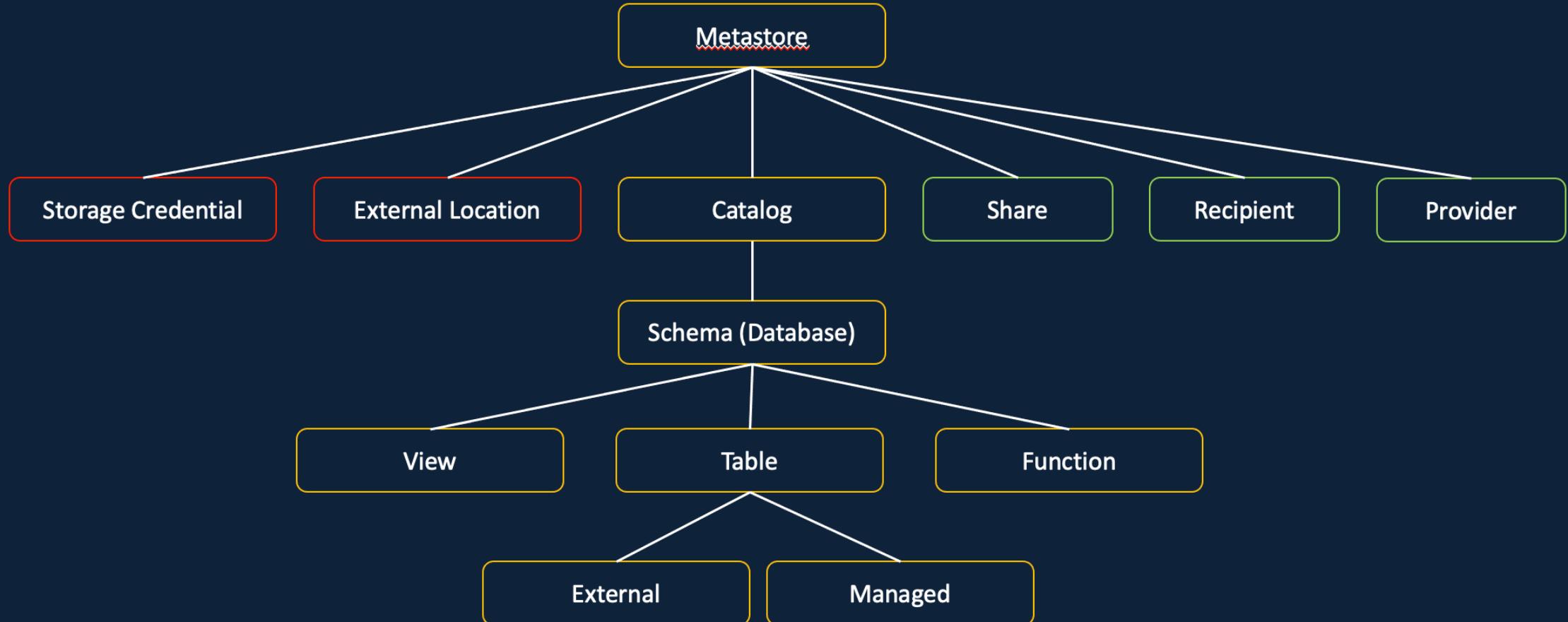
Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

Unity Catalog – Data Access Control

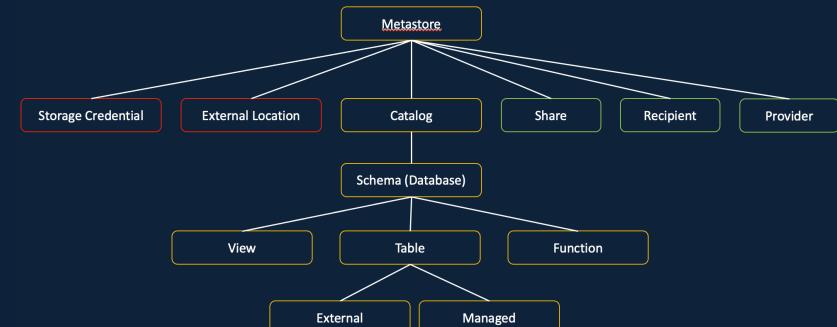
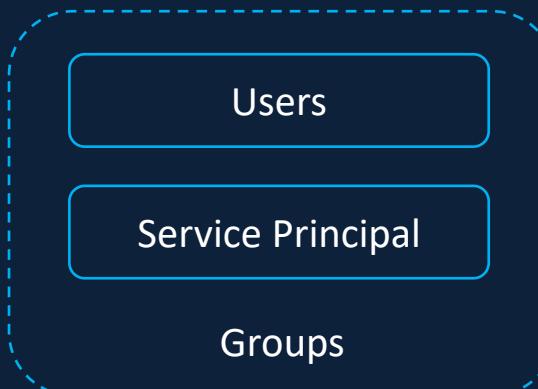


Unity Catalog is a Databricks offered **unified solution** for implementing **data governance** in the Data Lakehouse

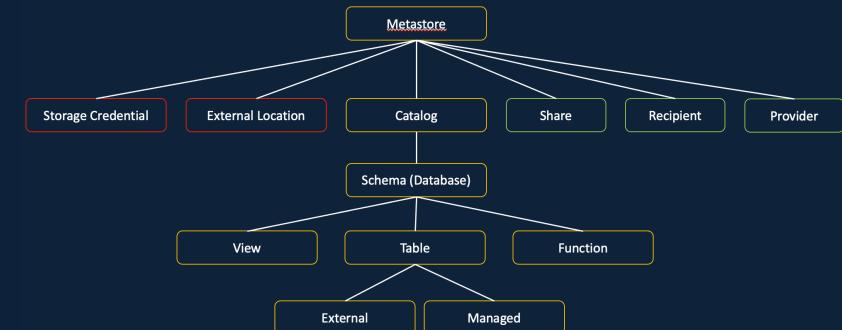
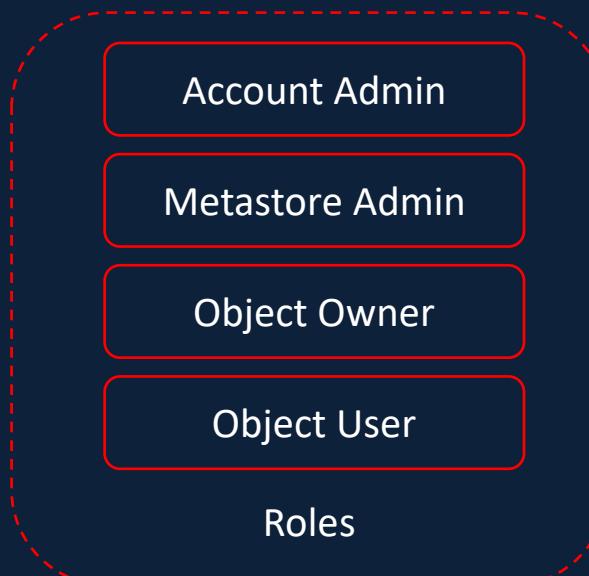
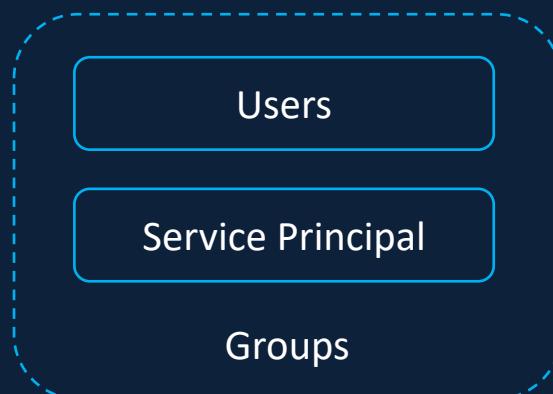
Unity Catalog - Securable Objects



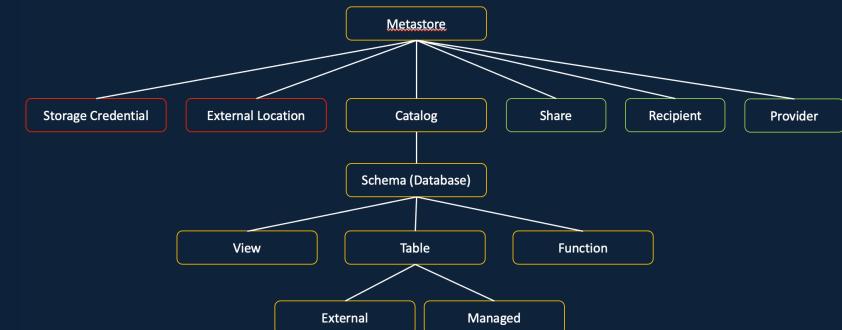
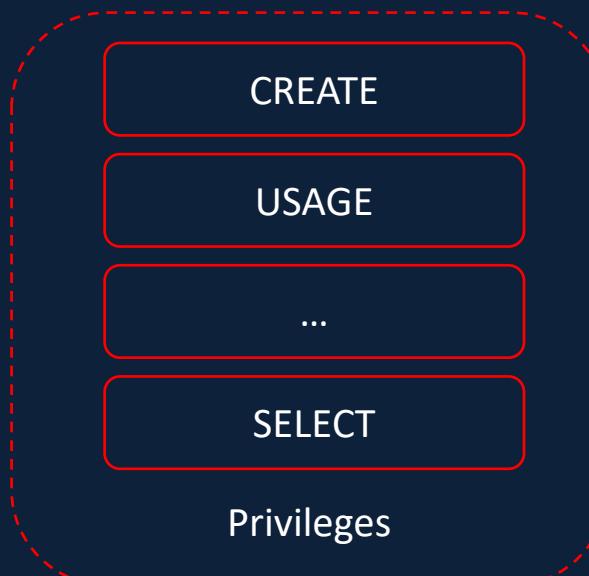
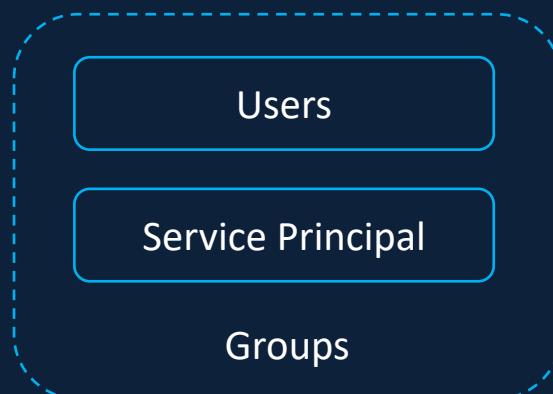
Unity Catalog - Security Model



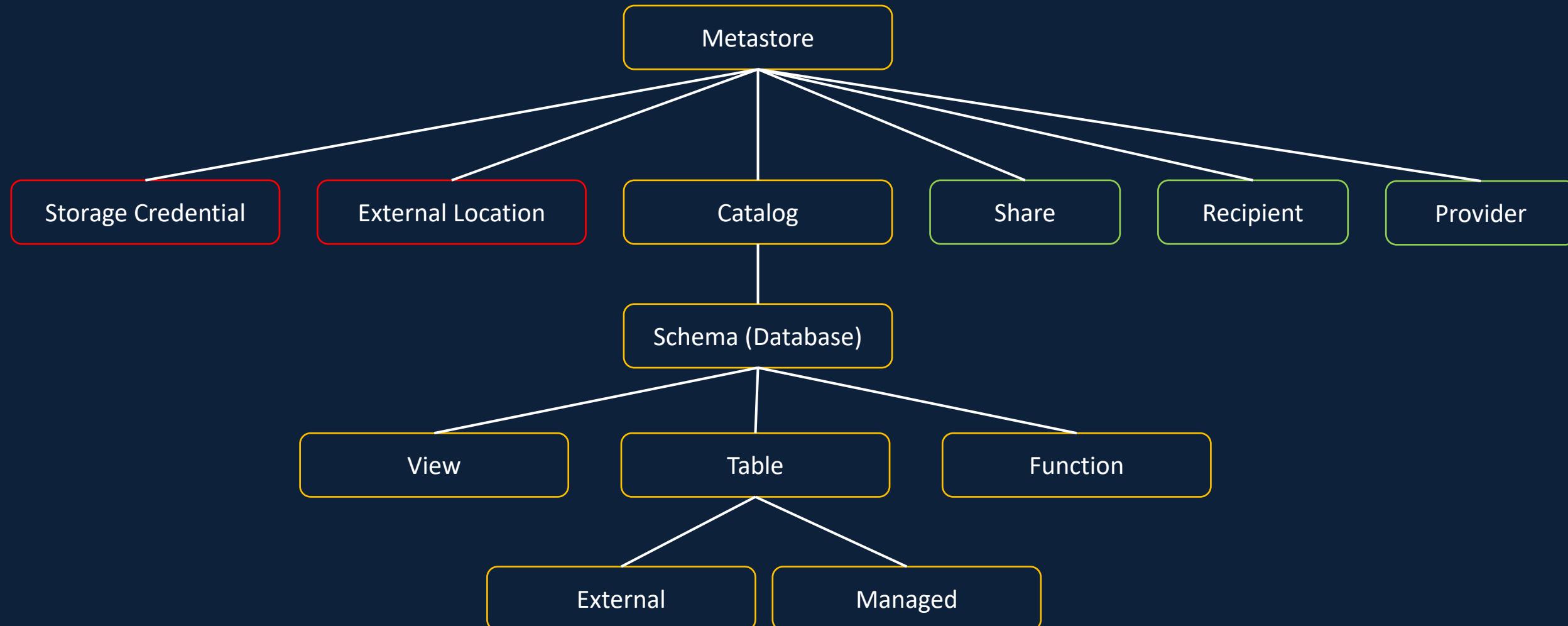
Security Model – Role Based Access



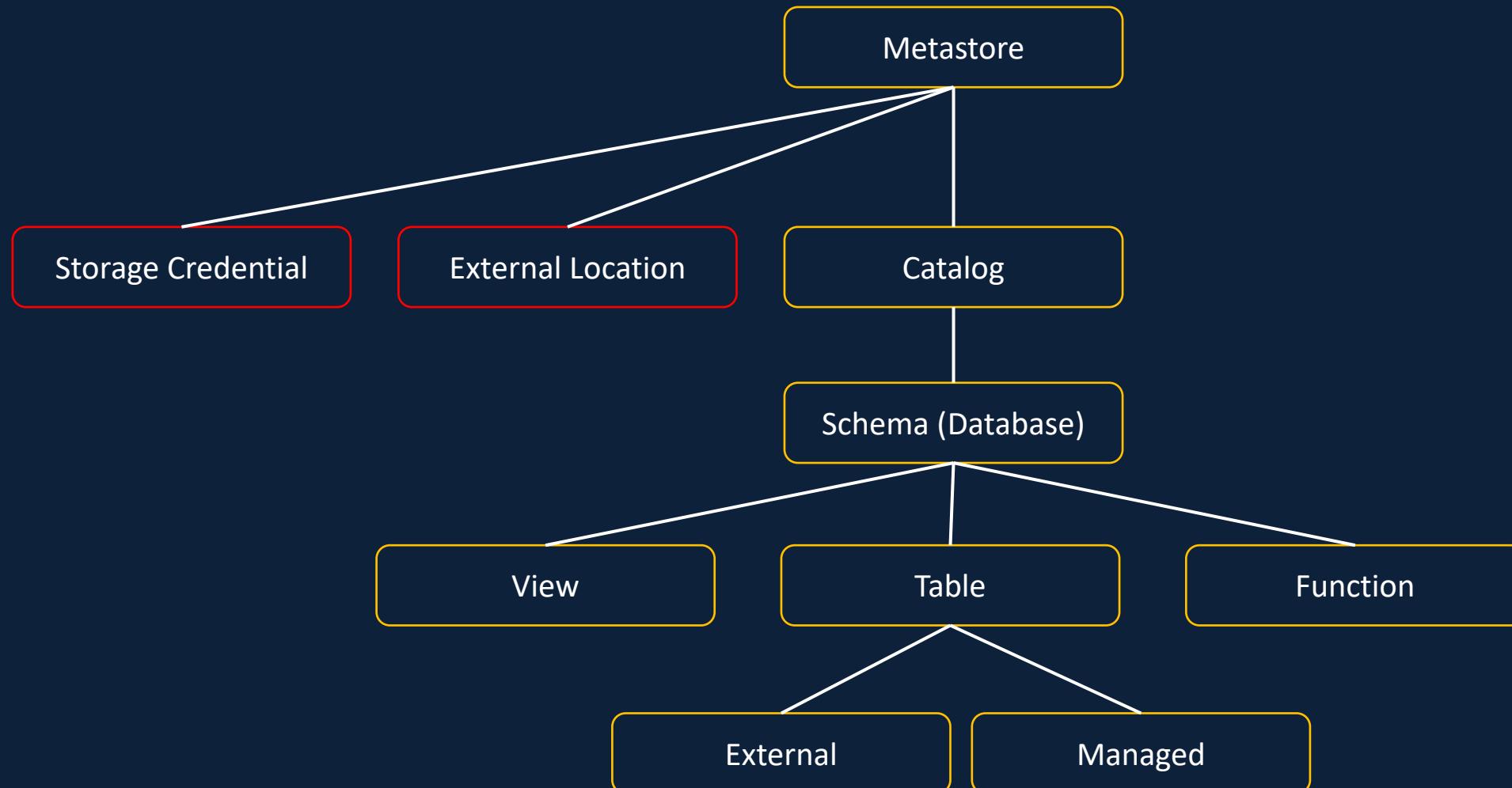
Security Model – Access Control List



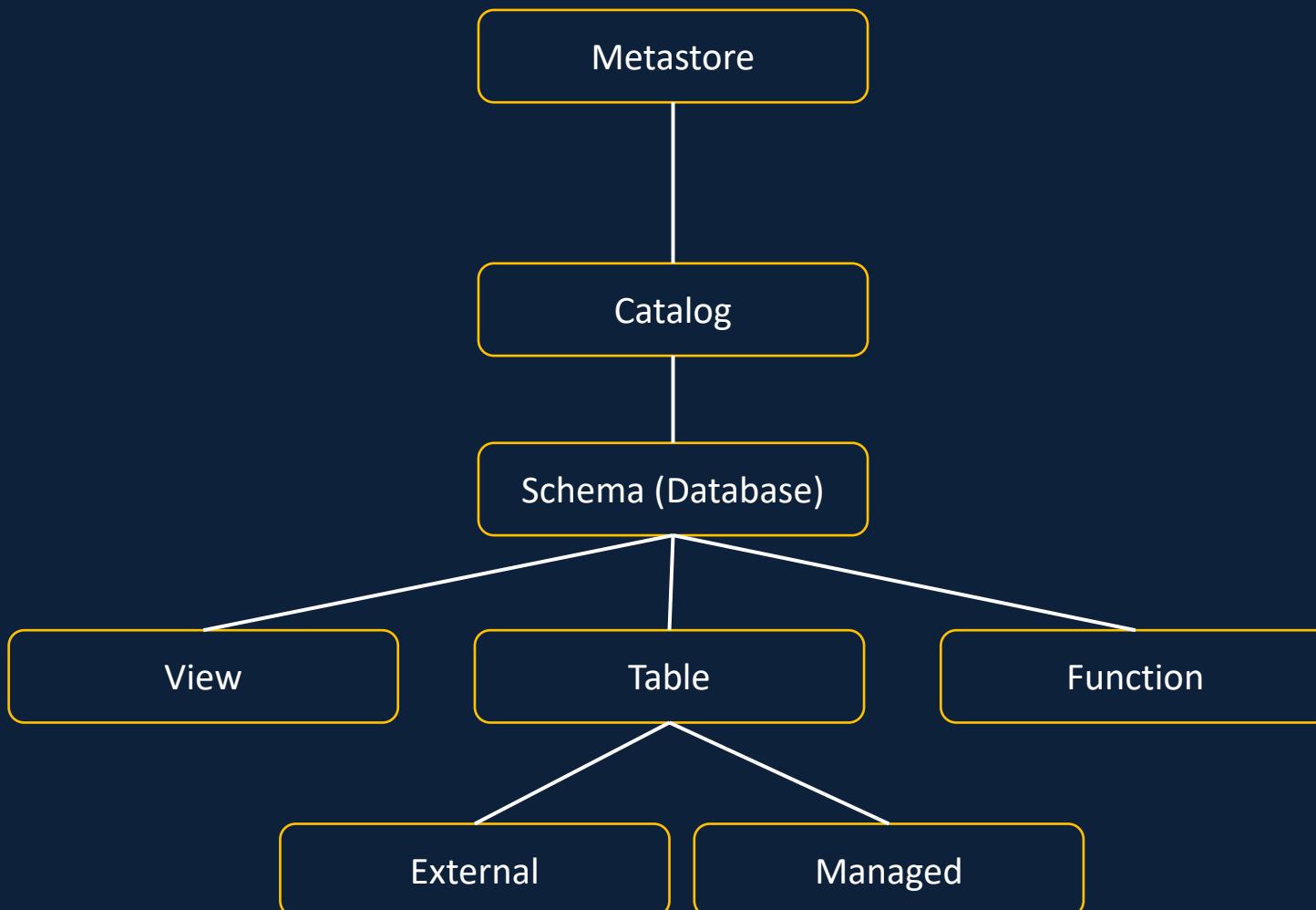
Security Model – Access Control List



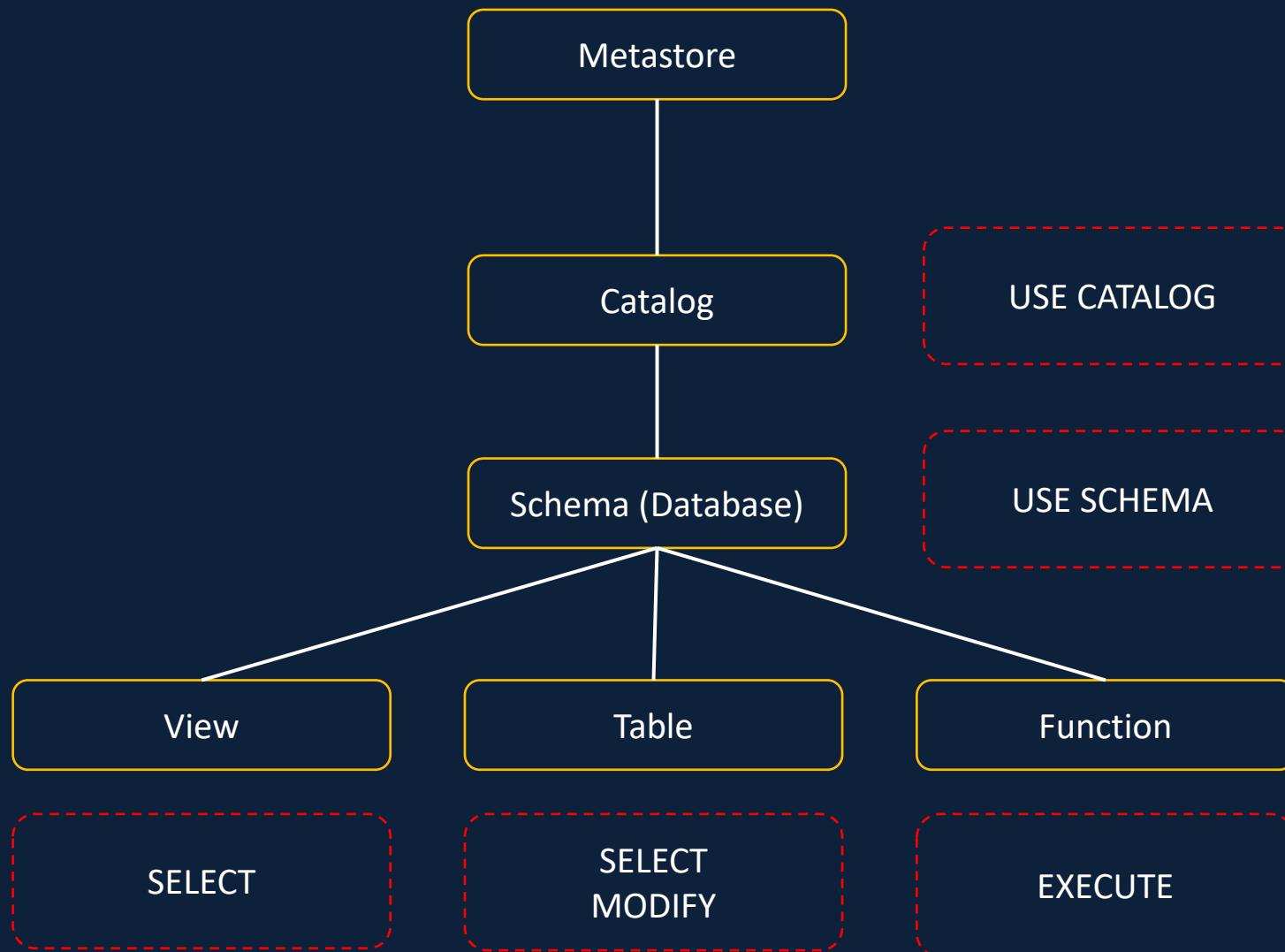
Security Model – Access Control List



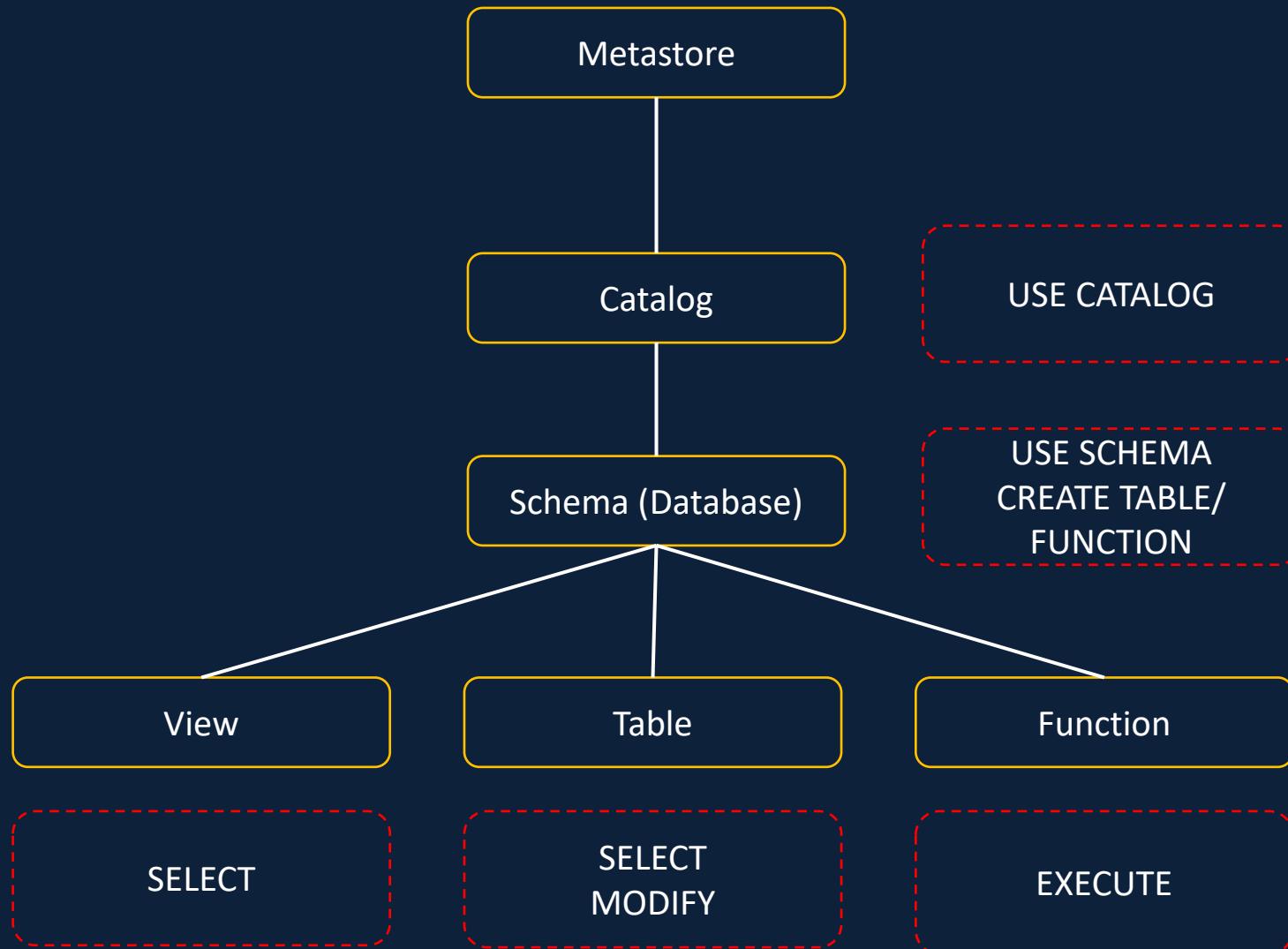
Security Model – Access Control List



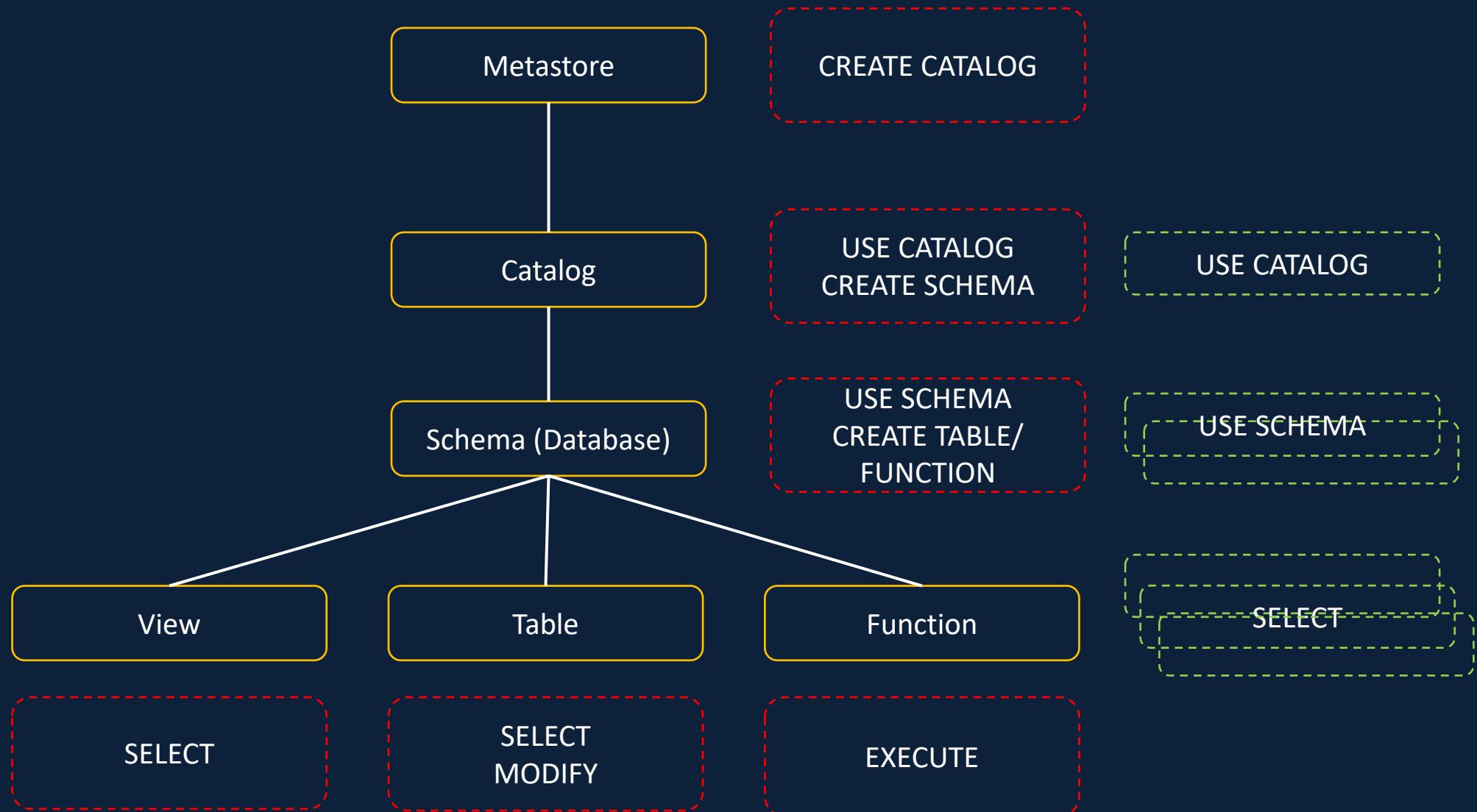
Security Model – Access Control List



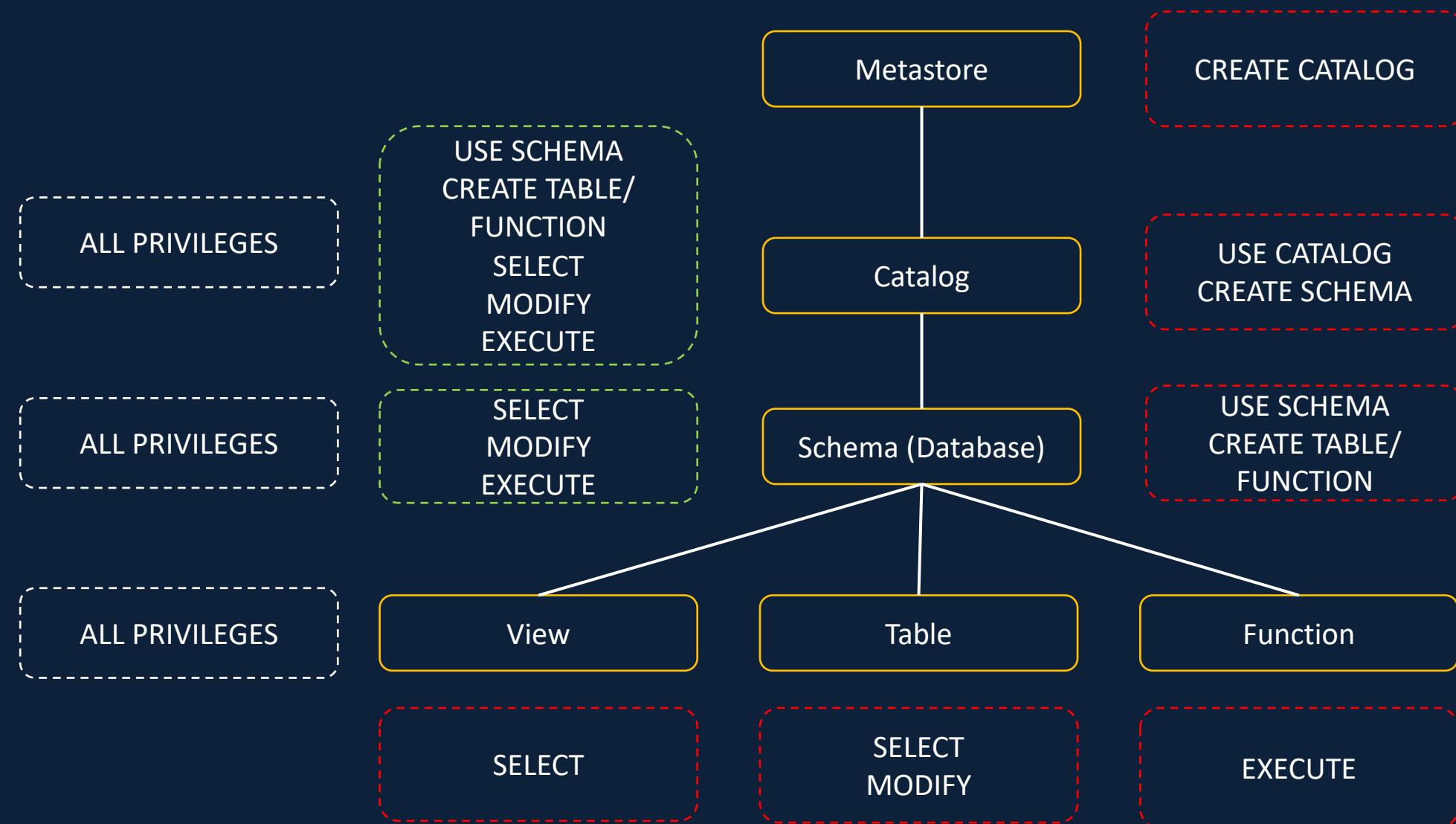
Security Model – Access Control List



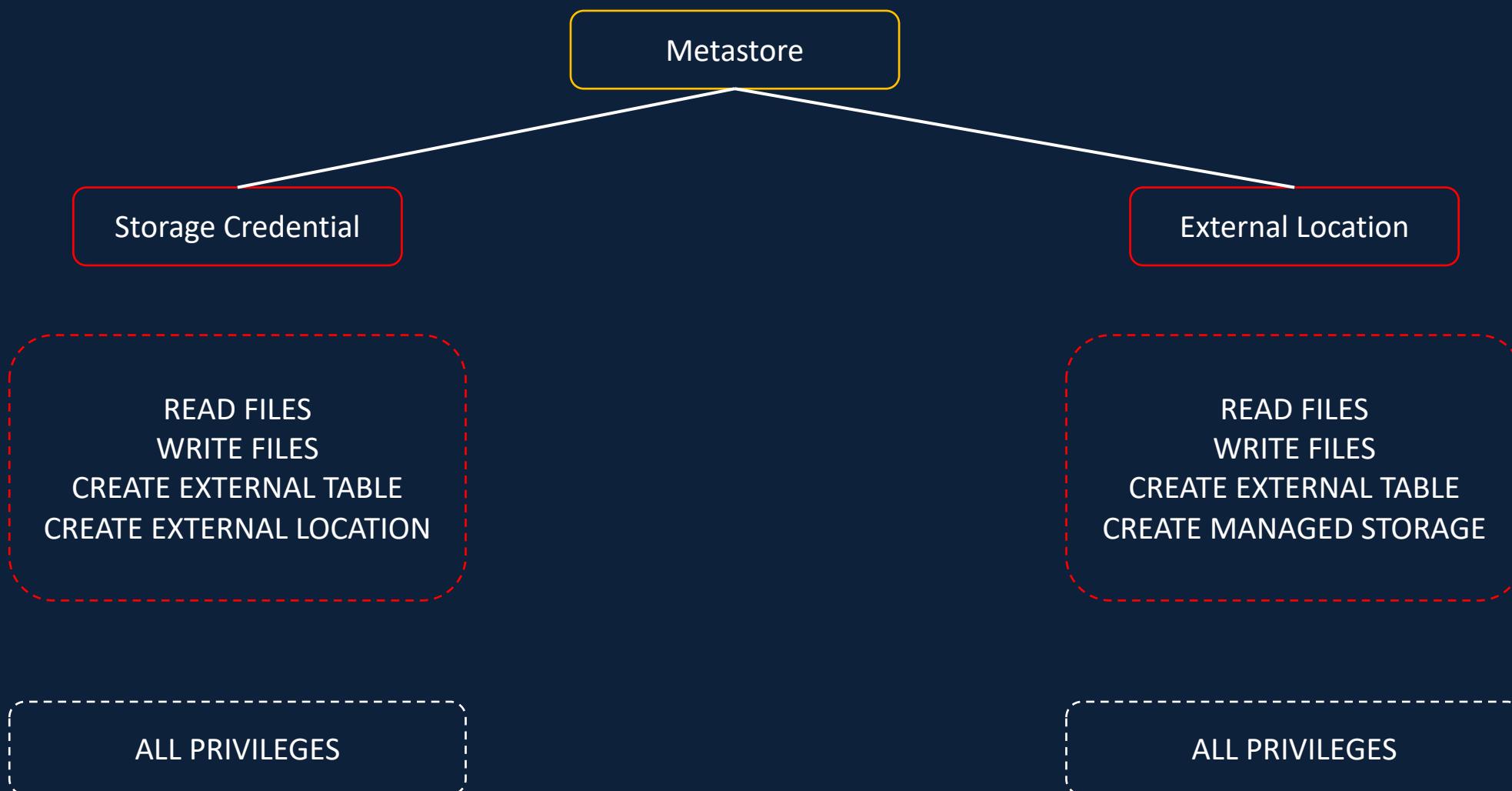
Security Model – Access Control List



Access Control List – Inheritance Model



Security Model – Access Control List



Congratulations!

&

Thank you

Feedback

Ratings & Review

Thank you
&
Good Luck!

Document History

Version	Date	Description
1	July 2021	Initial Version
2	Dec 2022	Sections 3 to 5 updated for UI Changes
3	Mar 2023	Revamped Databricks Mounts and Access to ADLS sections as a whole
4	May 2023	Addition of Unity Catalog