ADVANCING ANALYTICS

Unity Catalog

Databricks Governance

What Governance Problem?

What Problems Are We Addressing?

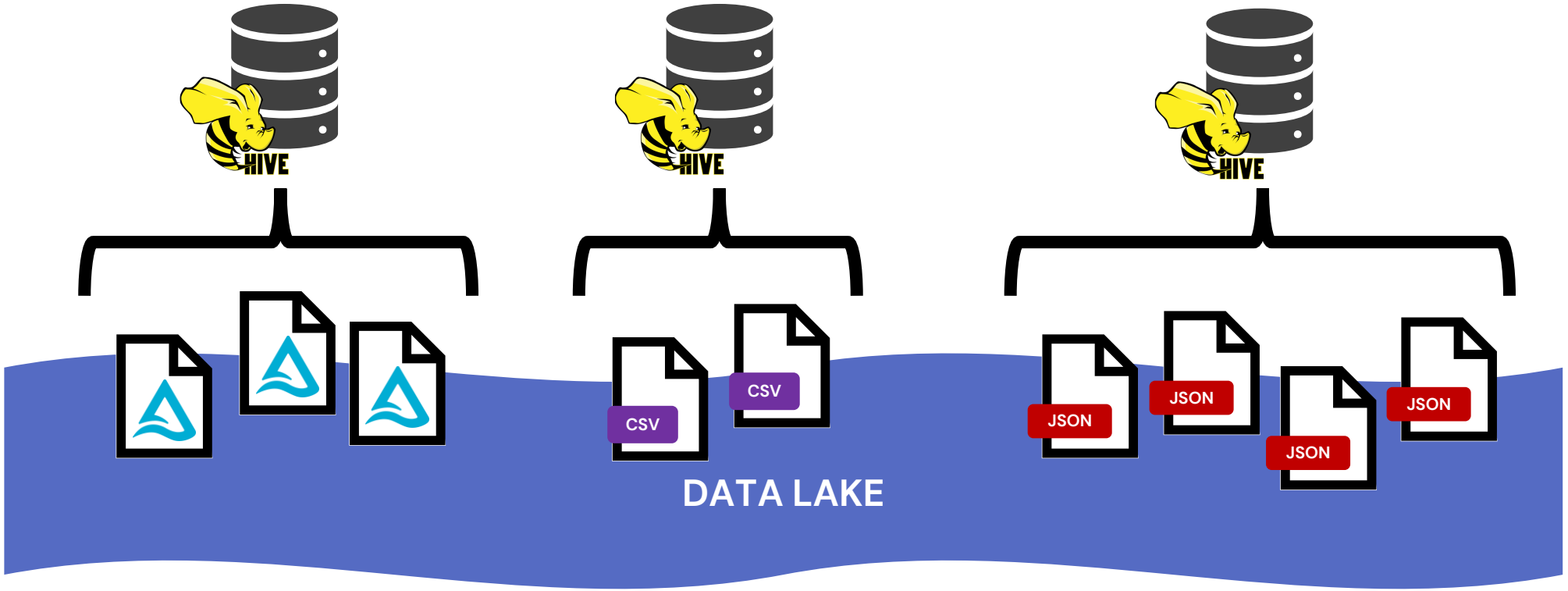
- Who has access to what data?
- What data is in the Lake?
- Where did the data come from? What is it used for?
- What do we know about the data?
- How can we securely share data across estate boundaries?

ADVANCING ANALYTICS

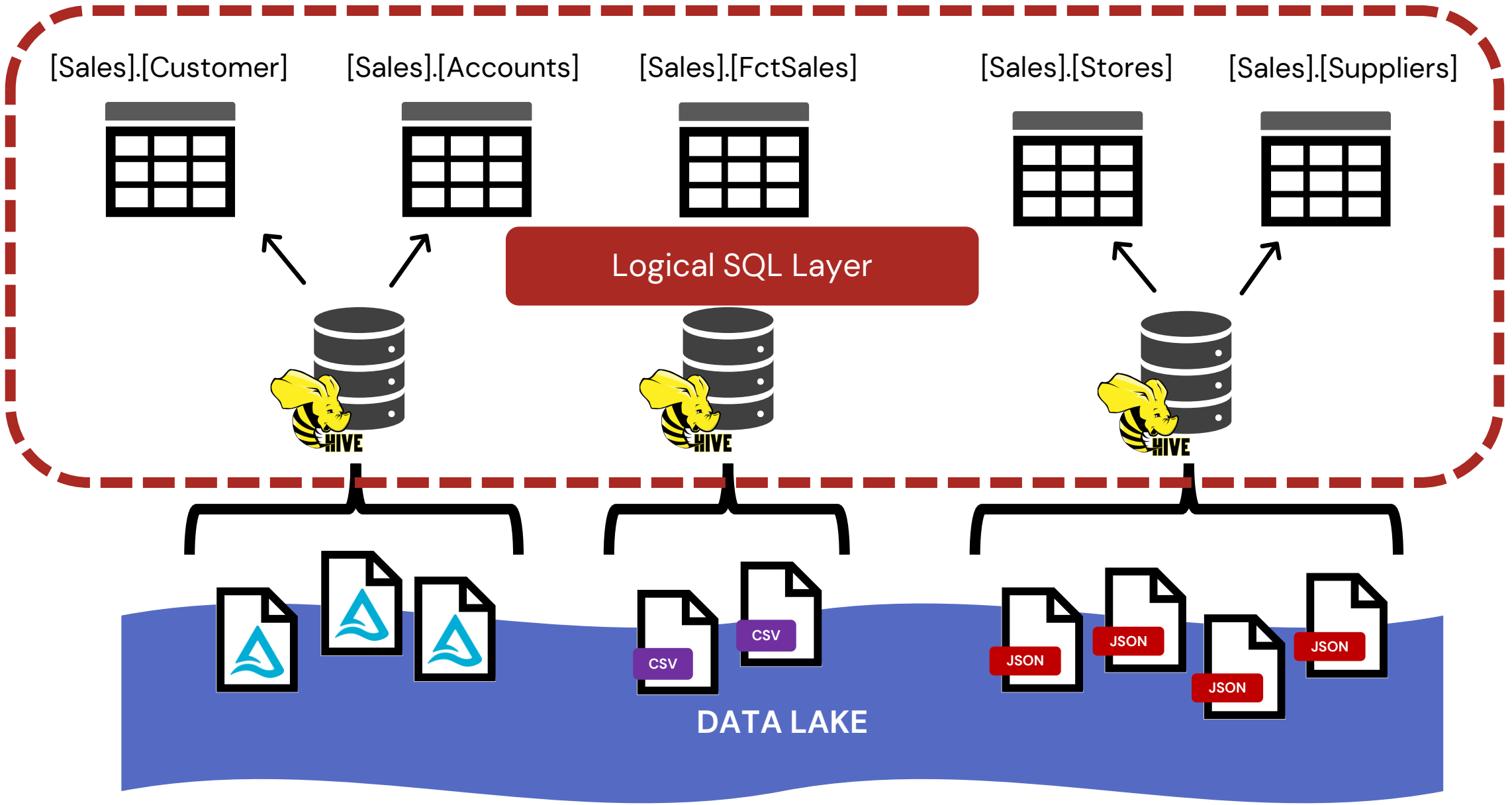
Legacy Databricks Metastore

The Hive Metastore (Apache Hive) is a data warehouse system built on top of Apache Hadoop. It enables users to read, write, and manage large datasets stored in distributed storage using SQL.

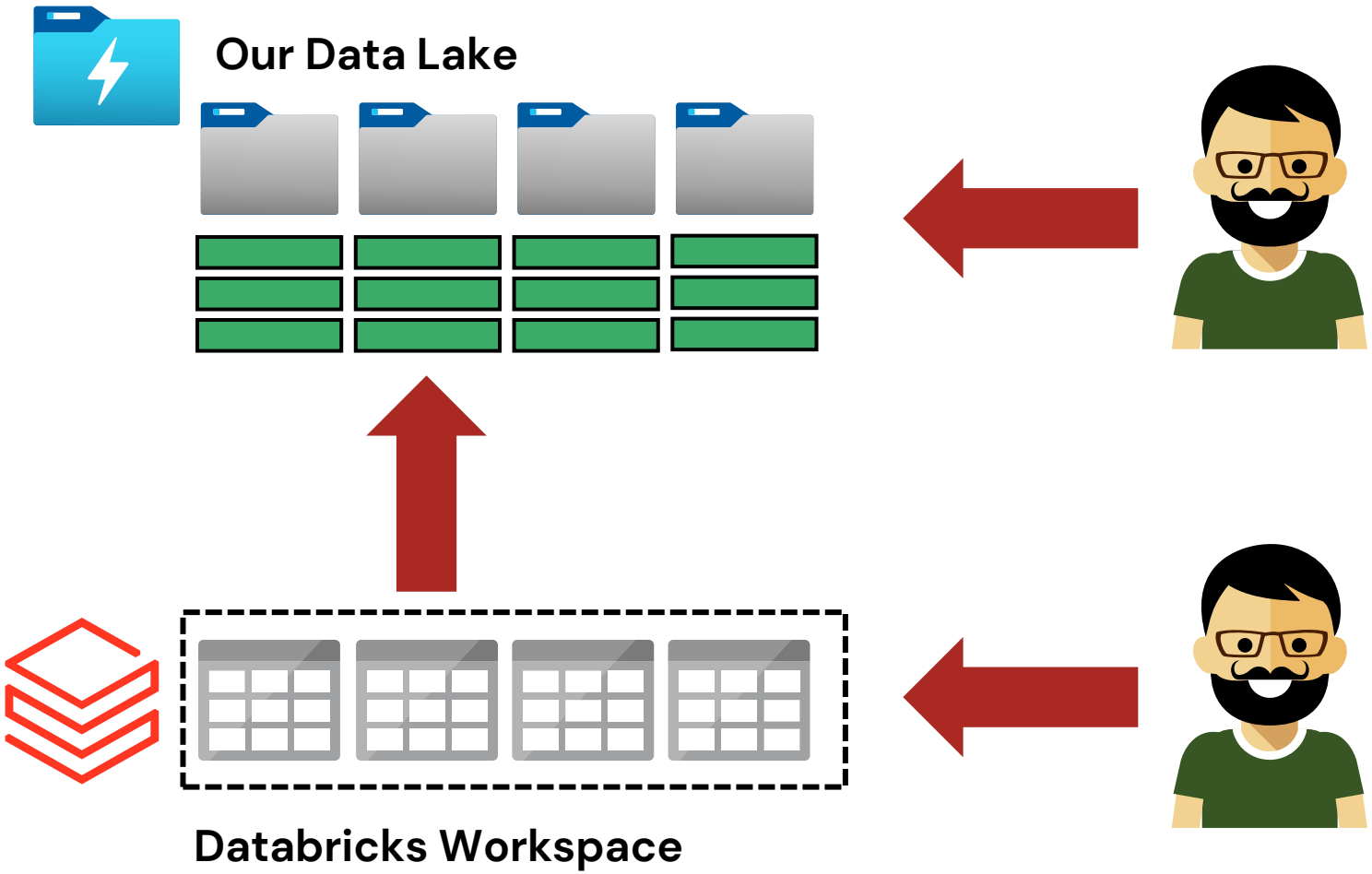
The Hive metastore in Databricks was used to manage and store metadata for entities like databases, tables, and columns.



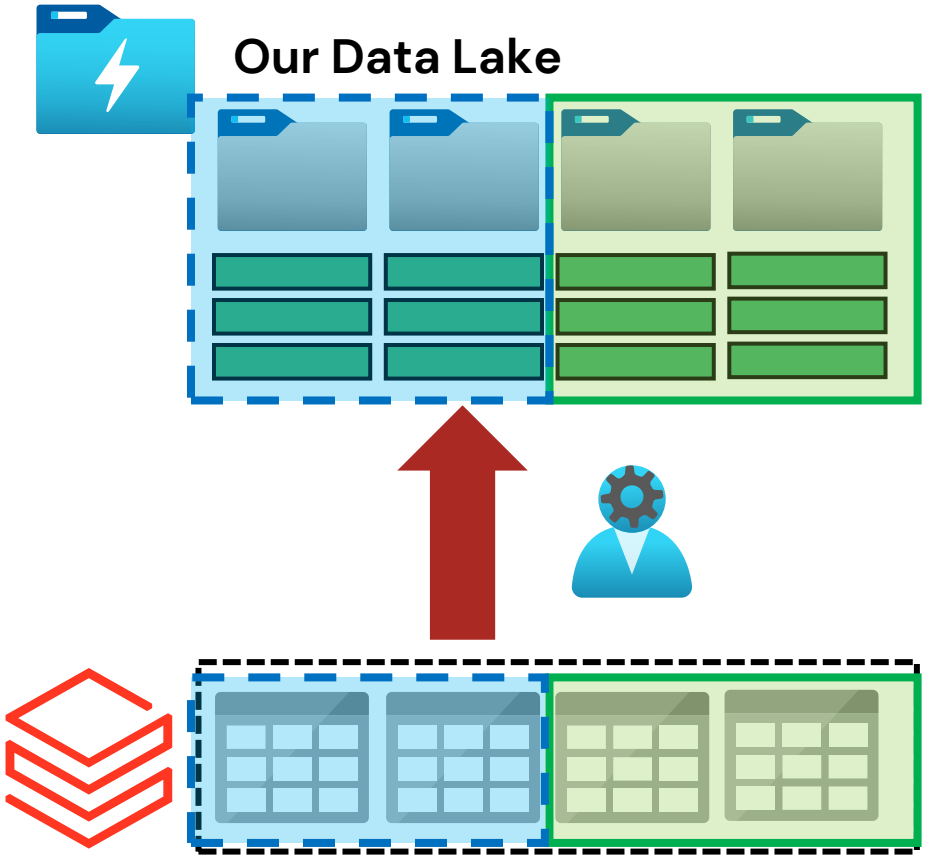
Legacy Databricks Metastore

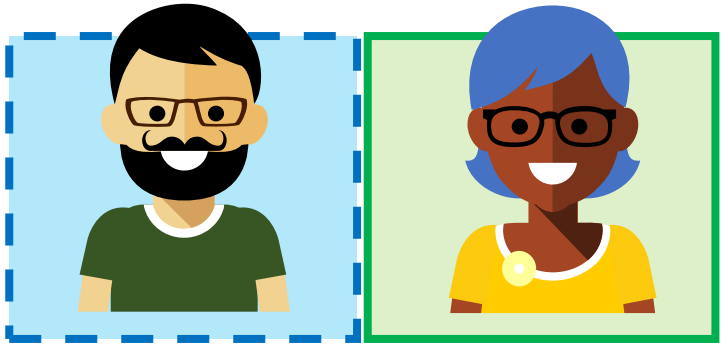


The Problem

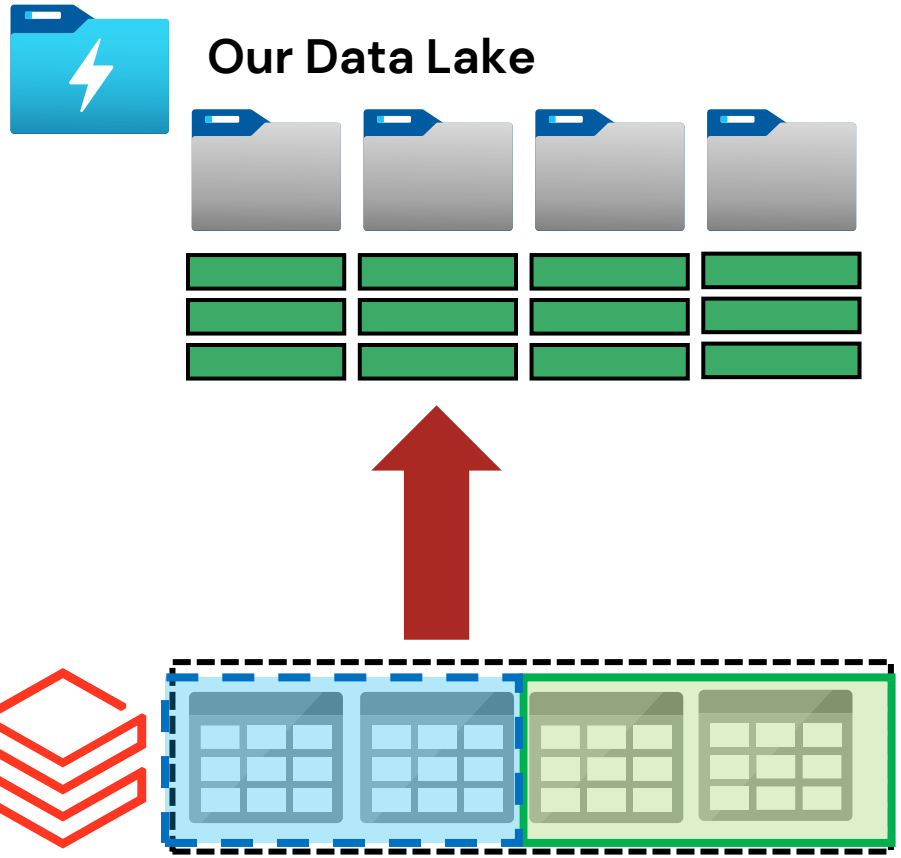


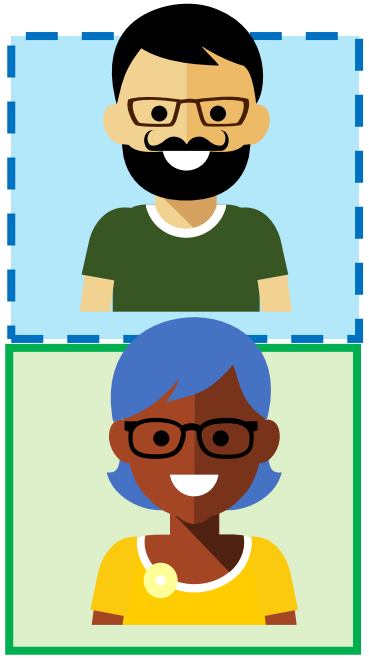
The Problem



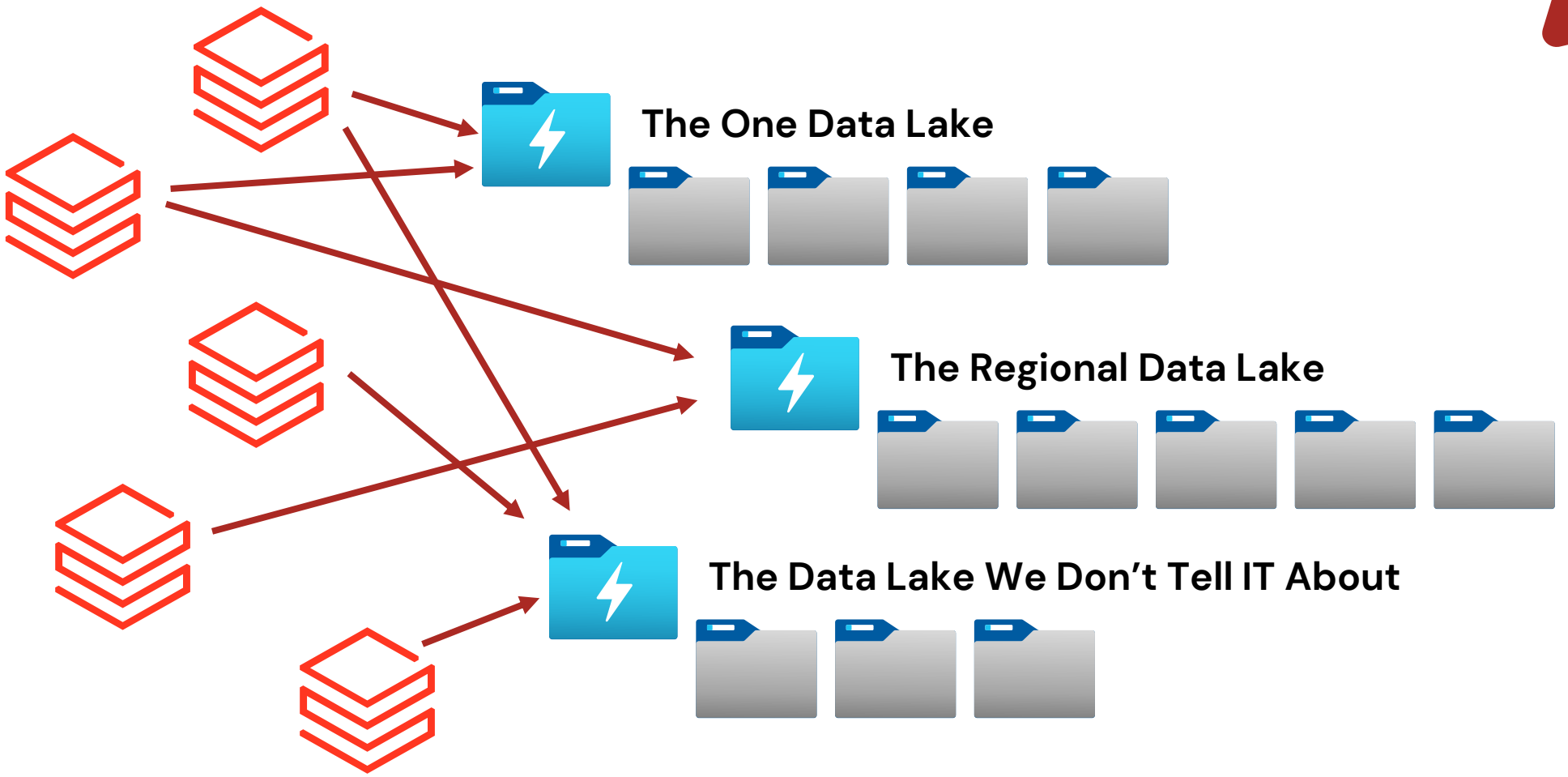


The Problem


Our Data Lake
Engineering Workspace
Data Science Workspace

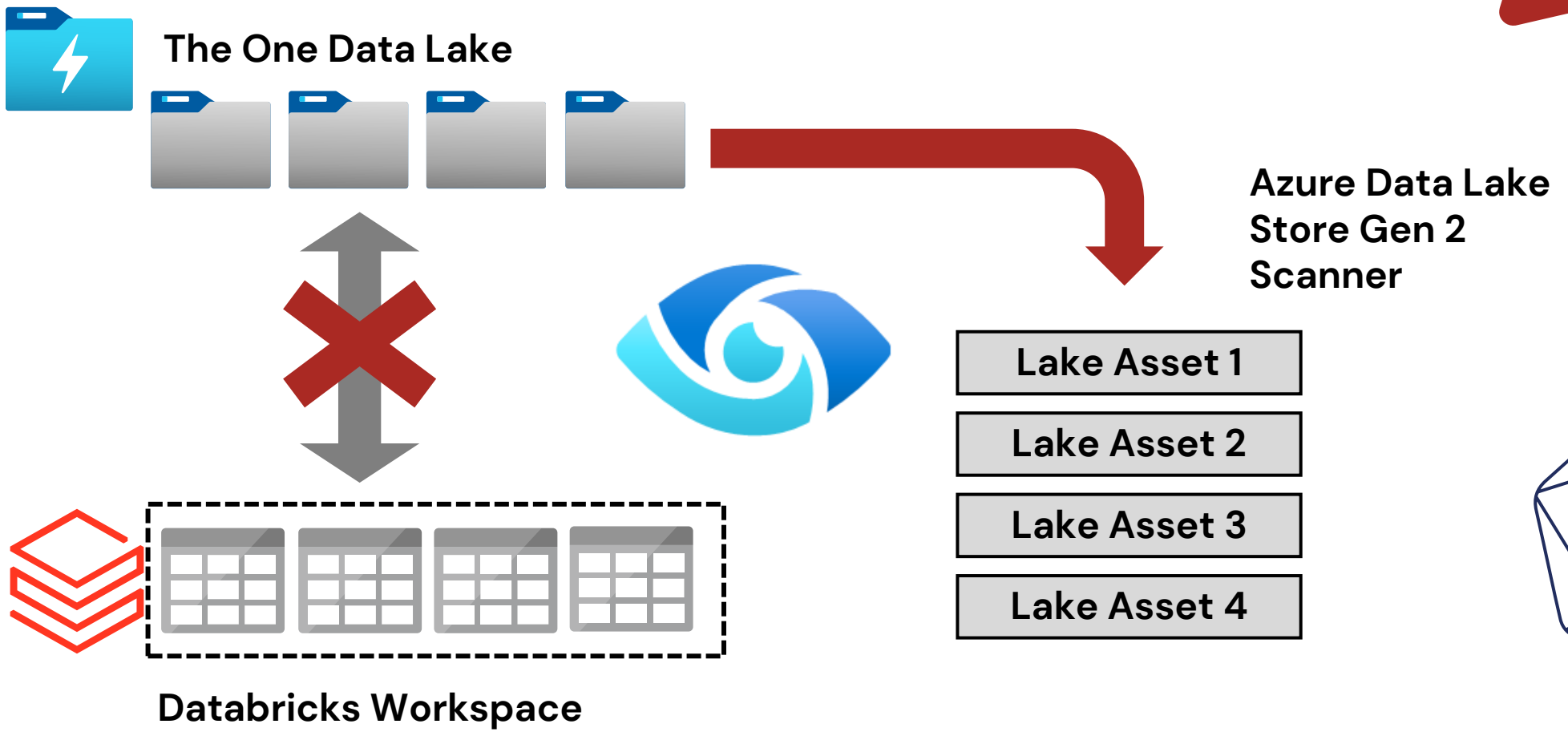


It's Rarely So Straight Forward



ADVANCING ANALYTICS

Azure Purview



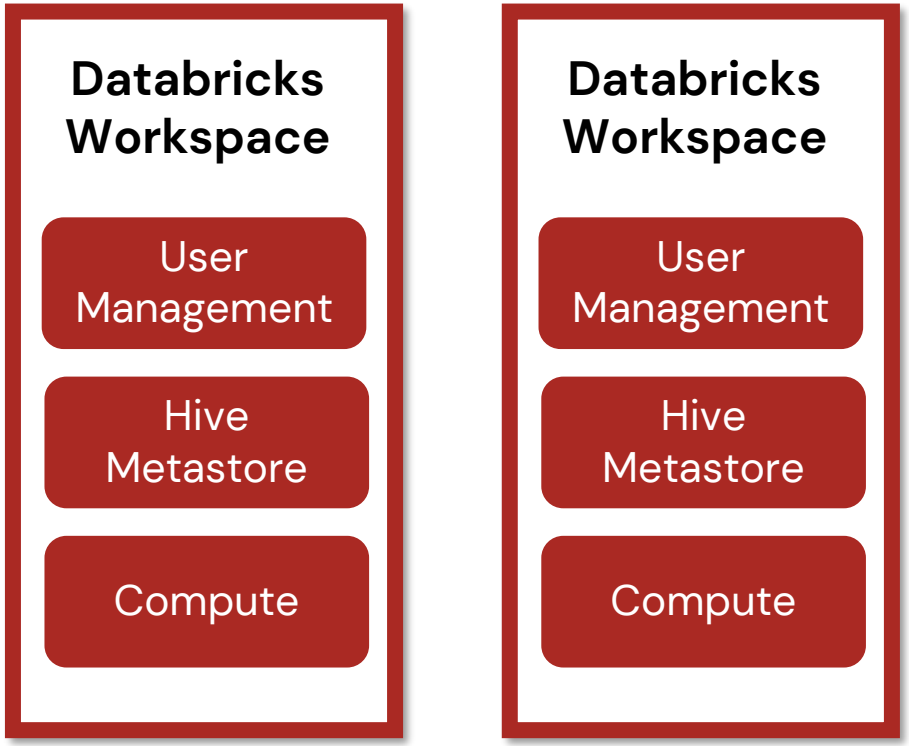
ADVANCING ANALYTICS

Account Console, Metastores & Workspace

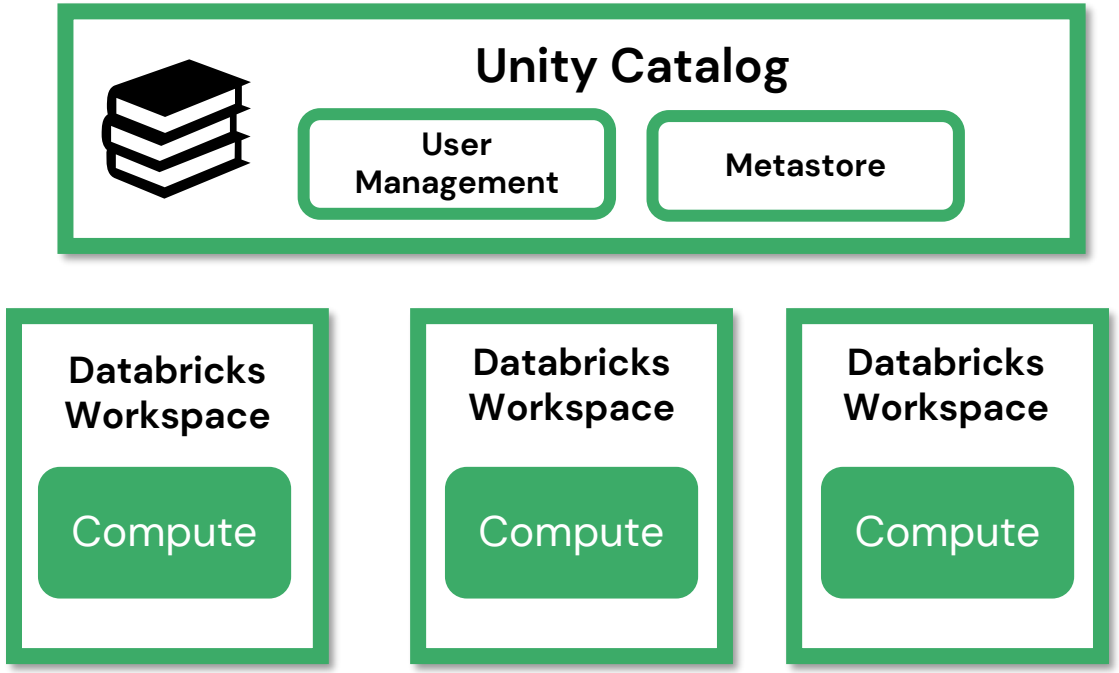
Databricks Unity Catalog (UC)

Unity Catalog provides centralised access control, auditing, lineage, and data discovery capabilities across Azure Databricks workspaces.

Legacy Databricks

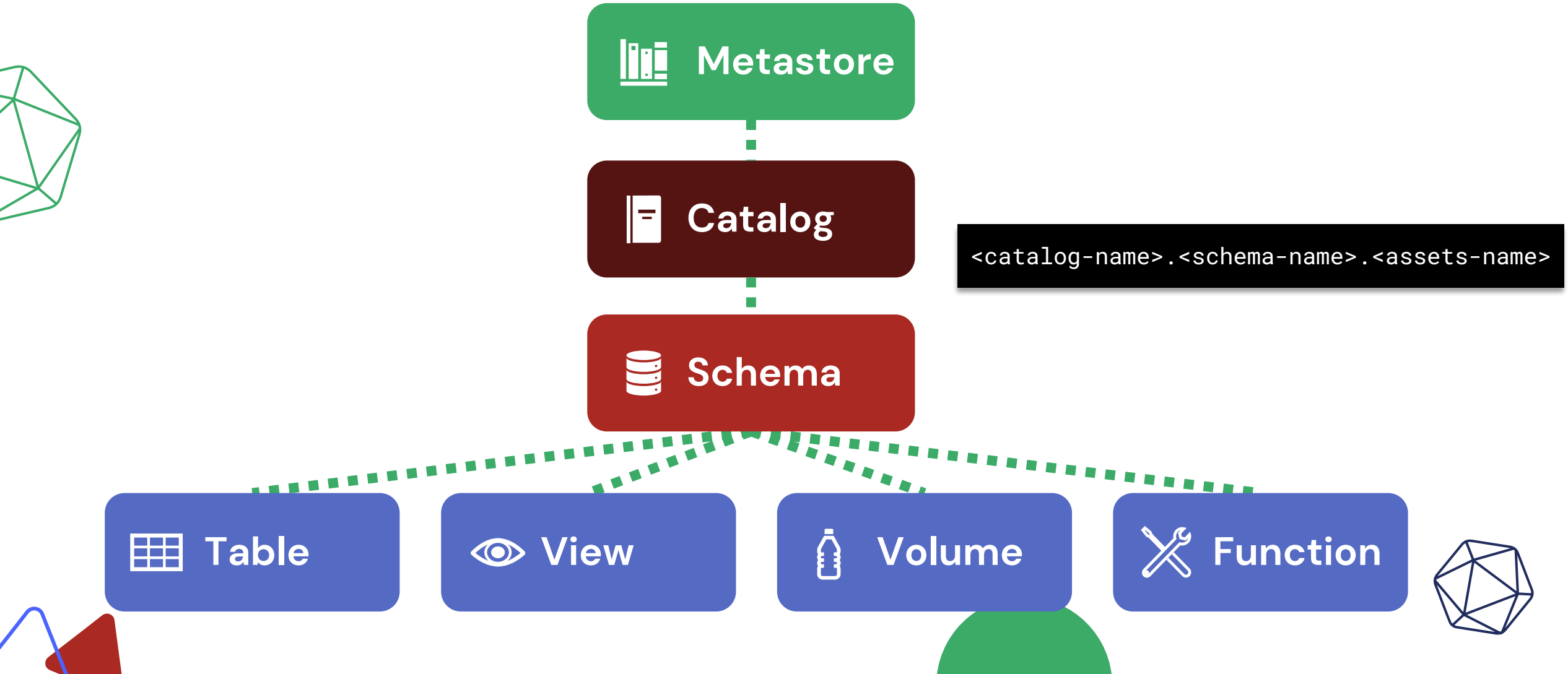


Databricks Unity Catalog



Unity Catalog Object Model

Unity Catalog provides centralised access control, auditing, lineage, and data discovery capabilities across Azure Databricks workspaces.




```
<catalog-name>.<schema-name>.<assets-name>
```

Unity Catalog - Metastore

Metastore

Metastore - A central repository for metadata about data assets.

Top-Level Container: The metastore is the top-level container for data in Unity Catalog. It registers metadata about securable objects like tables, volumes, external locations, and shares

Three-Level Namespace: Each metastore exposes a three-level namespace (catalog, schema, table) for organizing

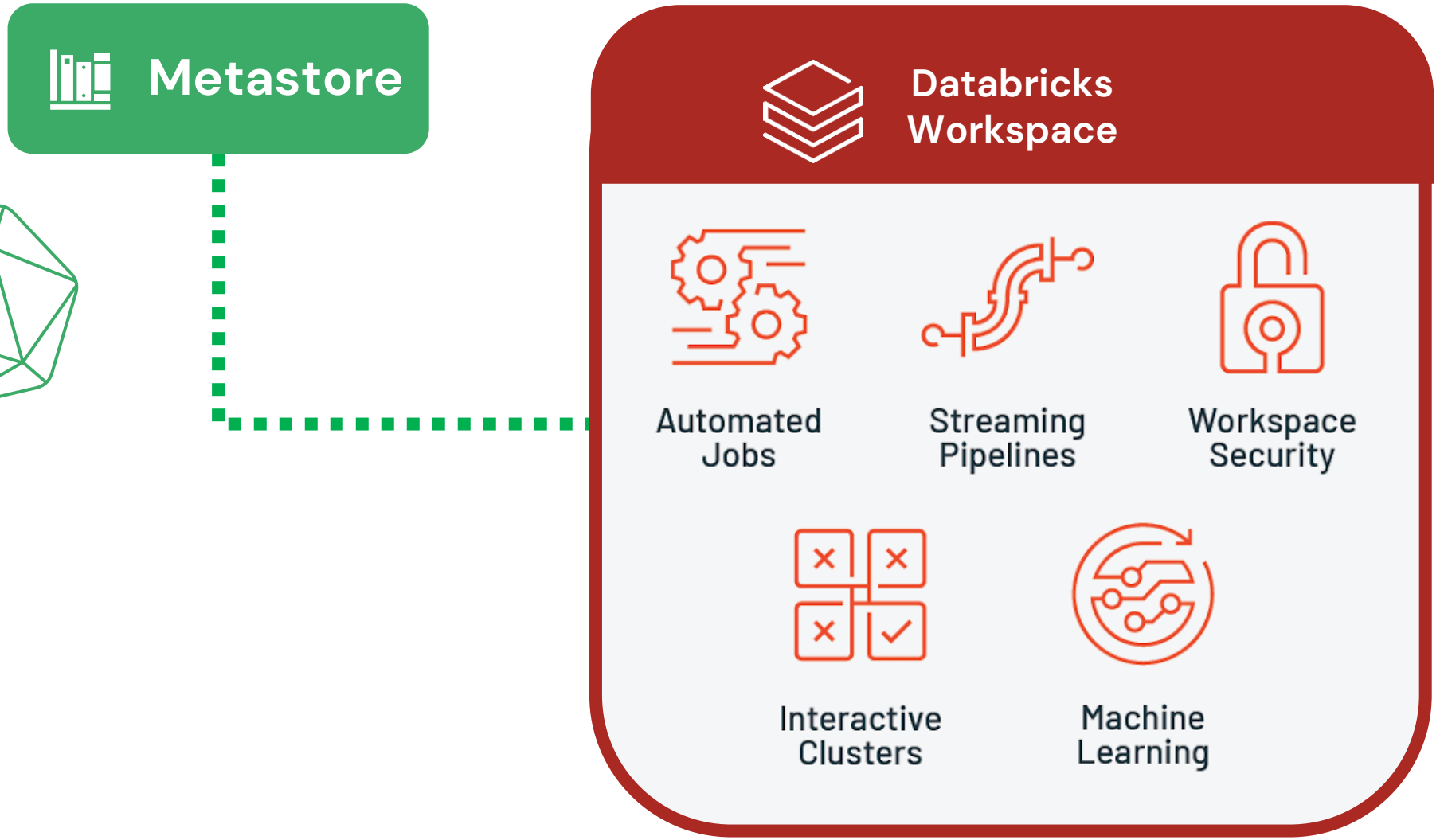
Regional Requirement: You must have one metastore for each region in which your organization operates

Permissions Management: Metastores manage permissions that govern access to securable objects.

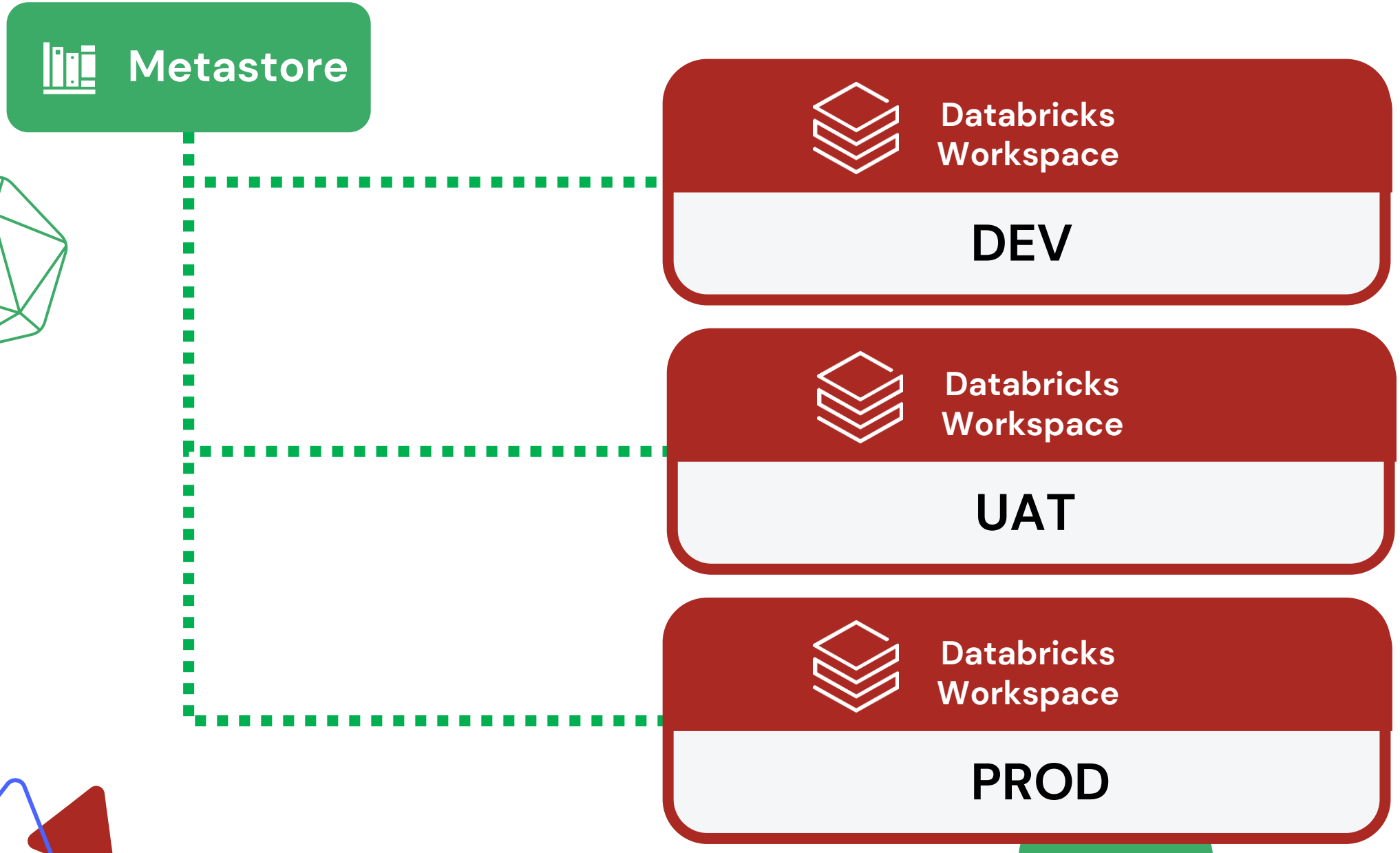
Workspace Attachment: Users must be on a workspace that is attached to a metastore in their region to work with Unity Catalog.

Managed Storage: Optionally, you can create metastore-level managed storage for managed tables and volumes.

Unity Catalog - Workspaces

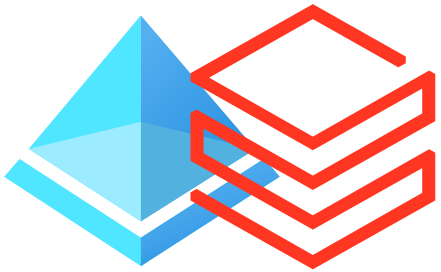


Unity Catalog - Workspaces



Account Console

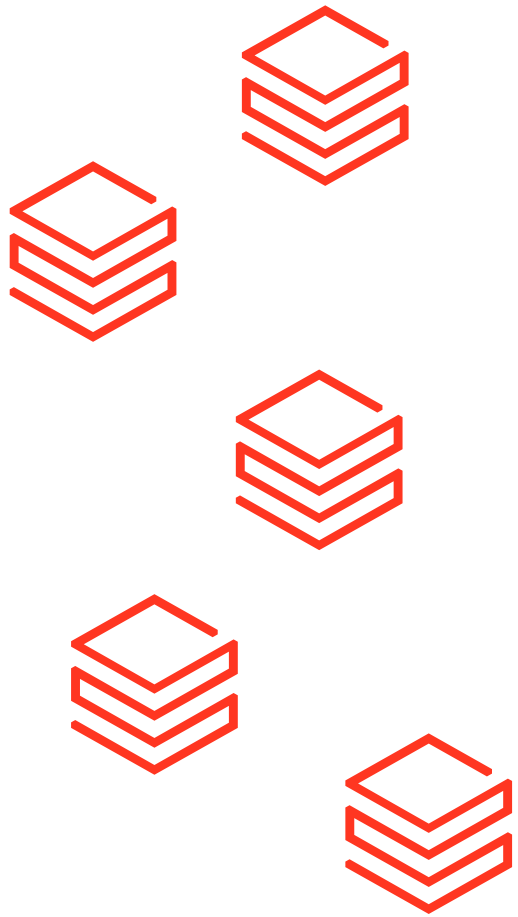
Account Console



Central Databricks Account at the AD Tenant level

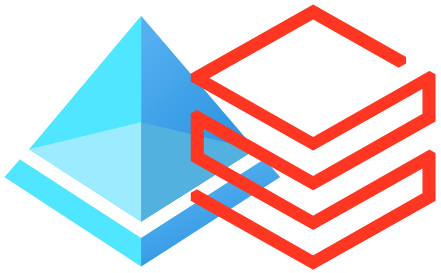
Unity Catalog

- Data Access
- Data Documentation
- Data Lineage

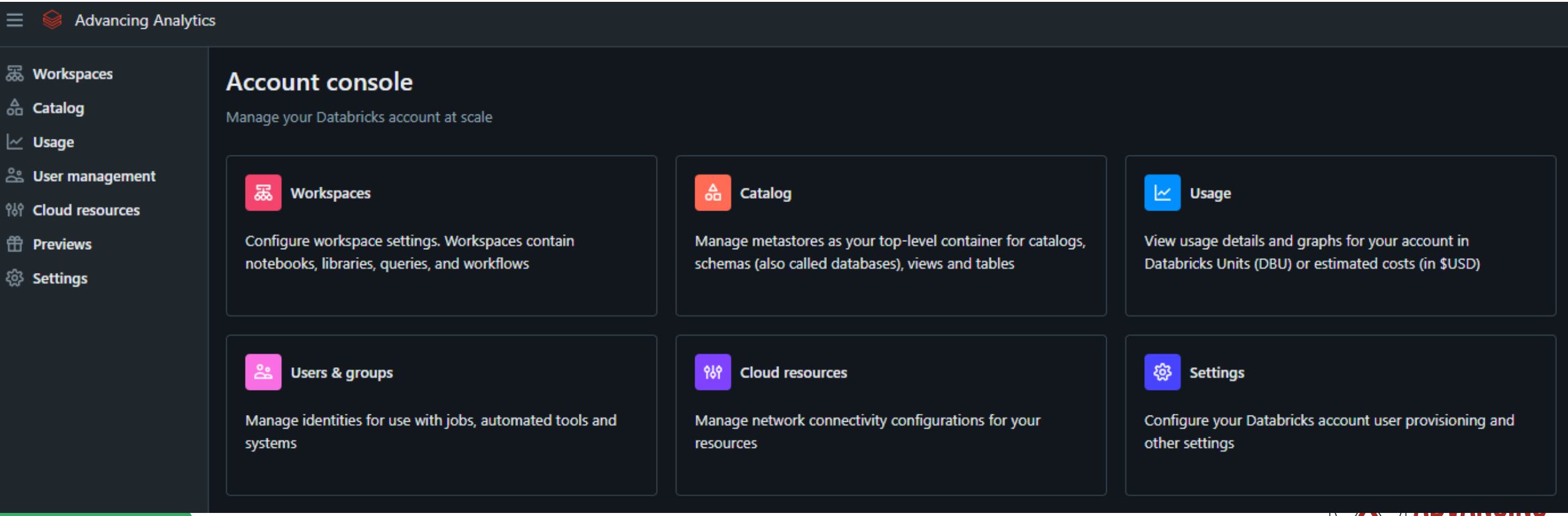


ADVANCING ANALYTICS

Account Console



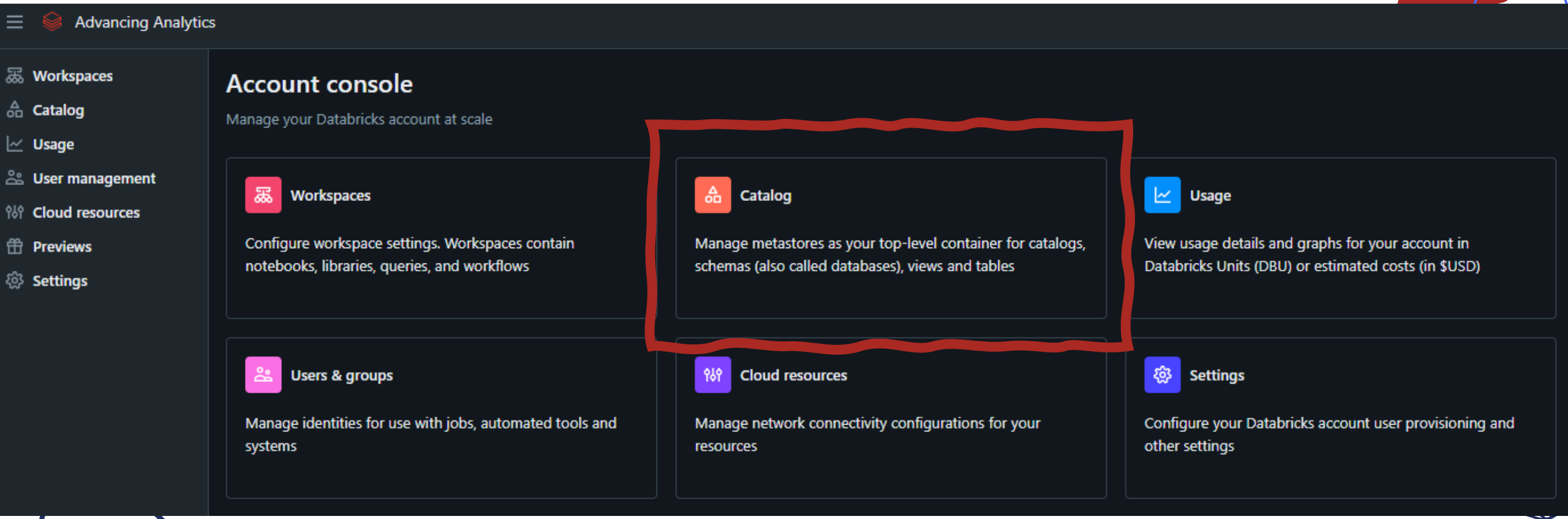
- Workspace Logging & Management
- User / Service Principal / Group Management
- Metastore Configuration



Create a Metastore

Create a Metastore

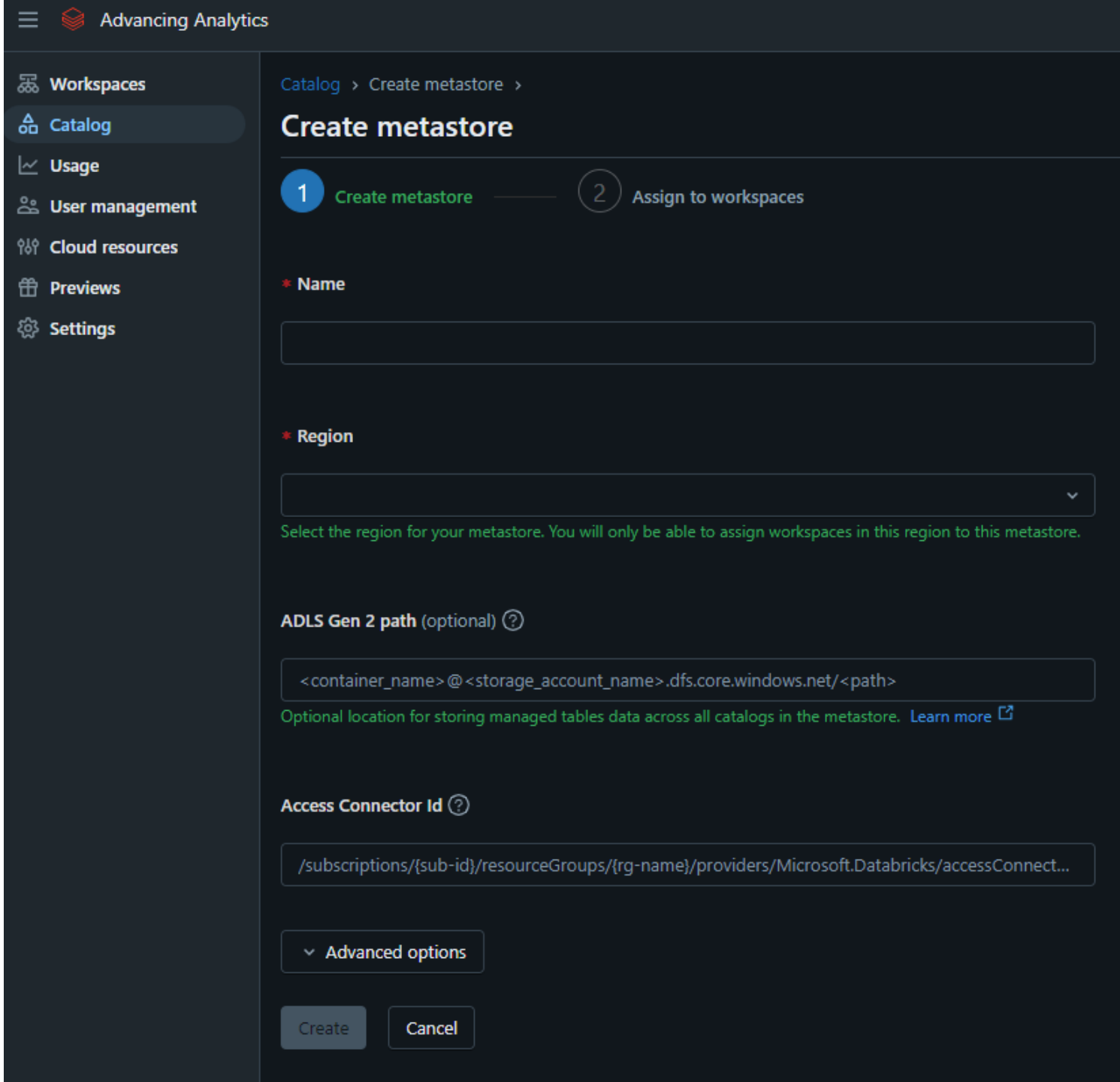
Click ‘Catalog’ to View/Create a Metastore and assign a storage account (ADLS)



ADVANCING ANALYTICS

Create a Metastore

- Select a **Name** for you metastore
- Choose a **Region** for your metastore
  Remember there is one per region.
- Select the **full path** for your chosen storage account (ADLS) <container>@<storage_account>....
- *Assign the **Resource ID for your Access Connector** to authenticate access to your metastore storage account*



Databricks Access Connector

Databricks Access Connector

The **Access Connector for Azure Databricks** allows you to connect **managed identities** to an Azure Databricks account for accessing data registered in Unity Catalog.

Primary Uses:

- Connecting to the metastore's storage accounts for managed tables.
- Connecting to external storage accounts for file-based access or external tables.

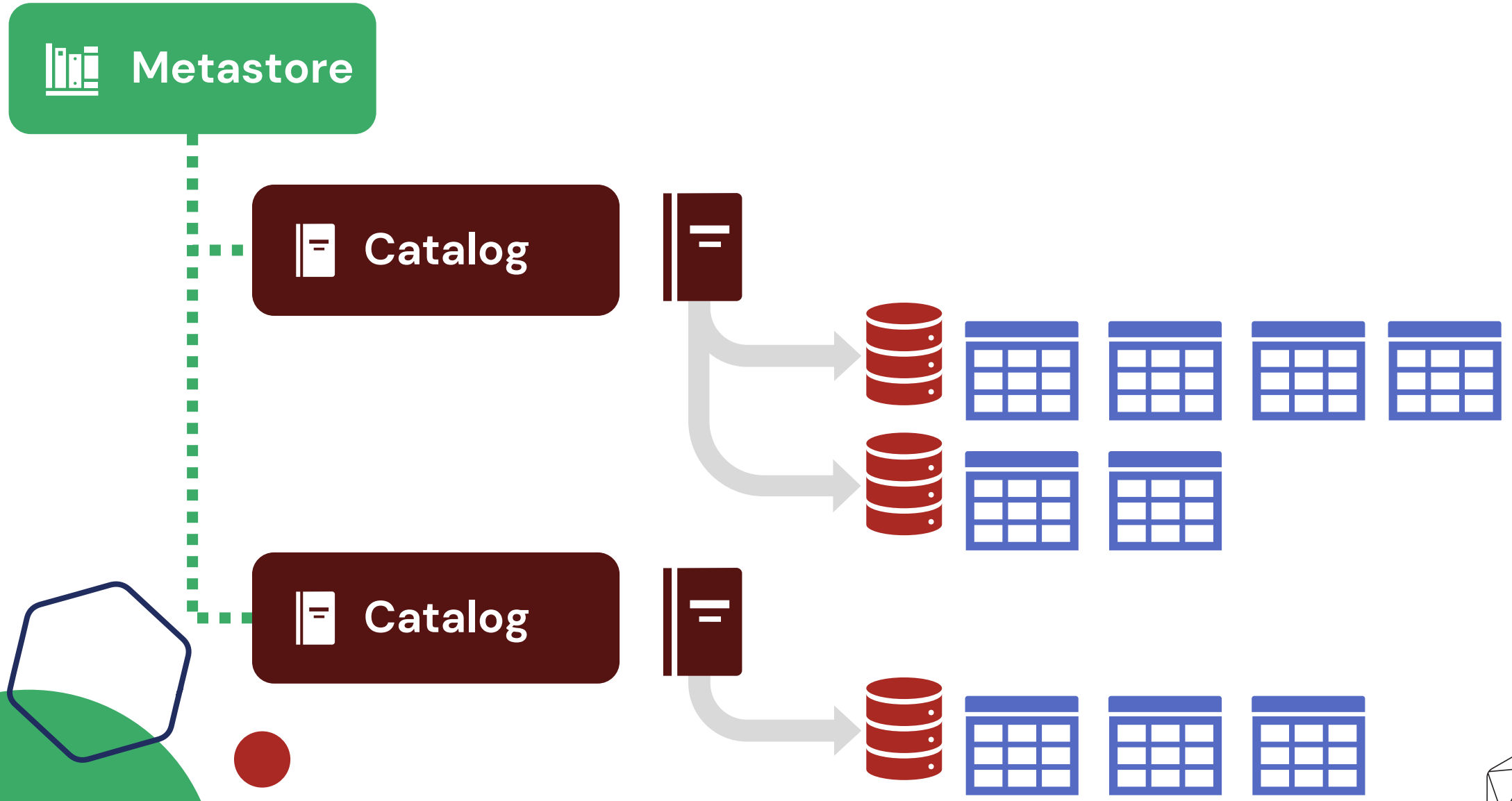
Benefits:

- Access to Azure Data Lake Storage Gen2 accounts protected by a storage firewall.
- No need to maintain credentials or rotate secrets.

ADVANCING ANALYTICS

Unity Catalog Objects

Unity Catalog - Catalog



ADVANCING ANALYTICS

Unity Catalog - Catalog Syntax

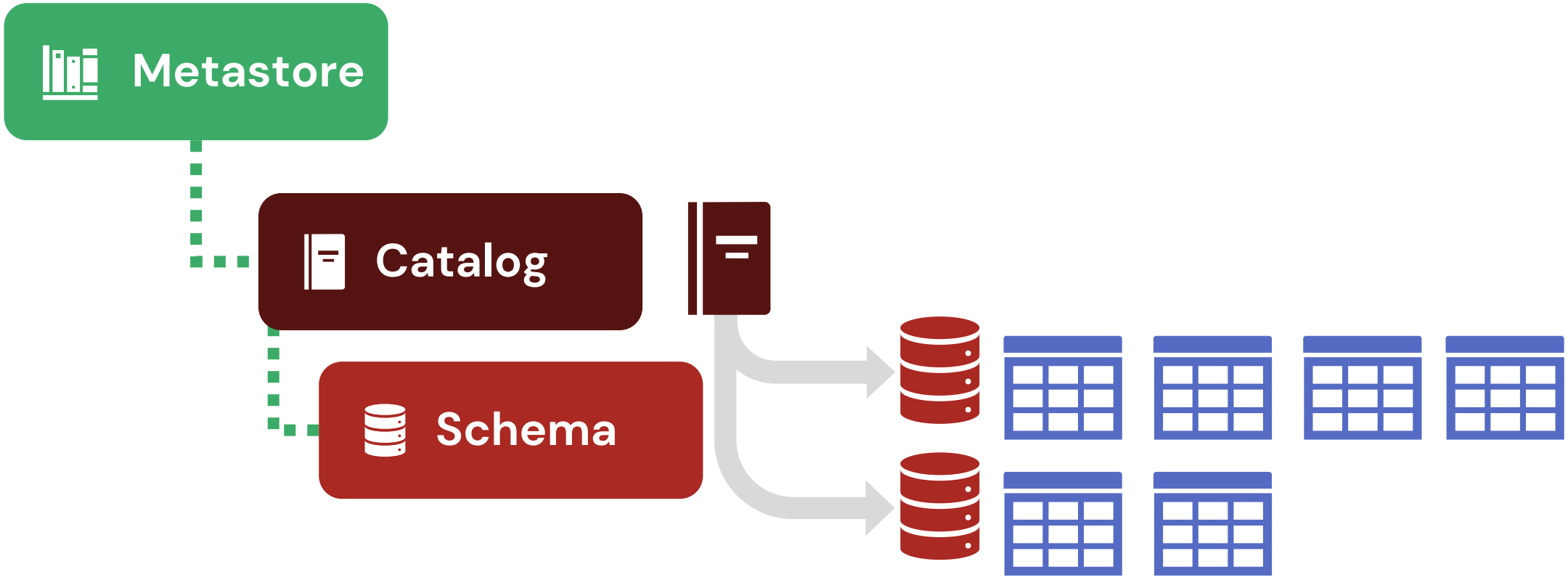

```
-- SQL Commands
-- Create a new Catalog Object
CREATE CATALOG SalesMart COMMENT 'Analytics Mart for Sales'

-- Drop a Catalog Object
DROP CATALOG IF EXISTS SalesMart
```

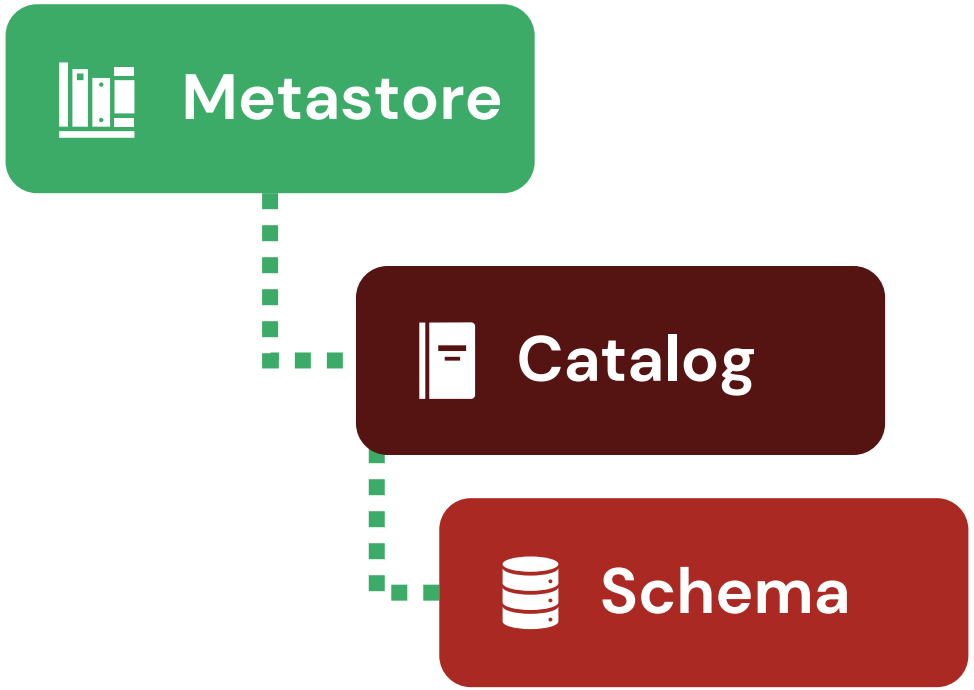
Catalogs are created and managed as SQL objects – the emphasis all throughout Unity Catalog is using SQL to manage security!

ADVANCING ANALYTICS

Unity Catalog – Schema



Unity Catalog - Schema Syntax



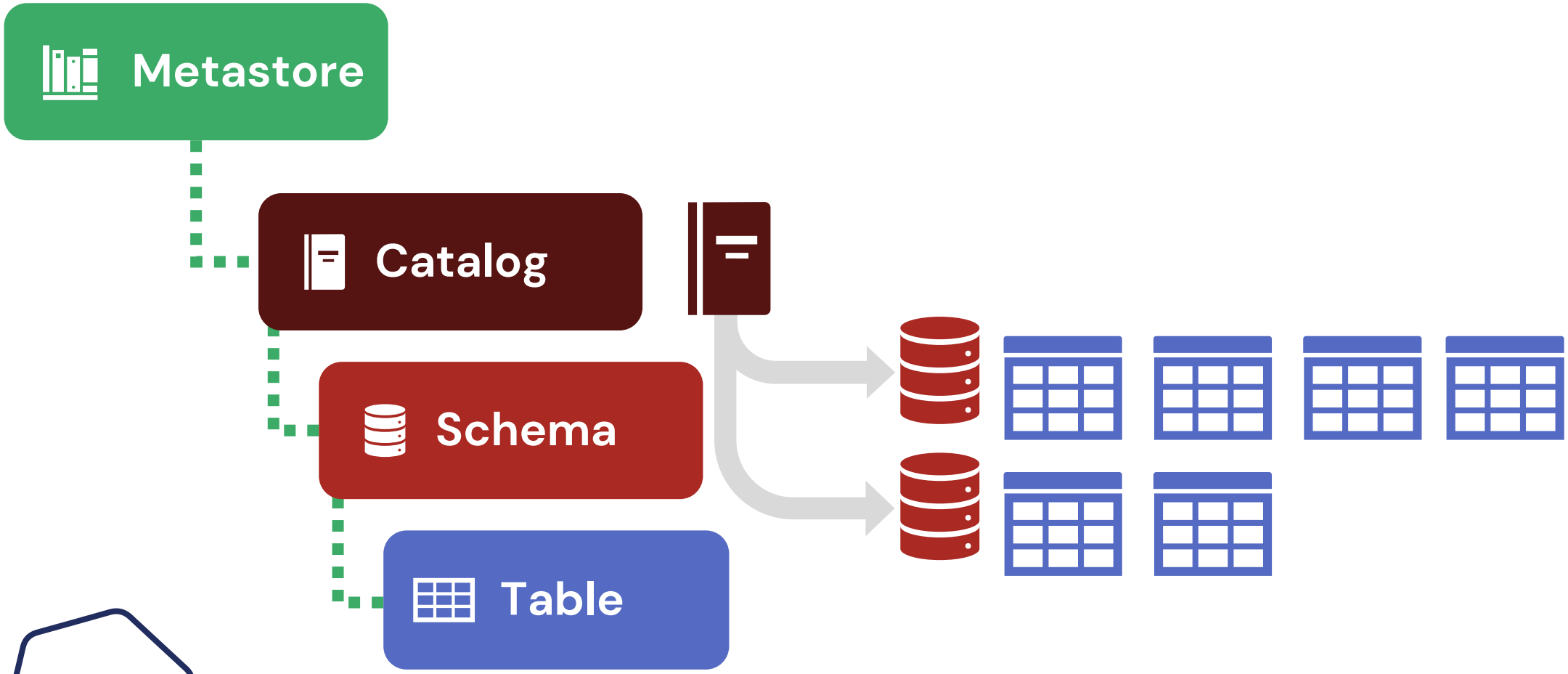

```
-- Create a new Schema in the current Catalog
USE CATALOG SalesMart;
CREATE SCHEMA ItemSales COMMENT 'Star Schema for the ItemSales
Model'

-- Explicit Two-Part Creation
CREATE SCHEMA SalesMart.ItemSales

-- Drop Schema object
DROP SCHEMA SalesMart.ItemSales
```

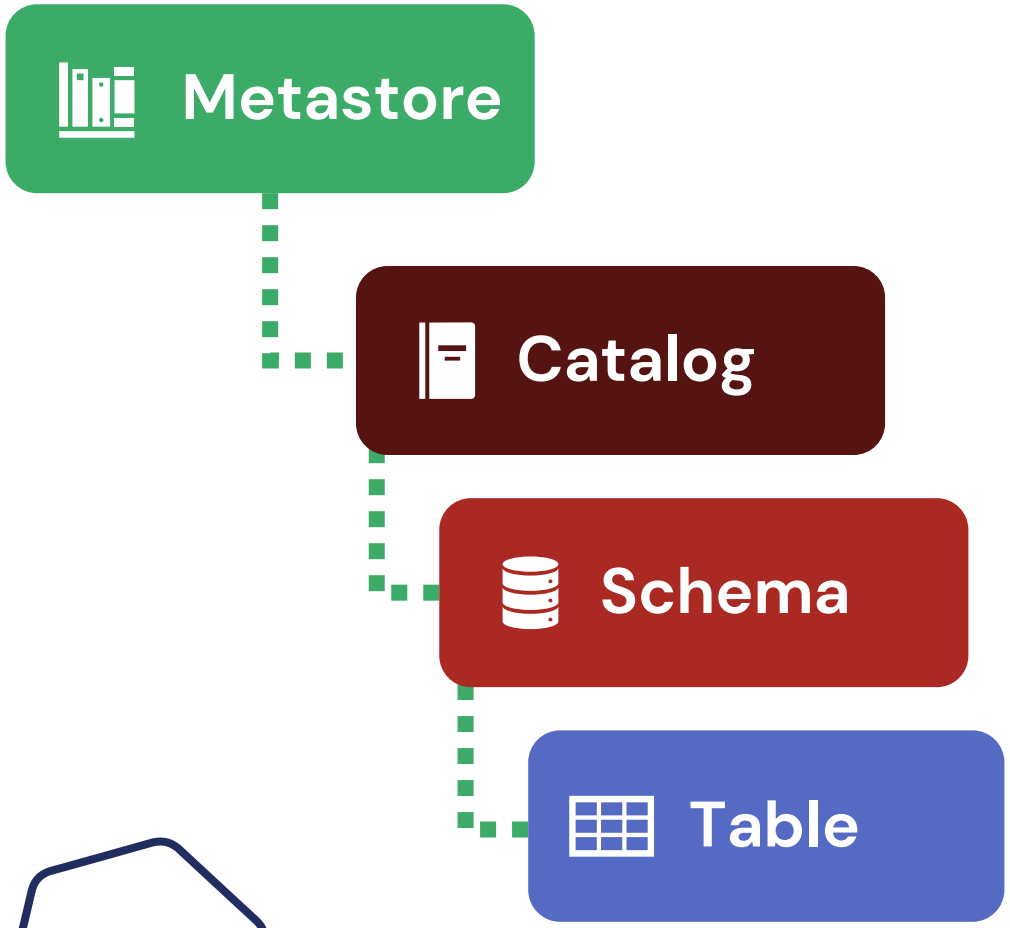
ADVANCING ANALYTICS

Unity Catalog – Table



ADVANCING ANALYTICS

Unity Catalog - Table Syntax




```
-- Create a Managed Table Object
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
        (ID int, ProductID int, Amount decimal)
```

This creates a ***Managed Table*** – that's a specific type of table in Spark. We'll come on to that!

ADVANCING ANALYTICS

Managed VS External

Managed VS Unmanaged Tables

Managed Table

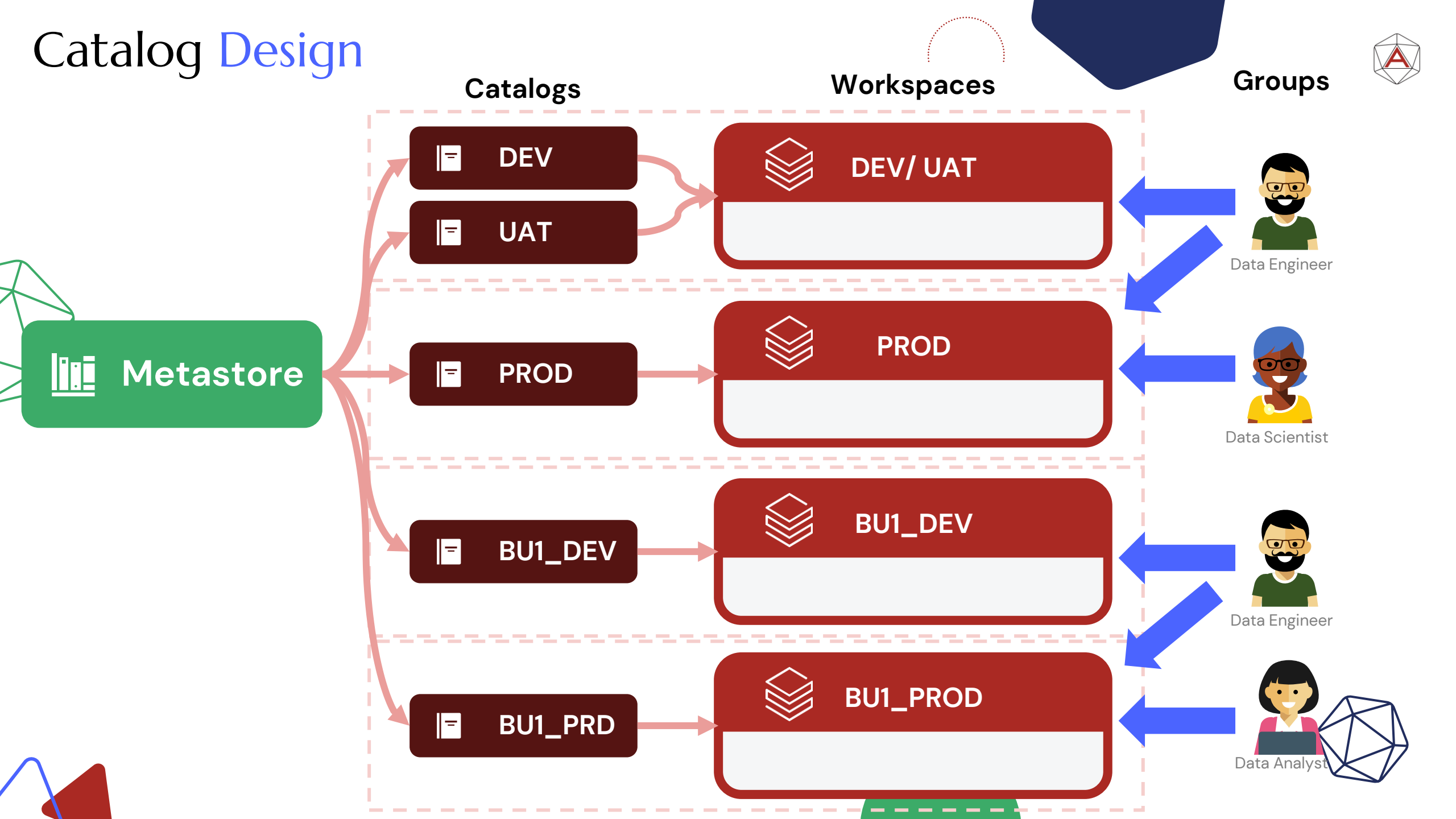
- Stored in managed default storage
- Dropping a table deletes data files
- Only supports Delta format
- Data Linked to SQL Object
- No Lake Folder Control
- No Storage Credentials Required

External Table

- Stored in user-defined/BYO storage
- Dropping a table does not delete data files
- Supports multiple data formats Data & SQL Object separated
- Full Lake Folder Flexibility
- Requires Storage Credentials

ADVANCING ANALYTICS

Catalog Design



Demo: Unity Catalog Objects

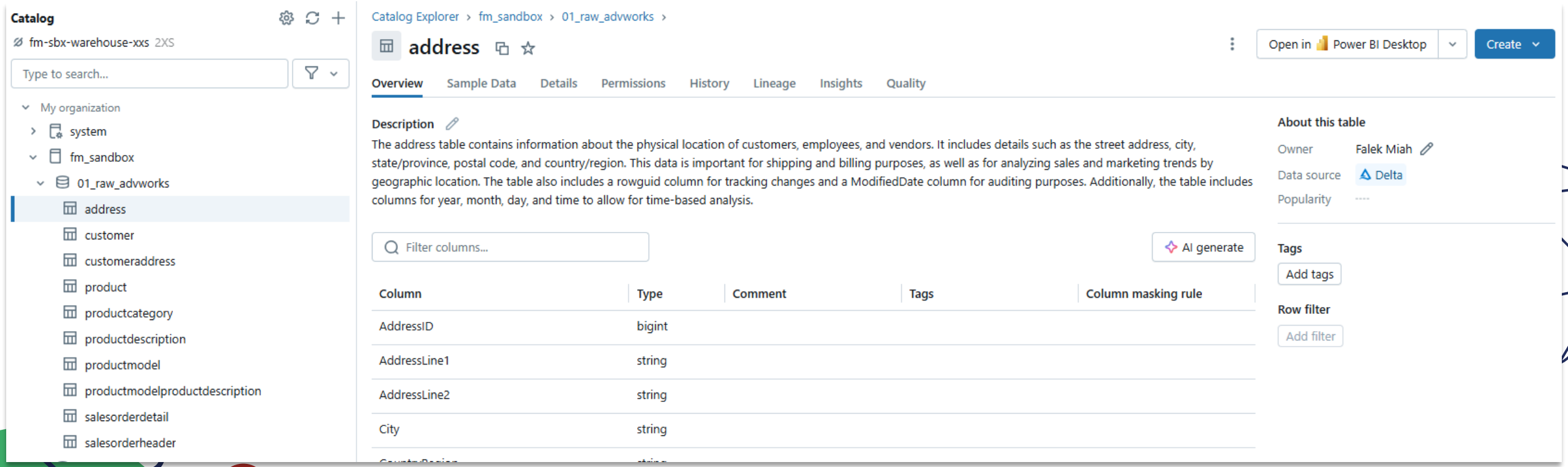
- Create a Catalog
- Create a Schema
- Create Tables & Views

Unity Catalog Features

Catalog Explorer

The Catalog Explorer in Databricks used to be fairly limited – you needed an active cluster to query it!

Unity Catalog brings a permanently active Catalog browser with much improved UI

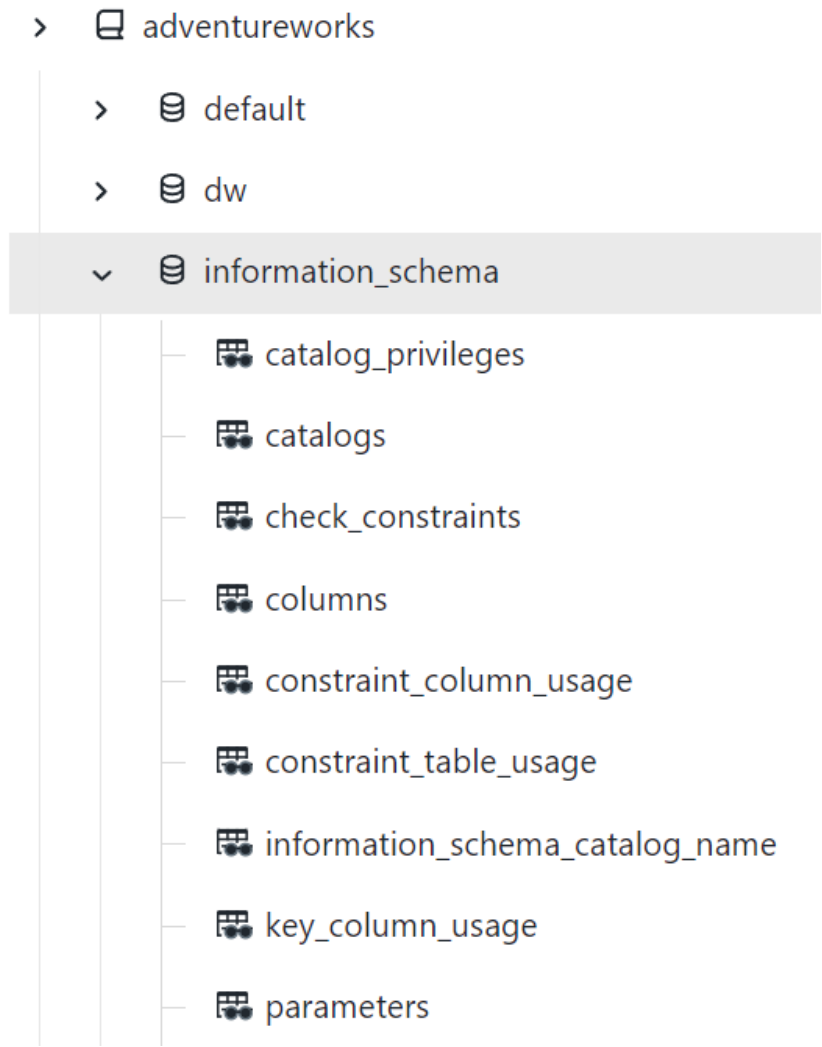


ADVANCING ANALYTICS

Information Schema

Another nice addition is the "information_schema" that is automatically added to every Catalog

This can be queried to automate common tasks, just as we would with SQL Server sys DMVs

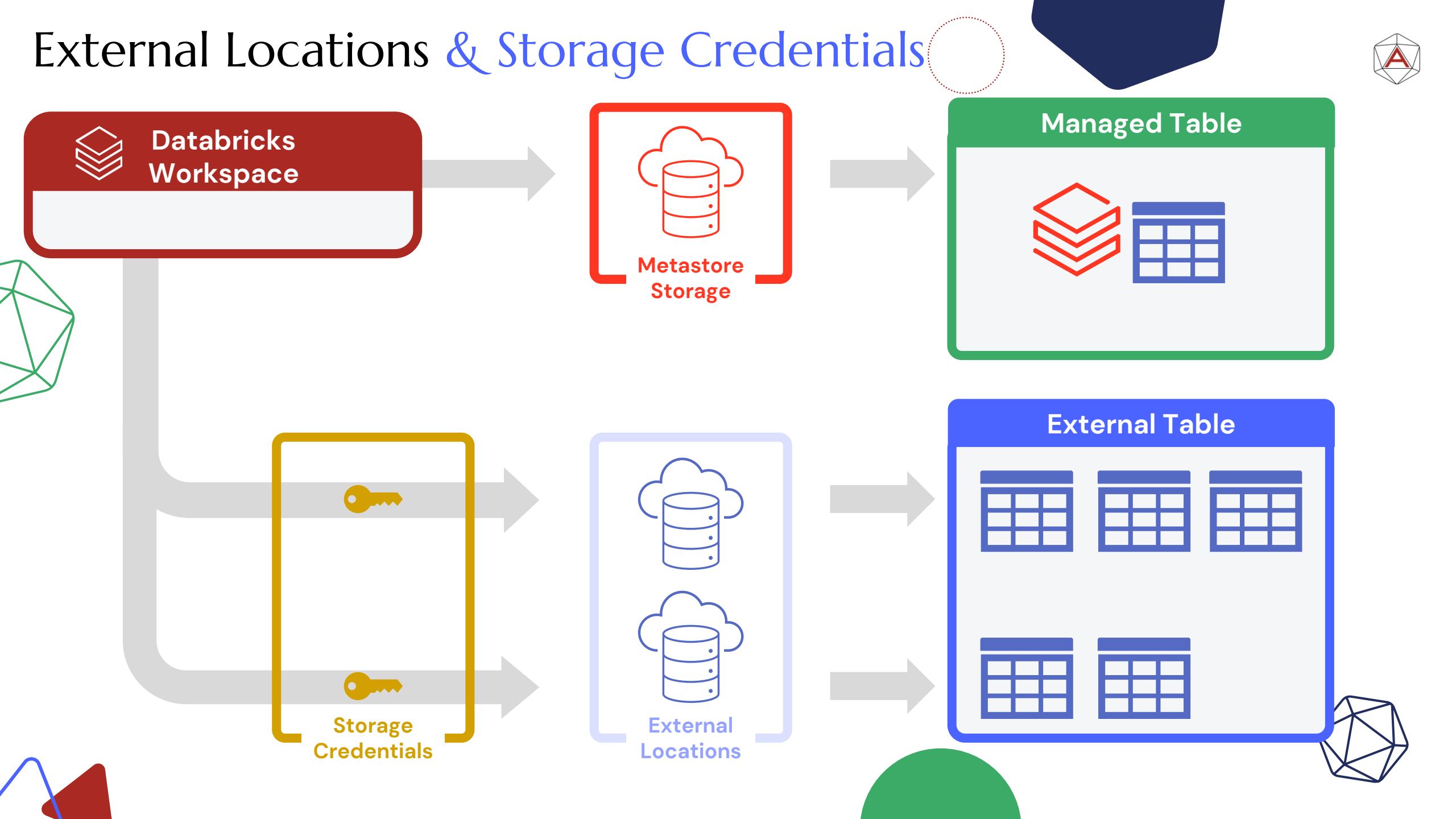


ADVANCING ANALYTICS

Demo: Catalog Explorer

External Locations & Storage Credentials

External Locations & Storage Credentials



External Locations & Storage Credentials

- Required to enable UC access to cloud data storage.
- Any privileges granted on storage credential apply to entire container.

Storage Credentials

- Supports fine-grained access control to folders and files.
- Combines storage credential and cloud storage path.
- Multiple External Locations can use the same Storage Credential.

External Locations

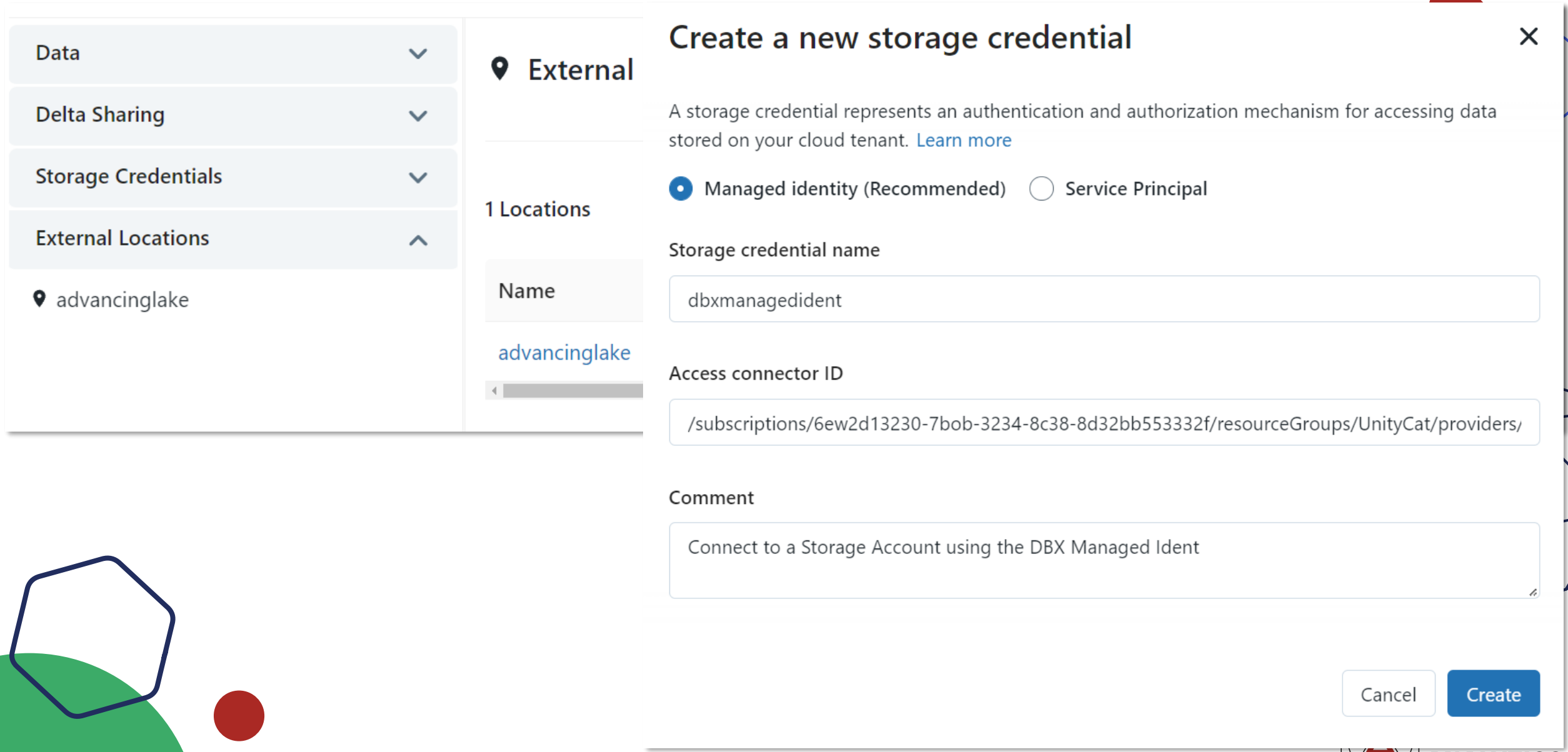
Storage Credential - Requirements

- Storage Credential represents an authentication and authorization mechanism for accessing data stored on your cloud tenant.

Storage Credentials

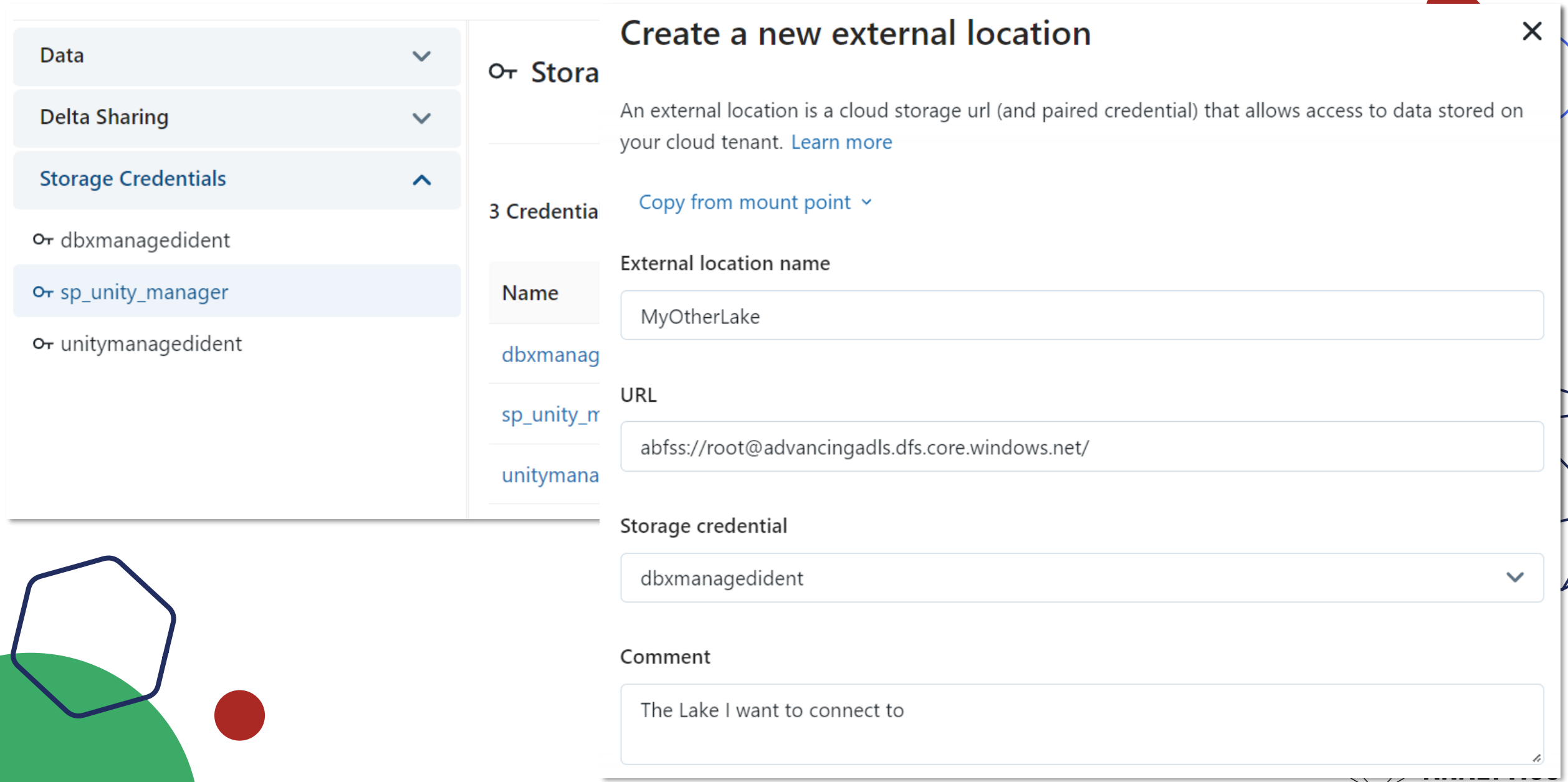
Cloud Provider	Storage Service	Credential for Authorisation
AWS	Amazon S3	IAM Role
Azure	Azure Data Lake Storage Gen 2	Managed Identity (via Access Connector)
Google Cloud	Google Cloud Storage (GCS)	Service Account

Storage Credential – Setup



ANALYTICS

External Locations – Setup



External Locations - Syntax

Managed Table

```
-- Create Managed Table
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
(ID int, ProductID int, Amount decimal)
```

External Table

```
-- Create External Table
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
        (ID int, ProductID int, Amount decimal)
LOCATION “abfss://root@mylake. dfs.core.windows.net/salesitems/factitemsales”


-- Create External Table with Explicit Credential
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
        (ID int, ProductID int, Amount decimal)
LOCATION “abfss://root@mylake. dfs.core.windows.net/salesitems/factitemsales”
WITH (CREDENTIAL dbxmanagedident);
```

Lineage

Unity Catalog - Lineage Tracking

Once activated, the Lineage feature will automatically trawl all Spark jobs to determine whether data moved between registered Unity Catalog table.

This lineage is automatically added to the Table's Catalog entry

Manage User Permissions

Unity Catalog - Privileges

Securable	Privileges
Metastore	CREATE CATALOG CREATE EXTERNAL LOCATION CREATE RECIPIENT CREATE SHARE
Catalog	ALL PRIVILEGES BROWSE CREATE SCHEMA USE CATALOG
Schema	ALL PRIVILEGES CREATE FUNCTION CREATE TABLE CREATE VIEW USE SCHEMA
Table / View	ALL PRIVILEGES SELECT MANAGE MODIFY

Securable inherit downwards

Granting "**ALL PRIVILEGES**" on a schema applies to all child table/view objects within that schema

```
-- Allow Data Scientists to Read All Tables
GRANT SELECT ON TABLE SalesMart.ItemSales.FactItemSales
TO `DataScientists`;
```

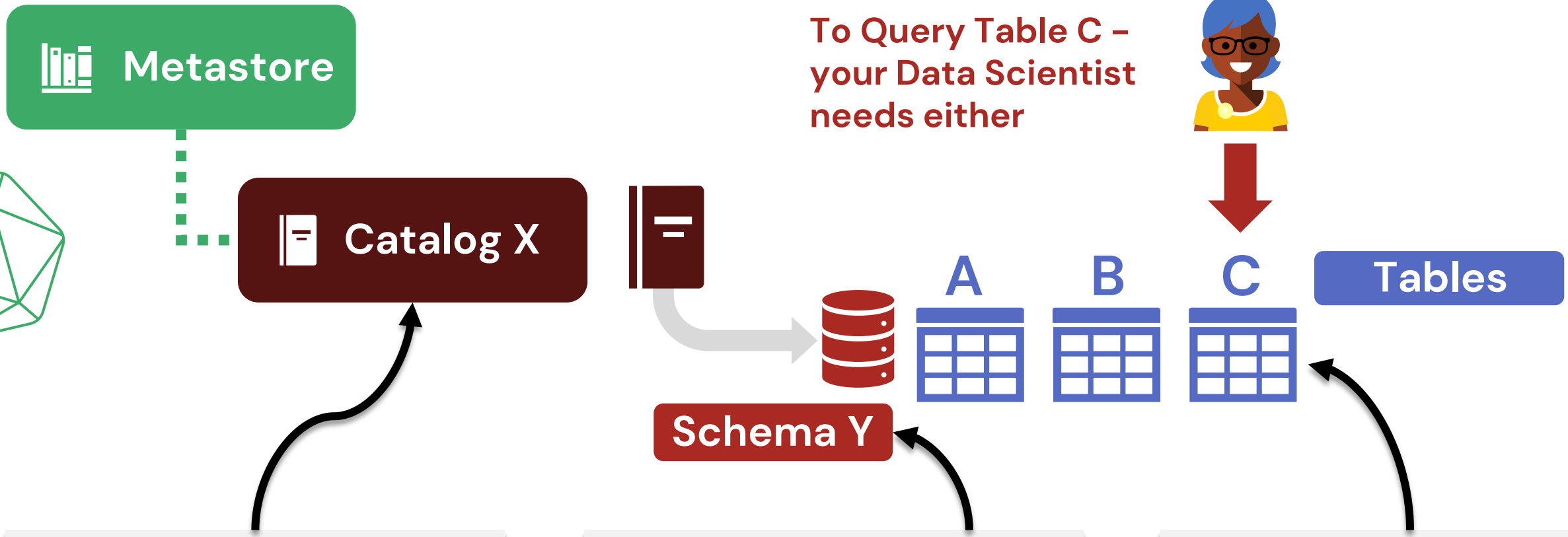
Unity Catalog - Privileges

Securable	Privileges
Metastore	CREATE CATALOG CREATE EXTERNAL LOCATION CREATE RECIPIENT CREATE SHARE
Catalog	ALL PRIVILEGES BROWSE CREATE SCHEMA USE CATALOG
Schema	ALL PRIVILEGES CREATE FUNCTION CREATE TABLE CREATE VIEW USE SCHEMA
Table / View	ALL PRIVILEGES SELECT MANAGE MODIFY

You cannot access child objects unless you have “USAGE” on the parent object

To SELECT from a table – you need at least “USE SCHEMA” and “USE CATALOG” on the parent items

Unity Catalog – Privileges Example



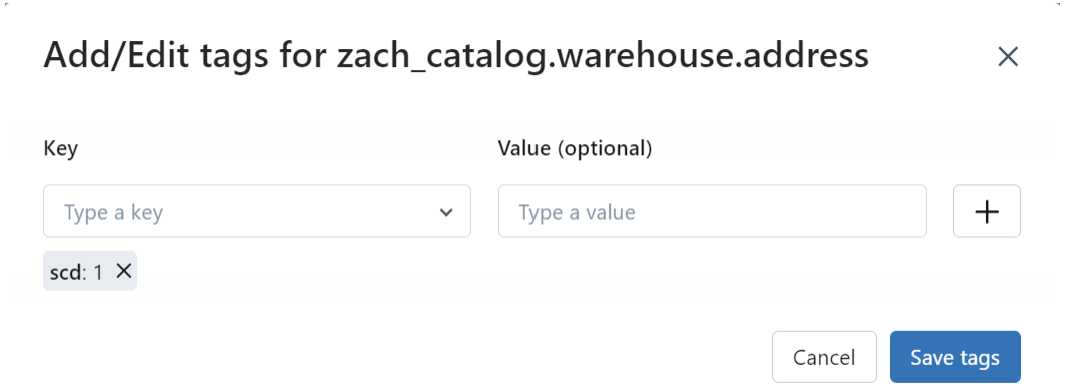
Demo: Managing Permissions

Data Discovery

Tagging

- Helps organize data objects
- Applied to tables, views, schemas, and columns
- Automated policy enforcement
- Enhances searchability
- Flows into the information schema.
- Constraints

- The following privileges are required:
  - APPLY TAG
  - USE SCHEMA
  - USE CATALOG



Comments

- Descriptive comments to data objects
- Insights into dataset usage, assumptions, and quality
- Improves knowledge sharing & collaboration
- Added to catalogs, schemas, tables, and columns
- View in Catalog Explorer or via SQL commands
- Maintains history of updates

```
COMMENT ON CATALOG catalog IS 'comment';

COMMENT ON SCHEMA schema IS 'comment';

COMMENT ON TABLE table IS 'comment';

COMMENT ON COLUMN table.c1 IS 'comment';
```

Column	Type	Comment	Tags
ProductKey	bigint	Add comment	Add tags
ProductId	int	Business Key for Product records.	
ProductName	string	Name of the product.	
ProductNumber	string	Unique product identification number.	
CurrentProductCategoryName	string	Product category.	
PreviousProductCategoryName	string	Product category.	
ProductCategoryName	string	Product category.	
CurrentParentProductCategoryName	string	Parent Product category.	
PreviousParentProductCategoryName	string	Parent Product category.	
ParentProductCategoryName	string	Parent Product category.	
ProductModelName	string	Product model.	

Demo: Data Discovery

LAB01 – Unity Catalog Objects

Creating and securing UC objects

UC Limitations

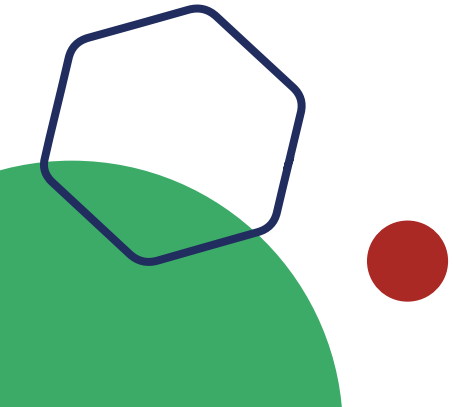
Unity Catalog Quotas

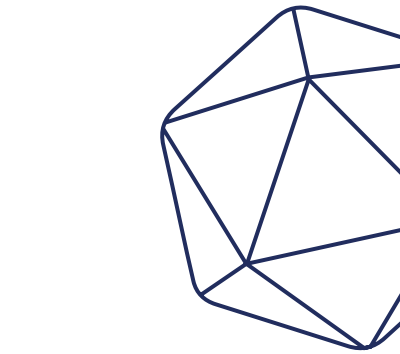
Object	Parent	Value
table	schema (database)	10000
schema	catalog	10000
catalog	metastore	1000
storage credentials	metastore	200
external locations	metastore	500
functions	schema	10000

Recap

Recap

- Hierarchical object structuring similar to SQL Databases
- Granular data access and security management
- Exploring data with UI and through information_schema tables
- External locations for managing data in ADLS
- Lineage tracking functionality
- Comments and tagging supports discoverability and governance





ADVANCING ANALYTICS