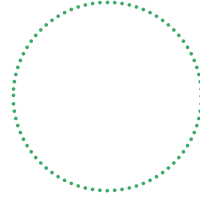


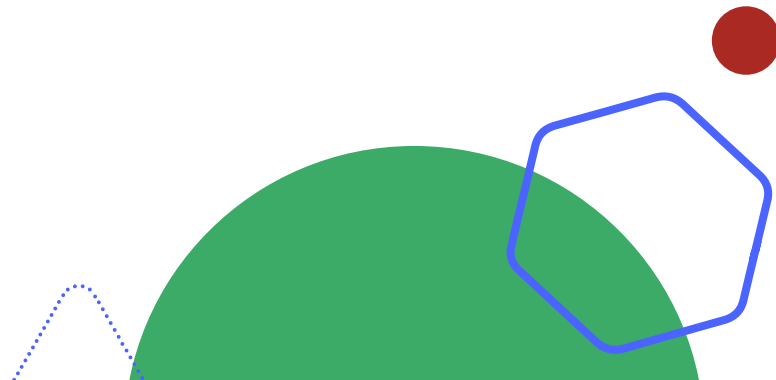


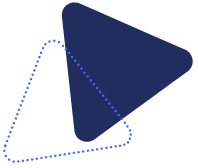
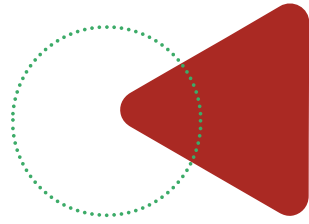
**ADVANCING
ANALYTICS**



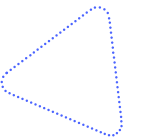
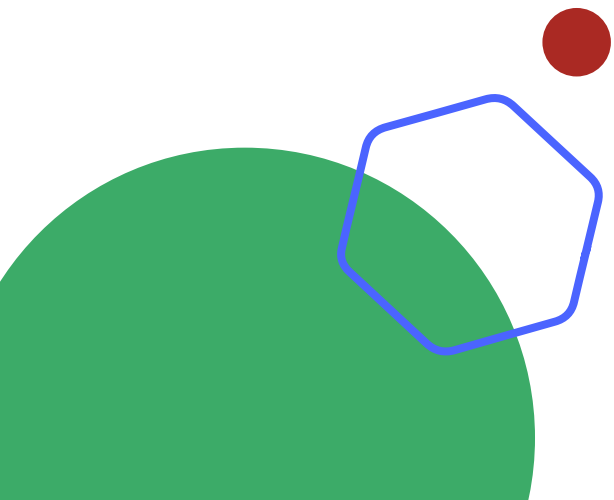
Unity Catalog

Databricks Governance



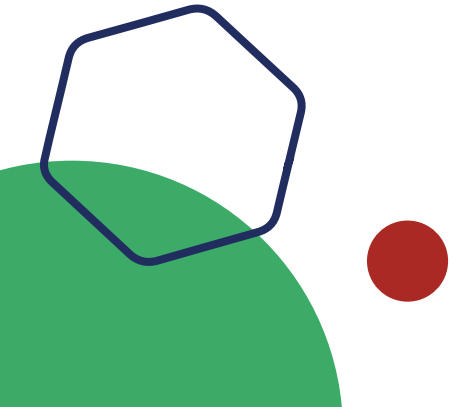
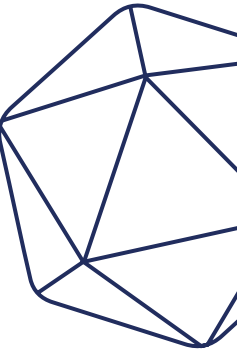
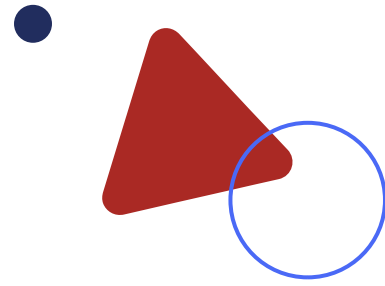


What Governance Problem?



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?

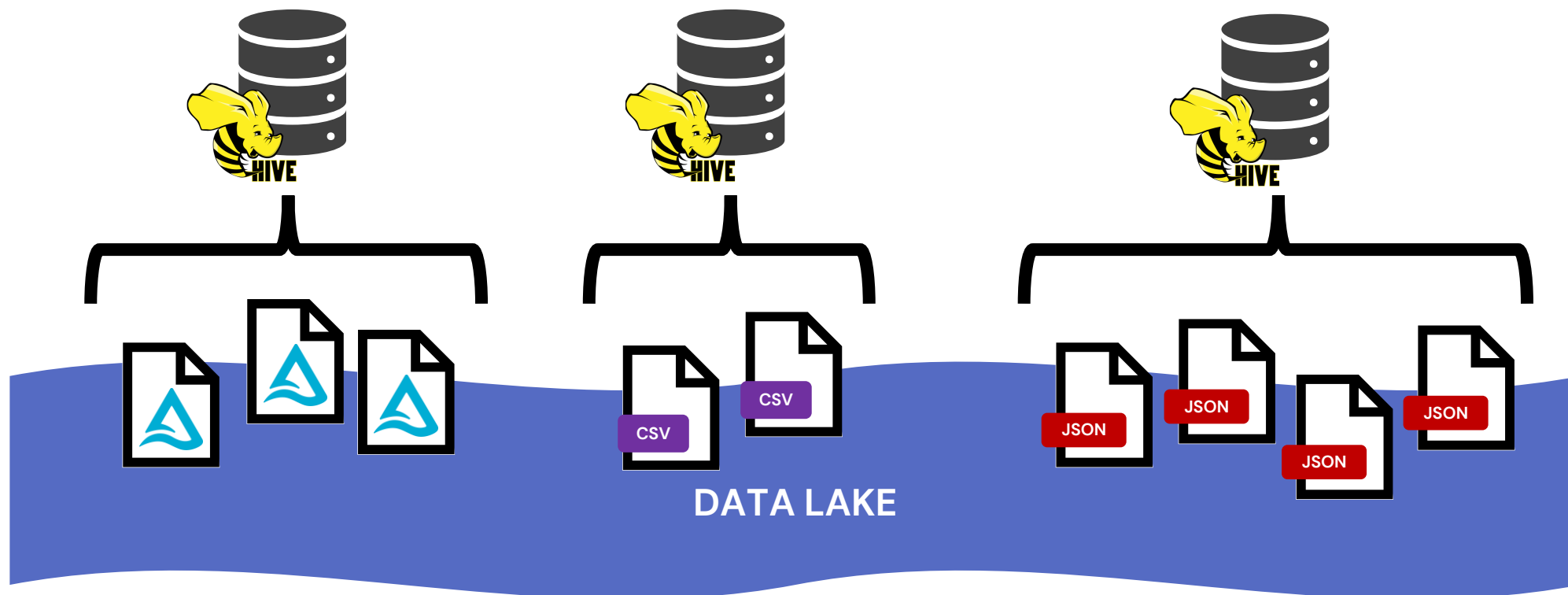


Legacy Databricks Metastore

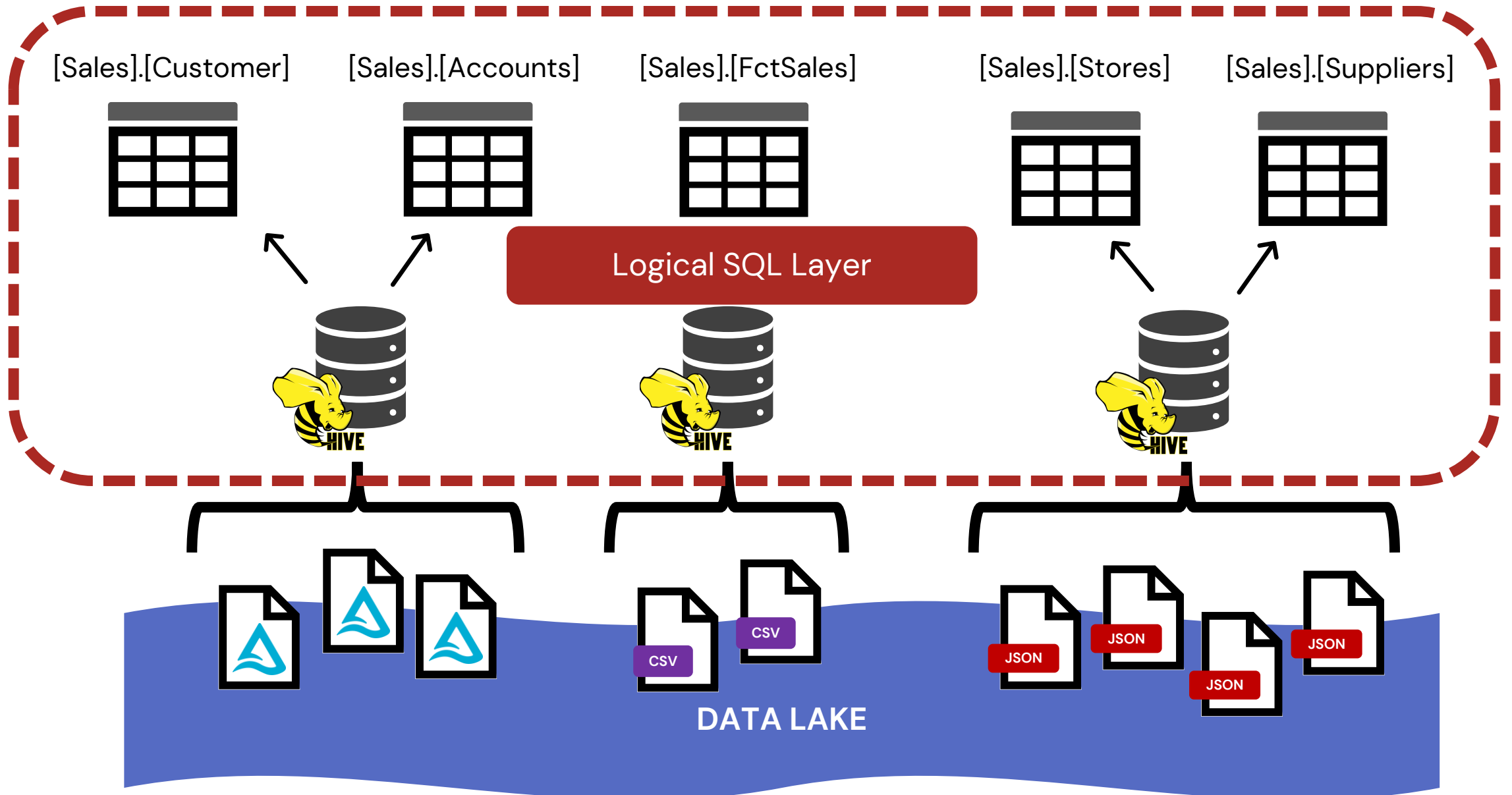


The Hive Metastore (Apache Hive) is a data warehouse system built on top of Apache Hadoop. It enables users to read, write, and manage large datasets stored in distributed storage using SQL.

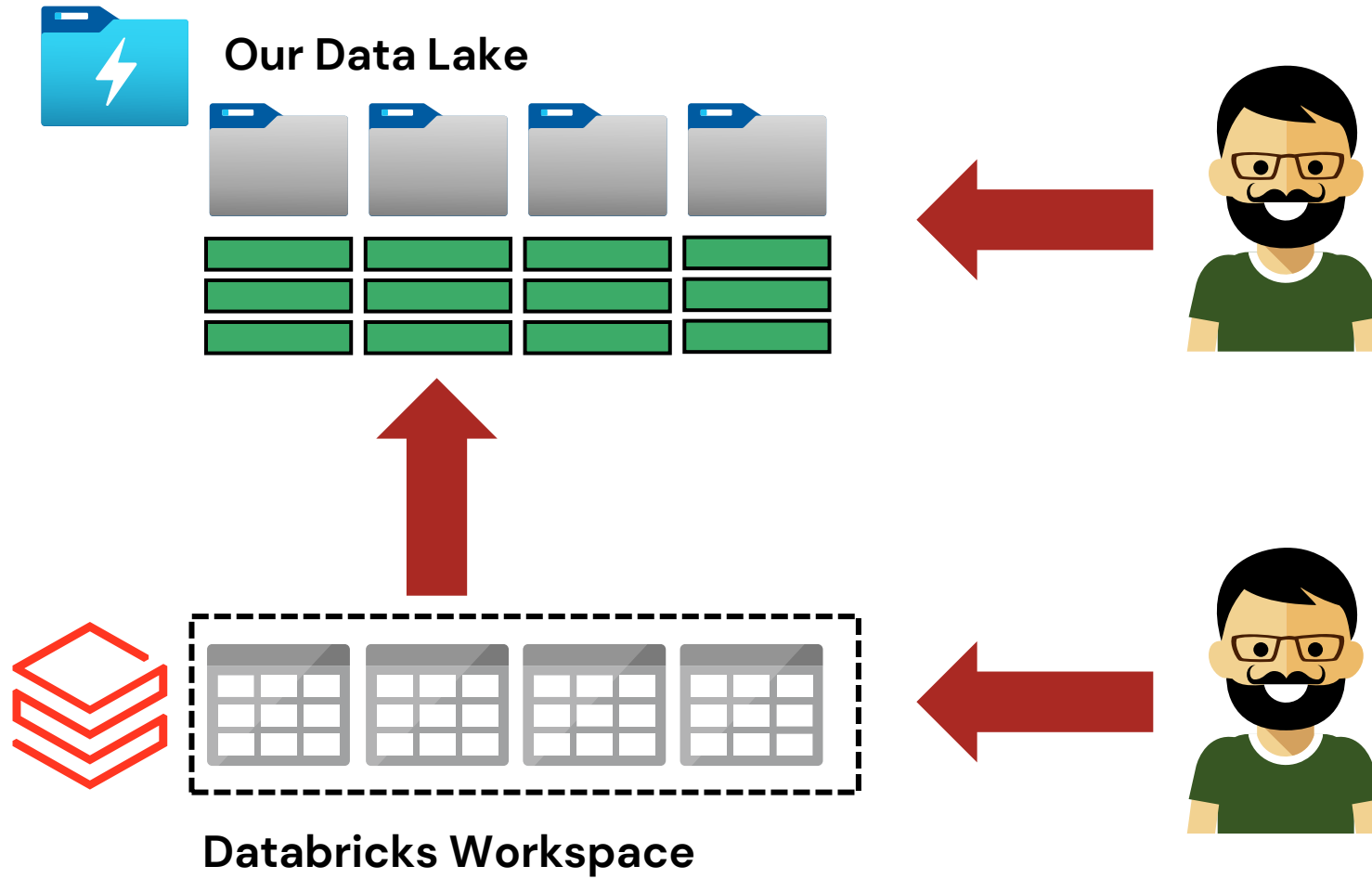
The Hive metastore in Databricks was used to manage and store metadata for entities like databases, tables, and columns.



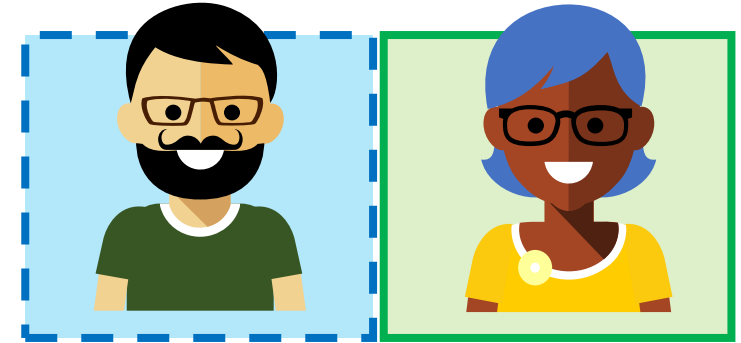
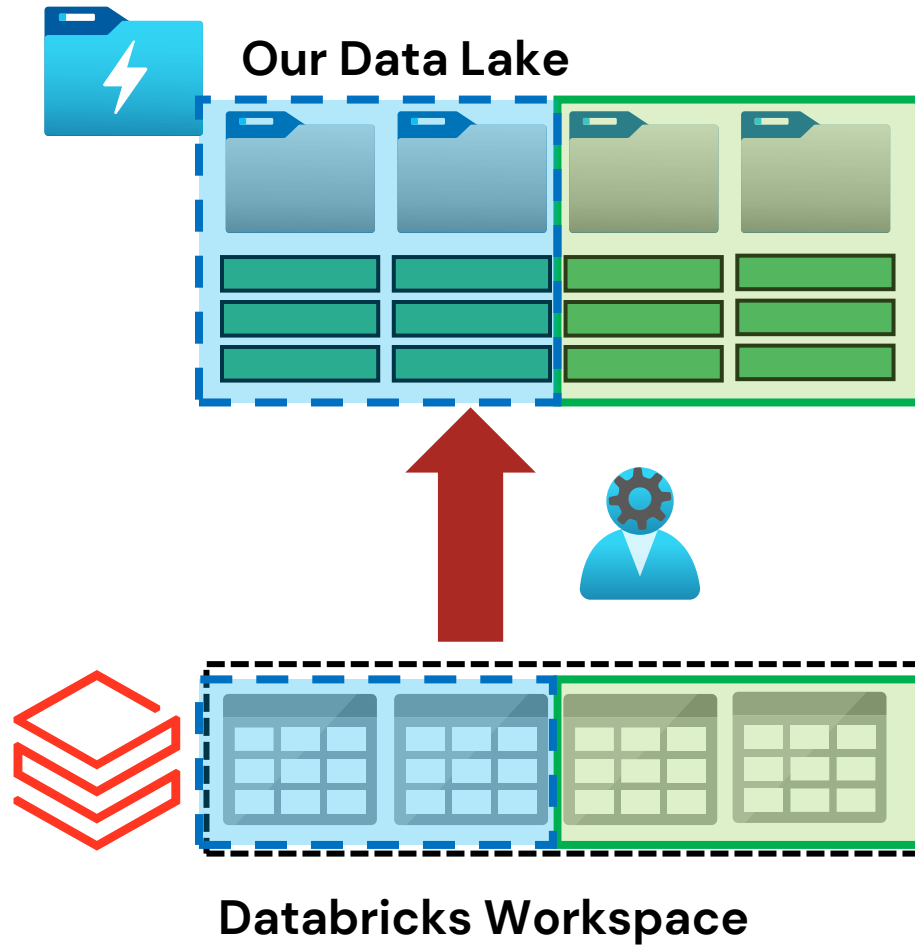
Legacy Databricks Metastore



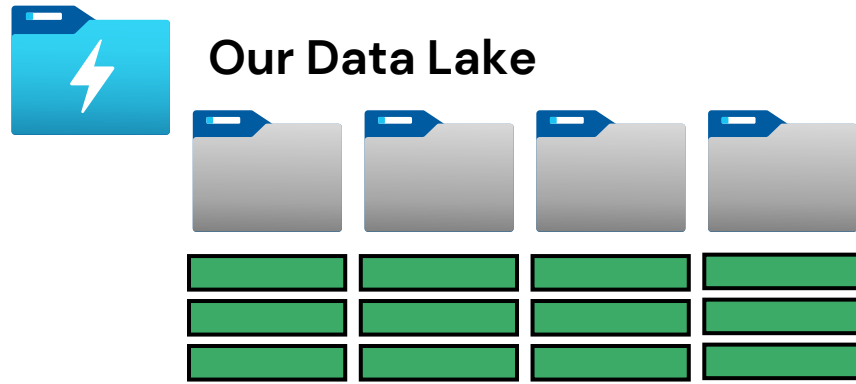
The Problem



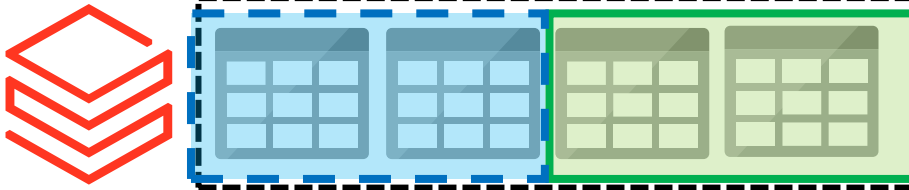
The Problem



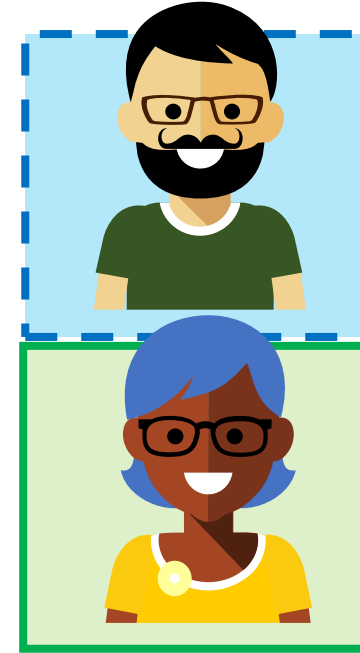
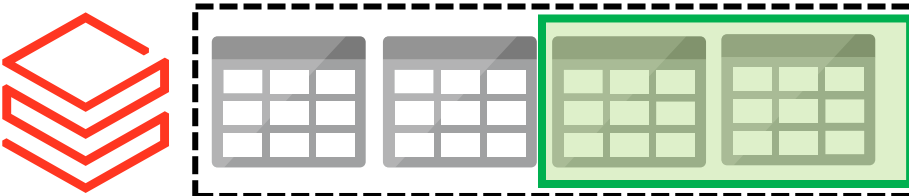
The Problem



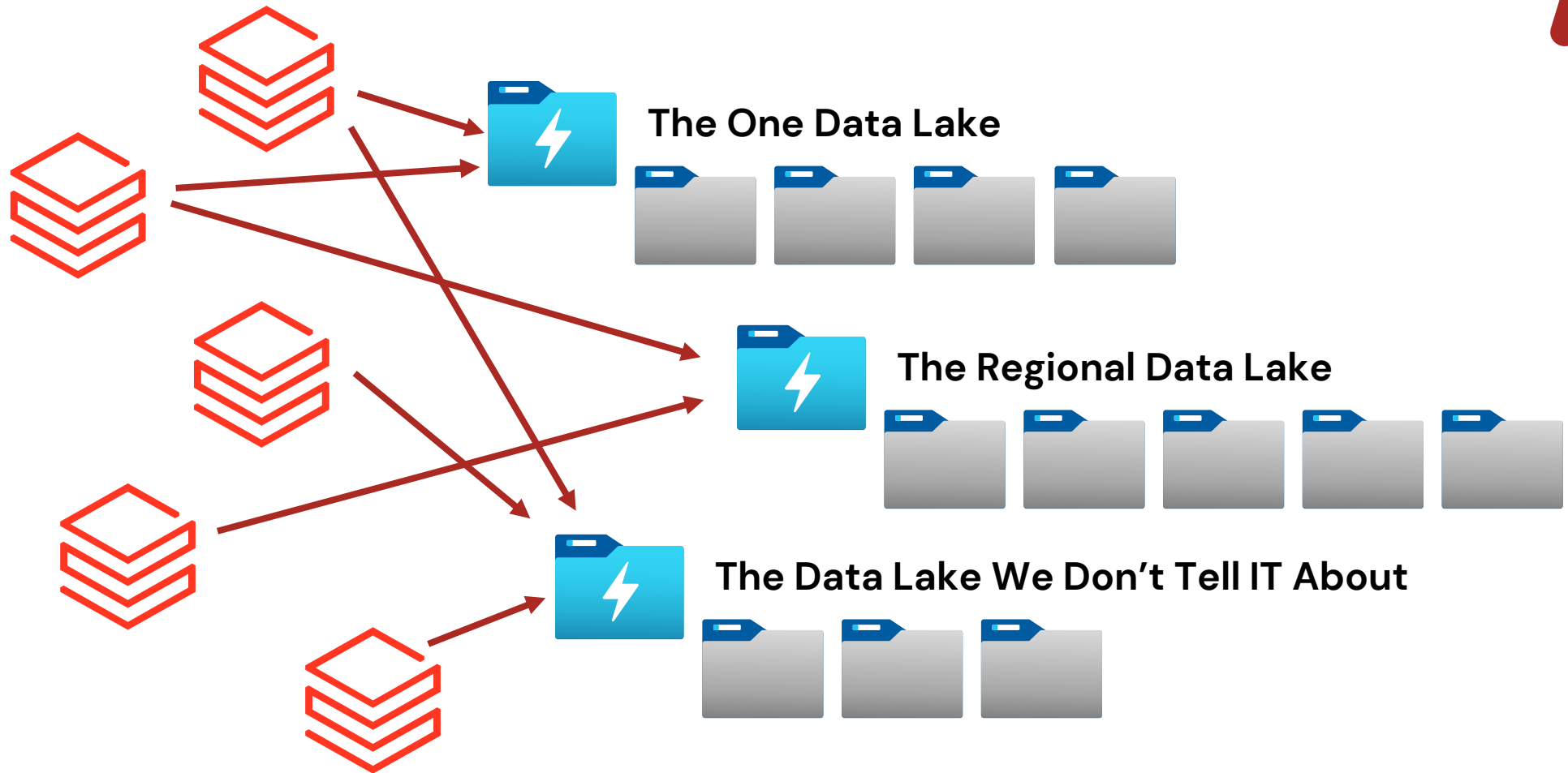
Engineering
Workspace



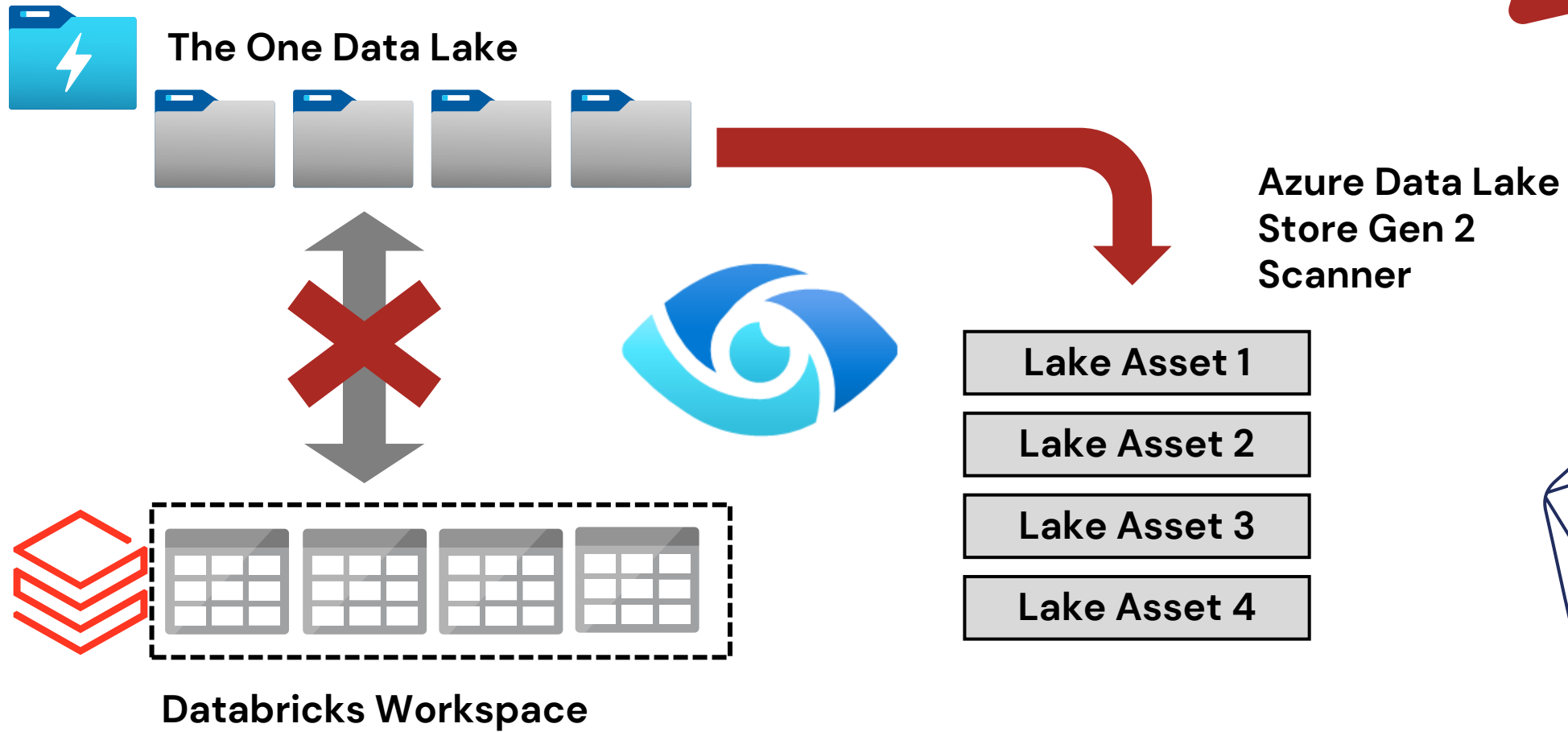
Data Science
Workspace

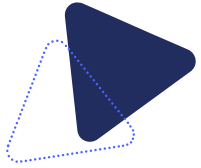
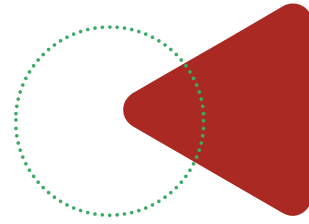


It's Rarely So Straight Forward

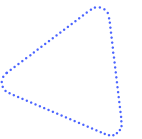
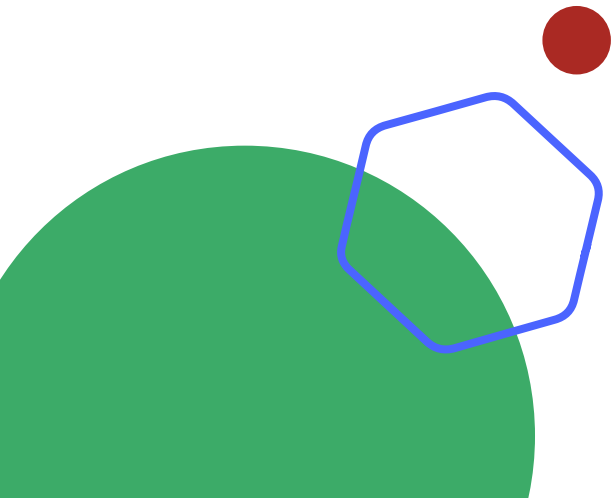


Azure Purview





Account Console, Metastores & Workspace



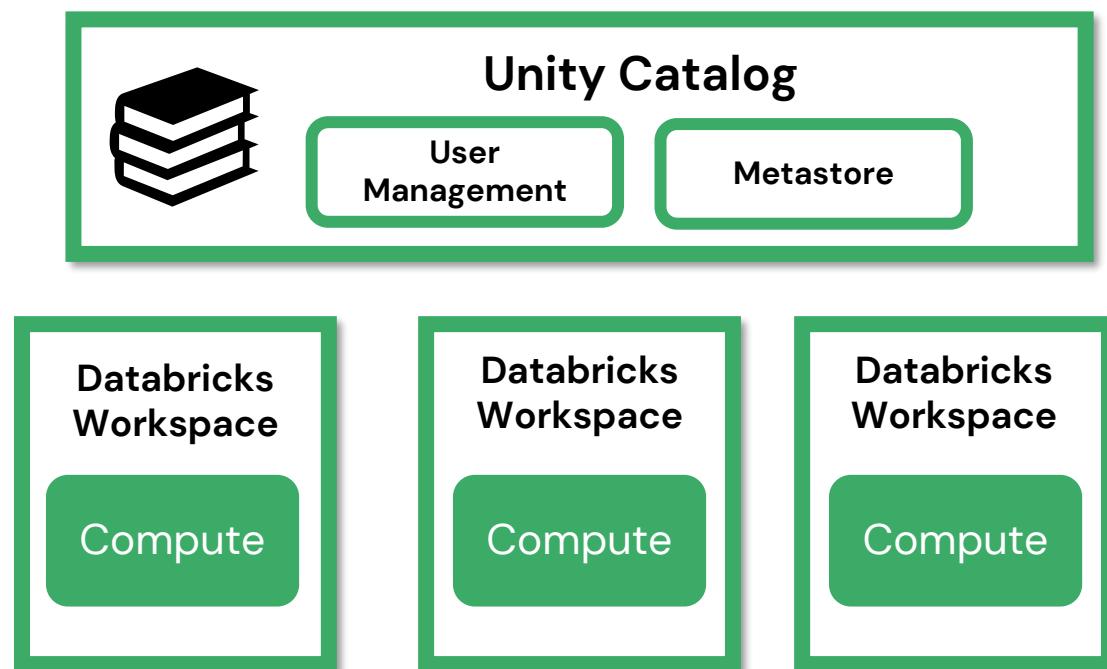
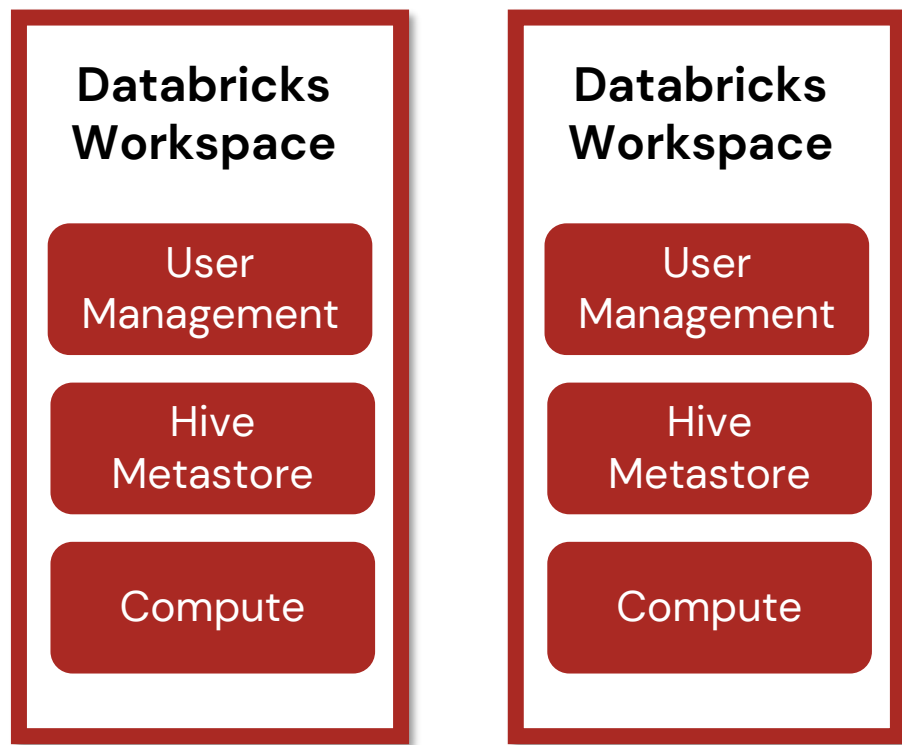
Databricks **Unity Catalog (UC)**



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

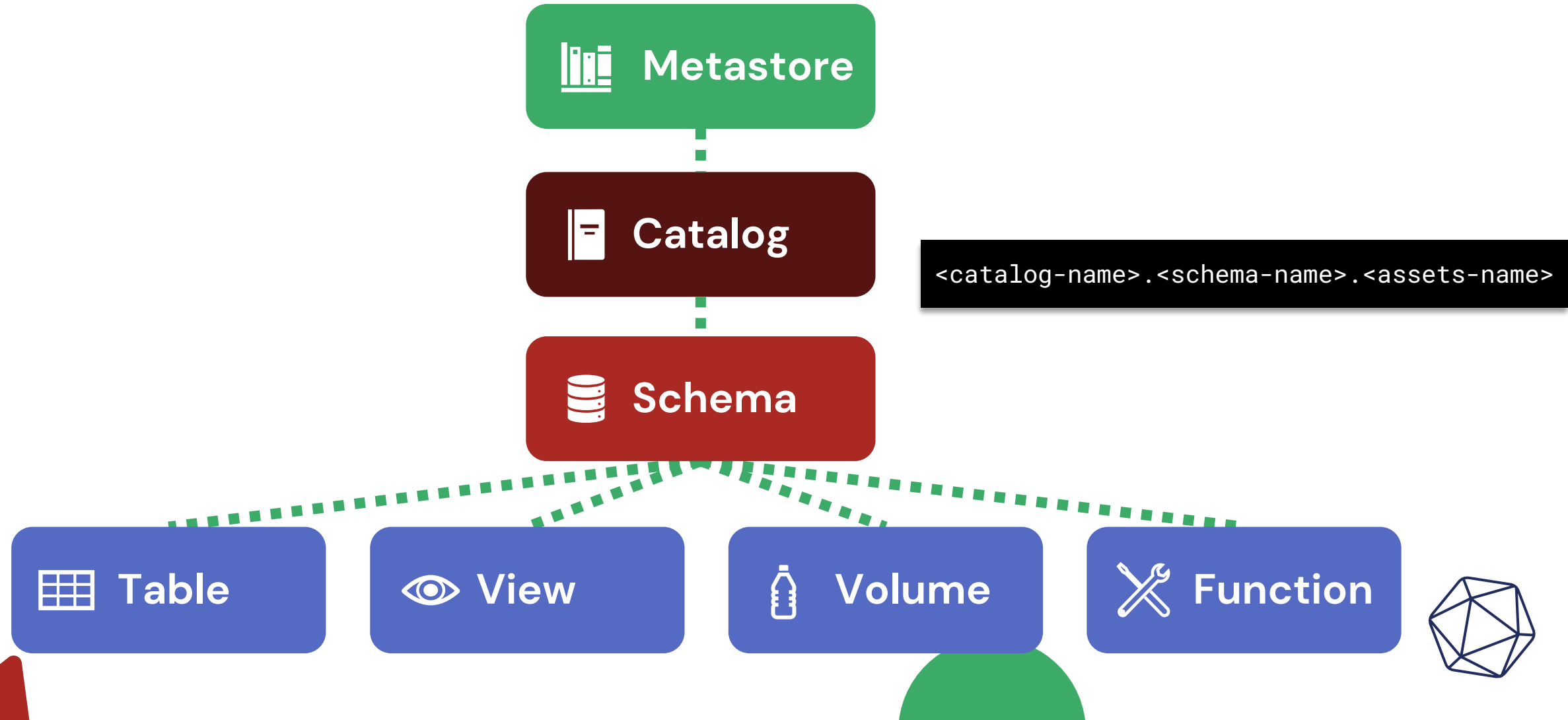
Legacy Databricks

Databricks Unity Catalog



Unity Catalog **Object Model**

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



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



 **Metastore**



**Databricks
Workspace**



**Automated
Jobs**



**Streaming
Pipelines**



**Workspace
Security**



**Interactive
Clusters**



**Machine
Learning**

Unity Catalog - Workspaces



 **Metastore**



**Databricks
Workspace**

DEV



**Databricks
Workspace**

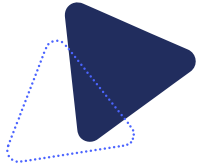
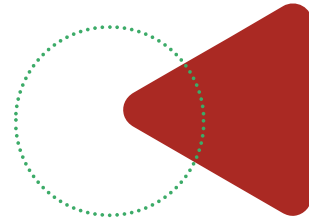
UAT



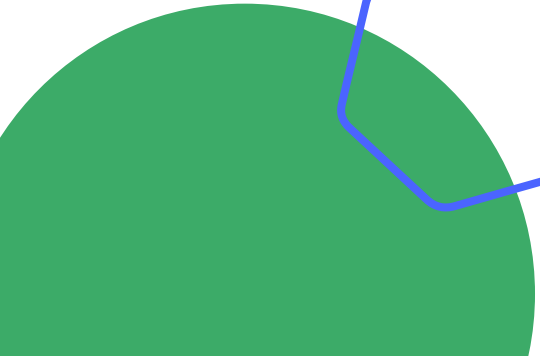
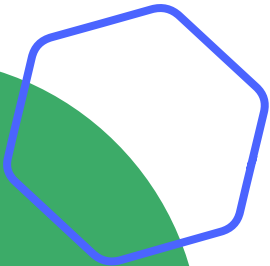
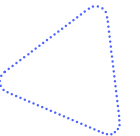
**Databricks
Workspace**

PROD

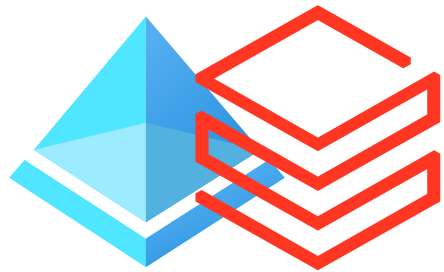




Account Console



Account Console



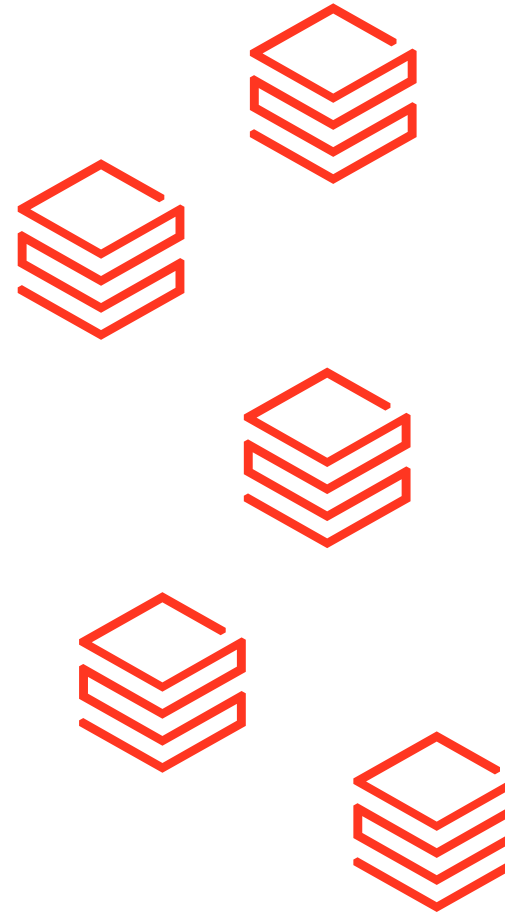
Central Databricks
Account at the AD
Tenant level

Unity Catalog

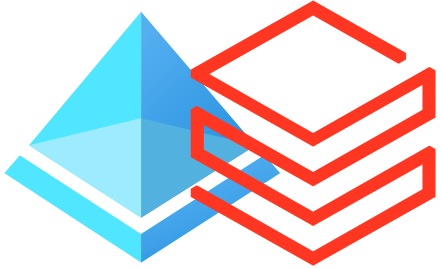
Data Access

Data Documentation

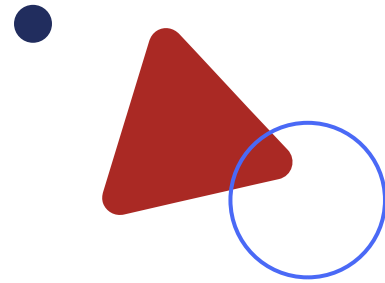
Data Lineage



Account Console



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



Advancing Analytics

Workspaces
Catalog
Usage
User management
Cloud resources
Previews
Settings

Account console

Manage your Databricks account at scale



Workspaces

Configure workspace settings. Workspaces contain notebooks, libraries, queries, and workflows



Catalog

Manage metastores as your top-level container for catalogs, schemas (also called databases), views and tables



Usage

View usage details and graphs for your account in Databricks Units (DBU) or estimated costs (in \$USD)



Users & groups

Manage identities for use with jobs, automated tools and systems



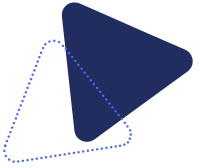
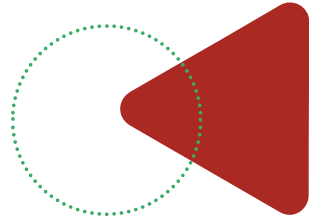
Cloud resources

Manage network connectivity configurations for your resources

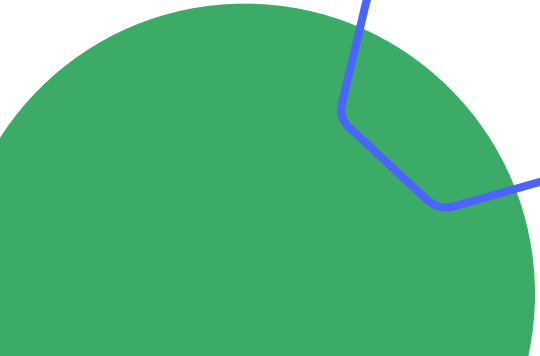
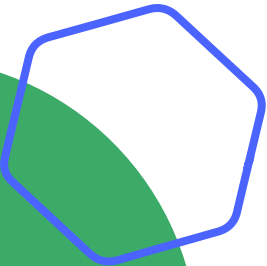
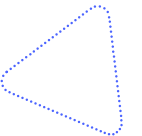


Settings

Configure your Databricks account user provisioning and other settings

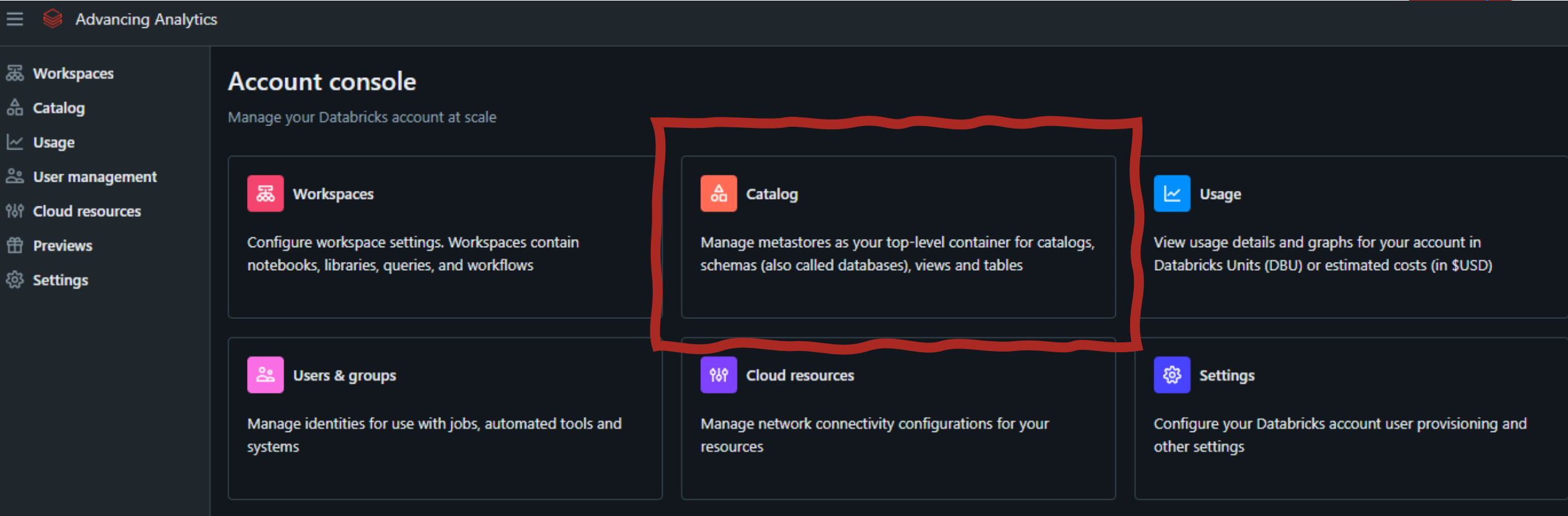


Create a *Metastore*



Create a **Metastore**

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



The screenshot shows the Databricks Account console interface. On the left is a sidebar with navigation links: Workspaces, Catalog, Usage, User management, Cloud resources, Previews, and Settings. The main area is titled 'Account console' with the subtitle 'Manage your Databricks account at scale'. It contains six tiles: Workspaces, Catalog, Usage, Users & groups, Cloud resources, and Settings. The 'Catalog' tile is highlighted with a red hand-drawn rectangle. The 'Catalog' tile text reads: 'Manage metastores as your top-level container for catalogs, schemas (also called databases), views and tables'.

Account console
Manage your Databricks account at scale

- Workspaces**
Configure workspace settings. Workspaces contain notebooks, libraries, queries, and workflows
- Catalog**
Manage metastores as your top-level container for catalogs, schemas (also called databases), views and tables
- Usage**
View usage details and graphs for your account in Databricks Units (DBU) or estimated costs (in \$USD)
- Users & groups**
Manage identities for use with jobs, automated tools and systems
- Cloud resources**
Manage network connectivity configurations for your resources
- Settings**
Configure your Databricks account user provisioning and other settings

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

Advancing Analytics

Workspaces
Catalog
Usage
User management
Cloud resources
Previews
Settings

Catalog > Create metastore >

Create metastore

1 Create metastore — 2 Assign to workspaces

* Name

* Region

Select the region for your metastore. You will only be able to assign workspaces in this region to this metastore.

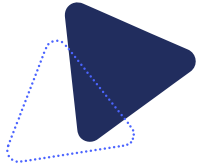
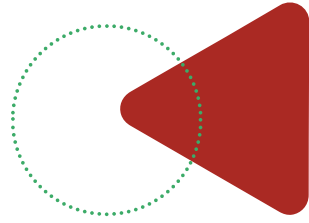
ADLS Gen 2 path (optional) ?

Optional location for storing managed tables data across all catalogs in the metastore. [Learn more](#)

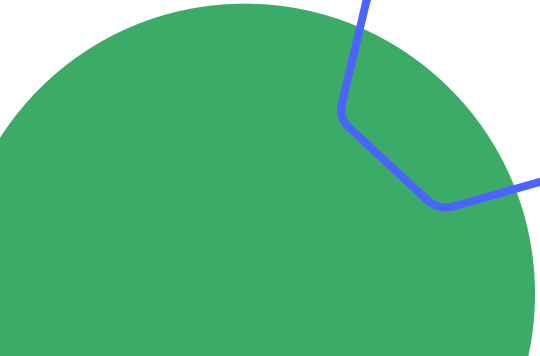
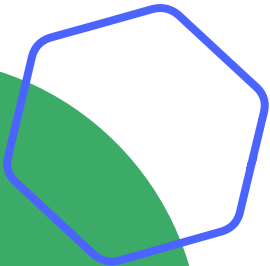
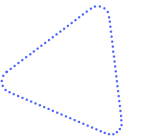
Access Connector Id ?

Advanced options

Create Cancel



Databricks Access Connector



Databricks Access Connector

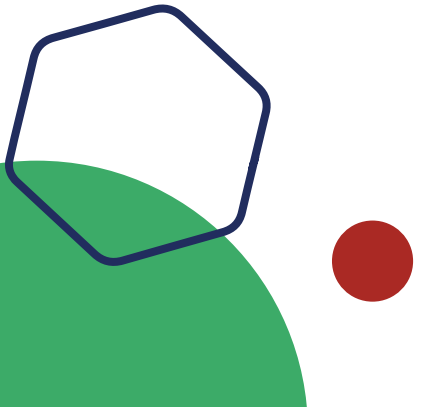
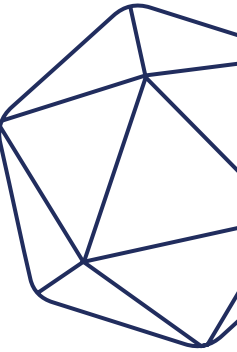
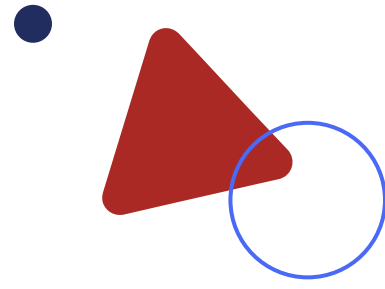
The **Access Connector for Azure Databricks** allows you to connect **managed identities** to an Azure Databricks account for accessing data registered in Unity Catalog.

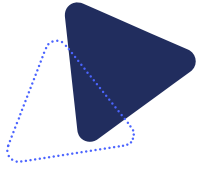
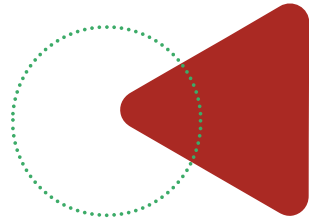
Primary Uses:

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

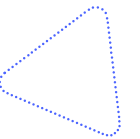
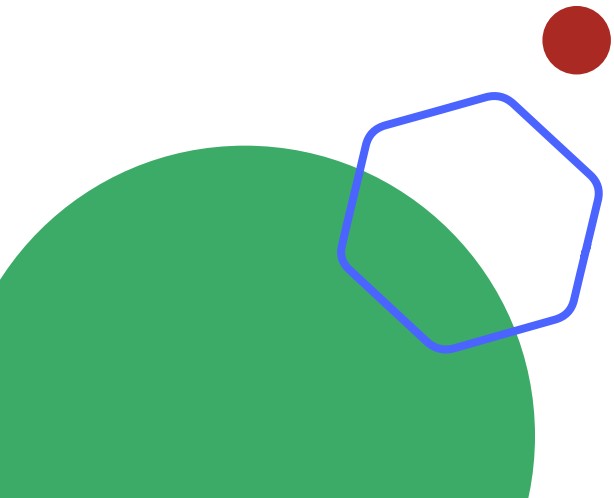
Benefits:

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

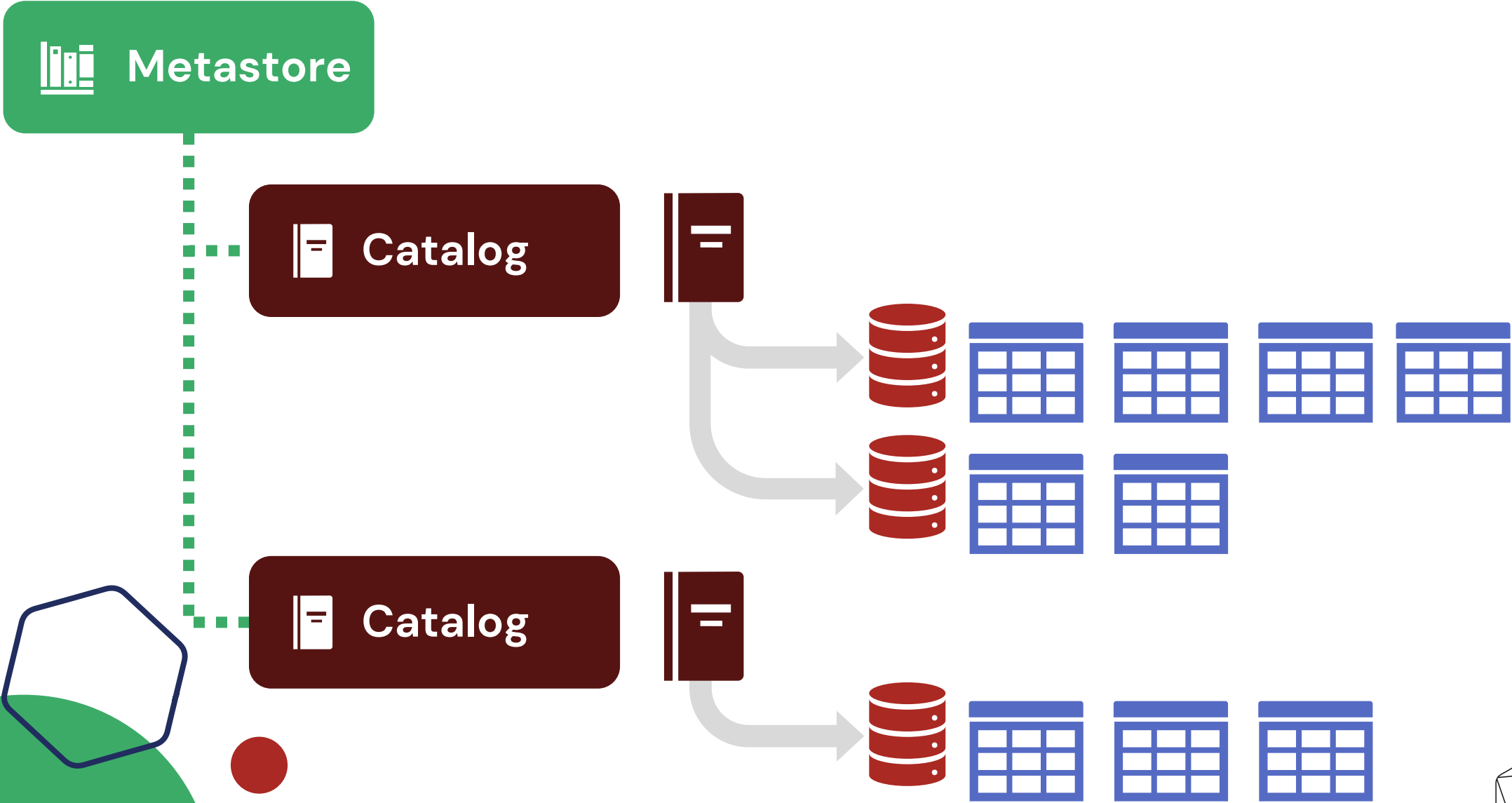




Unity Catalog Objects



Unity Catalog - Catalog



Unity Catalog – Catalog Syntax

 Metastore

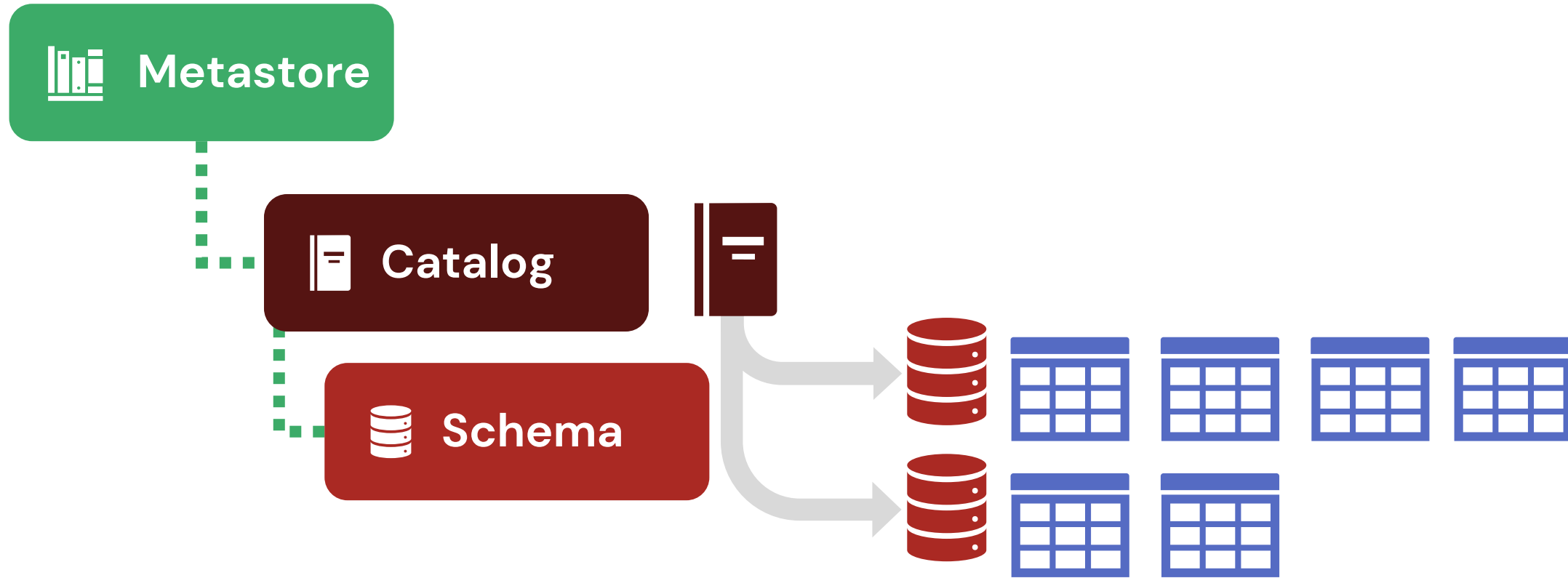
 Catalog

```
-- 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!

Unity Catalog – Schema



Unity Catalog – Schema Syntax



Metastore



Catalog



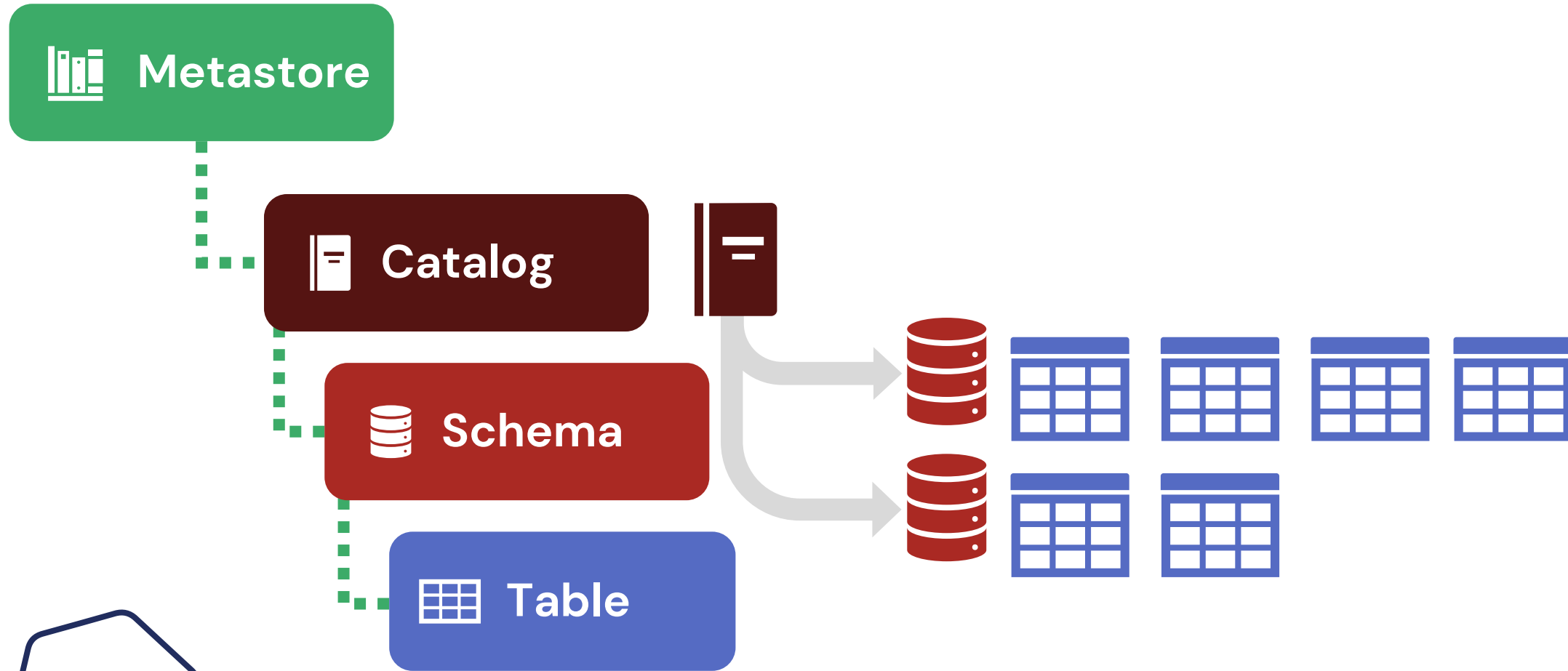
Schema

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

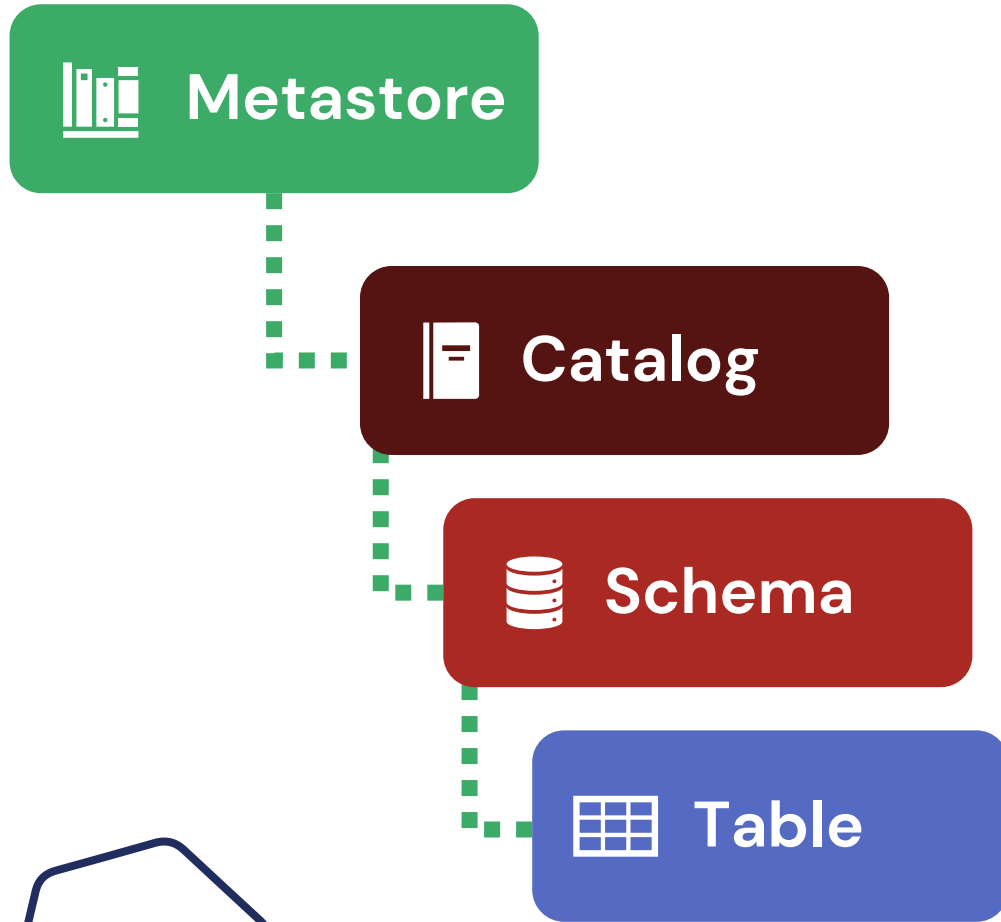
-- Explicit Two-Part Creation
CREATE SCHEMA SalesMart.ItemSales

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

Unity Catalog – Table

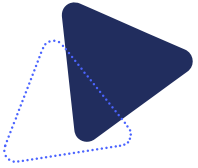
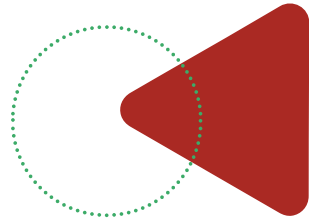


Unity Catalog – Table Syntax

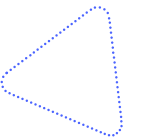
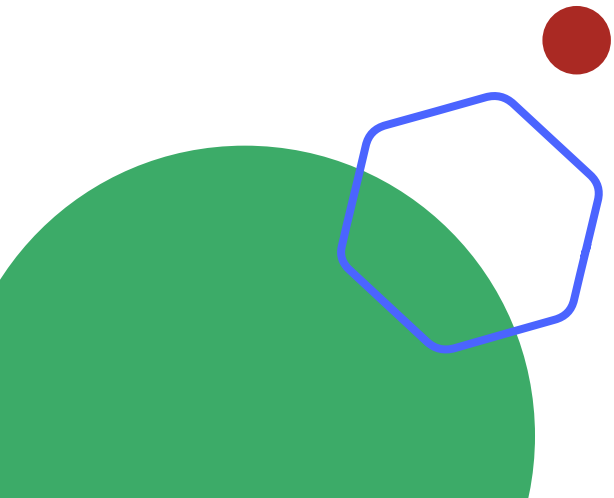


```
-- Create a Managed Table Object
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
    (ID int, ProductID int, Amount decimal)
```

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



Managed VS External



Managed VS Unmanaged Tables



Managed Table

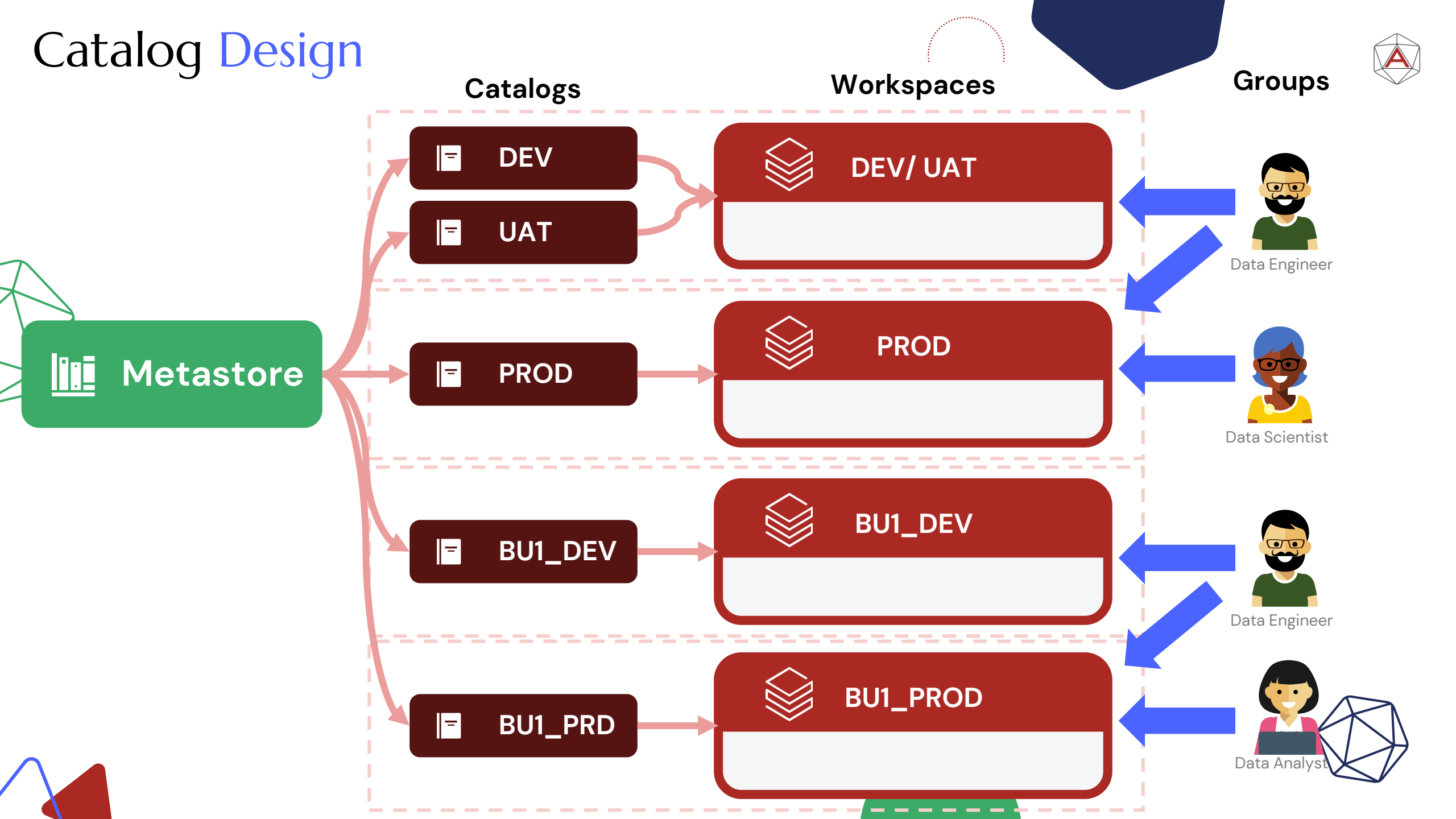
- Stored in managed default storage
- Dropping a table deletes data files
- Only supports Delta format
- Data Linked to SQL Object
- No Lake Folder Control
- No Storage Credentials Required

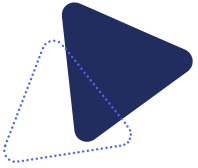
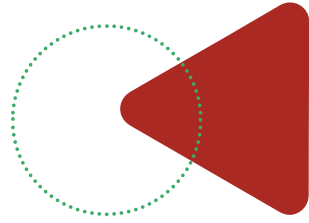


External Table

- 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

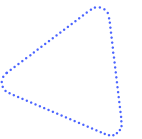
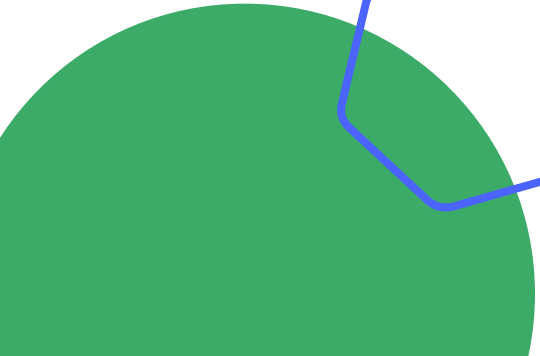
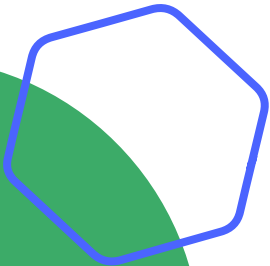
Catalog Design

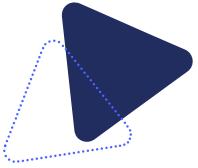
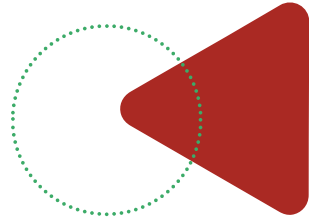




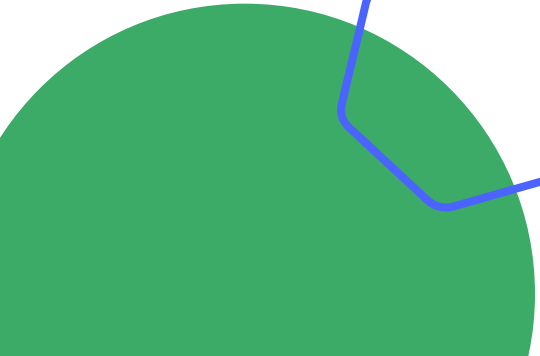
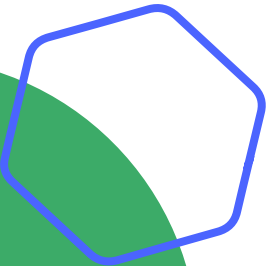
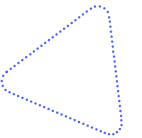
Demo: Unity Catalog Objects

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





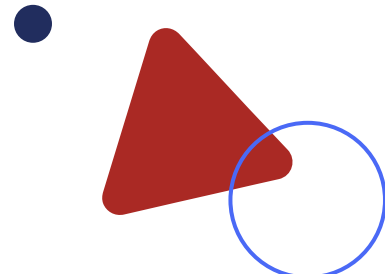
Unity Catalog Features



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



Catalog

fm-sbx-warehouse-xxs 2XS

Type to search...

My organization

system

fm_sandbox

01_raw_advworks

address

customer

customeraddress

product

productcategory

productdescription

productmodel

productmodelproductdescription

salesorderdetail

salesorderheader

Catalog Explorer > fm_sandbox > 01_raw_advworks >

address

Overview

Sample Data

Details

Permissions

History

Lineage

Insights

Quality

Description

The address table contains information about the physical location of customers, employees, and vendors. It includes details such as the street address, city, state/province, postal code, and country/region. This data is important for shipping and billing purposes, as well as for analyzing sales and marketing trends by geographic location. The table also includes a rowguid column for tracking changes and a ModifiedDate column for auditing purposes. Additionally, the table includes columns for year, month, day, and time to allow for time-based analysis.

Filter columns...

Column	Type	Comment	Tags	Column masking rule
AddressID	bigint			
AddressLine1	string			
AddressLine2	string			
City	string			
Country/Region	string			

AI generate

About this table

Owner: Falek Miah

Data source: Delta

Popularity: ****

Tags: Add tags

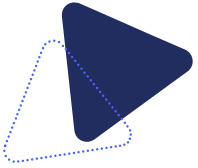
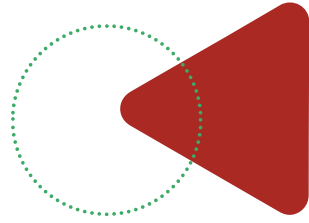
Row filter: Add filter

Information Schema

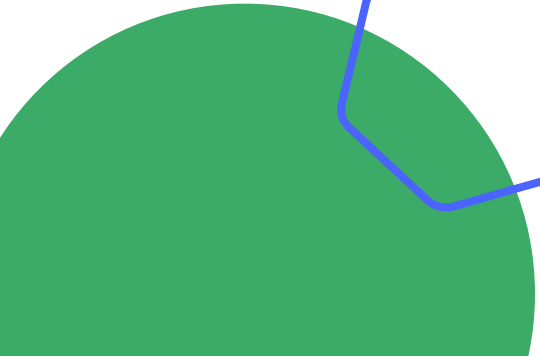
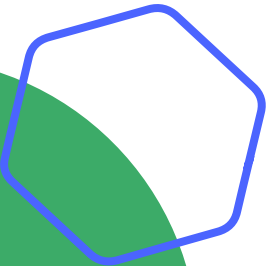
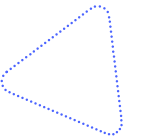
Another nice addition is the “[information_schema](#)” that is automatically added to every Catalog

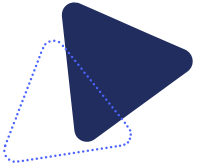
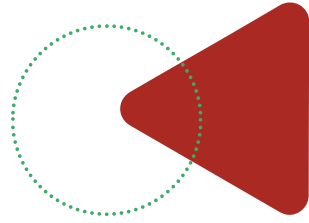
This can be queried to automate common tasks, just as we would with SQL Server sys DMVs

- >  adventureworks
 - >  default
 - >  dw
 - ✓  information_schema
 -  catalog_privileges
 -  catalogs
 -  check_constraints
 -  columns
 -  constraint_column_usage
 -  constraint_table_usage
 -  information_schema_catalog_name
 -  key_column_usage
 -  parameters

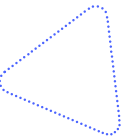
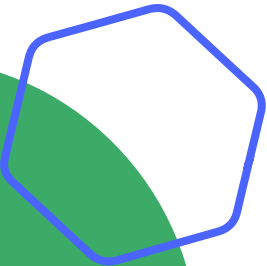


Demo: Catalog Explorer

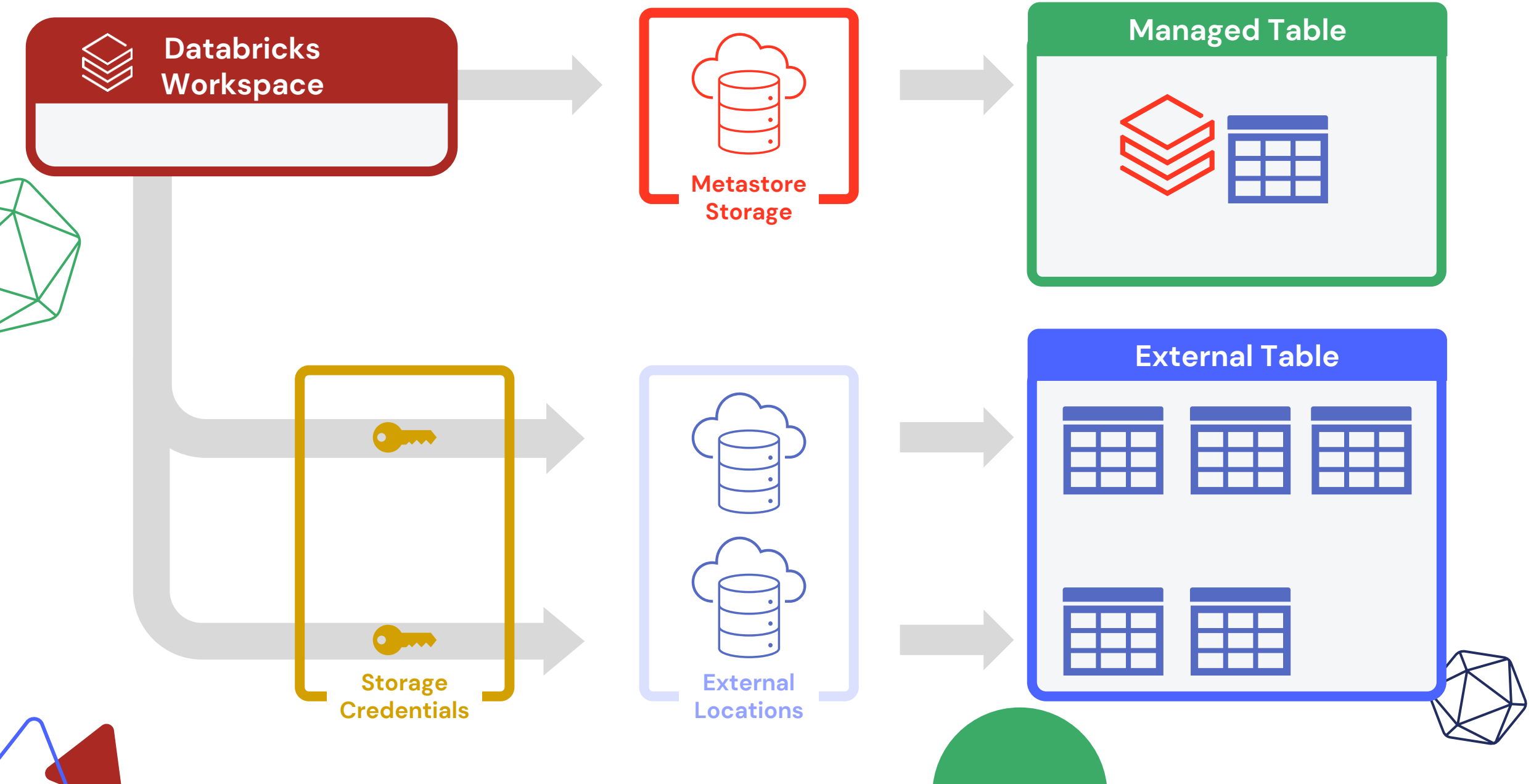




External Locations & Storage Credentials



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

Data

Delta Sharing

Storage Credentials

External Locations

advancinglake

External

1 Locations

Name

advancinglake

Create a new storage credential

A storage credential represents an authentication and authorization mechanism for accessing data stored on your cloud tenant. [Learn more](#)

☒ Managed identity (Recommended)

☐ Service Principal

Storage credential name

dbxmanagedident

Access connector ID

/subscriptions/6ew2d13230-7bob-3234-8c38-8d32bb553332f/resourceGroups/UnityCat/providers/

Comment

Connect to a Storage Account using the DBX Managed Identity

Cancel

Create

External Locations – Setup

Data

Delta Sharing

Storage Credentials

dbxmanagedident

sp_unity_manager

unitymanagedident

Storage

3 Credentials

Name

dbxmanag

sp_unity_r

unitymana

Create a new external location

An external location is a cloud storage url (and paired credential) that allows access to data stored on your cloud tenant. [Learn more](#)

Copy from mount point

External location name

MyOtherLake

URL

abfss://root@advancingadls.dfs.core.windows.net/

Storage credential

dbxmanagedident

Comment

The Lake I want to connect to

External Locations – Syntax



Managed Table



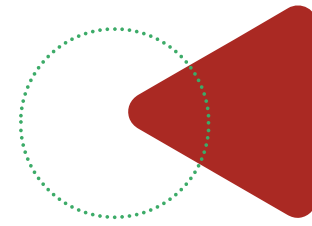
```
-- Create Managed Table
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
(ID int, ProductID int, Amount decimal)
```

External Table

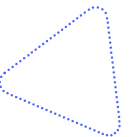
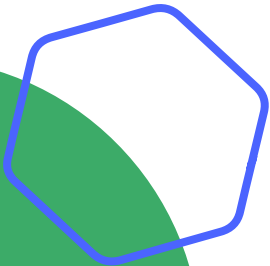
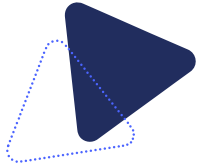


```
-- Create External Table
USE SalesMart.ItemSales;
CREATE TABLE FactItemSales
      (ID int, ProductID int, Amount decimal)
LOCATION "abfss://root@mylake.dfs.core.windows.net/salesitems/factitemsales"

-- 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);
```



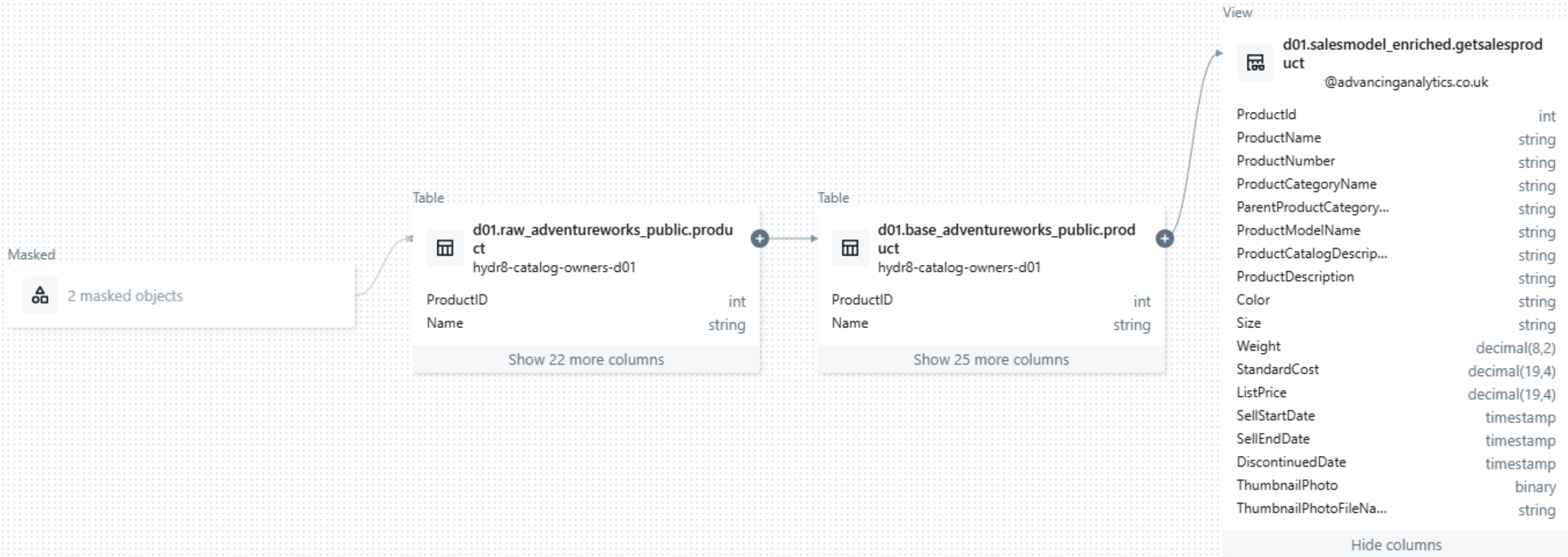
Lineage

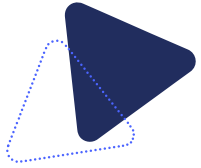
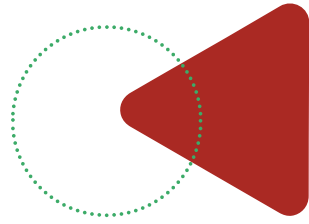


Unity Catalog - Lineage Tracking

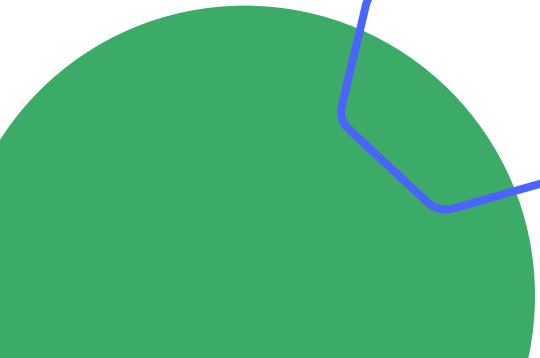
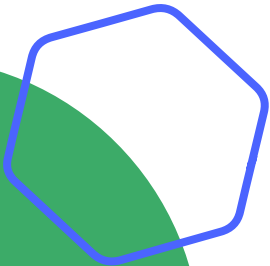
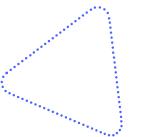
Once activated, the Lineage feature will automatically trawl all Spark jobs to determine whether data moved between registered Unity Catalog table.

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



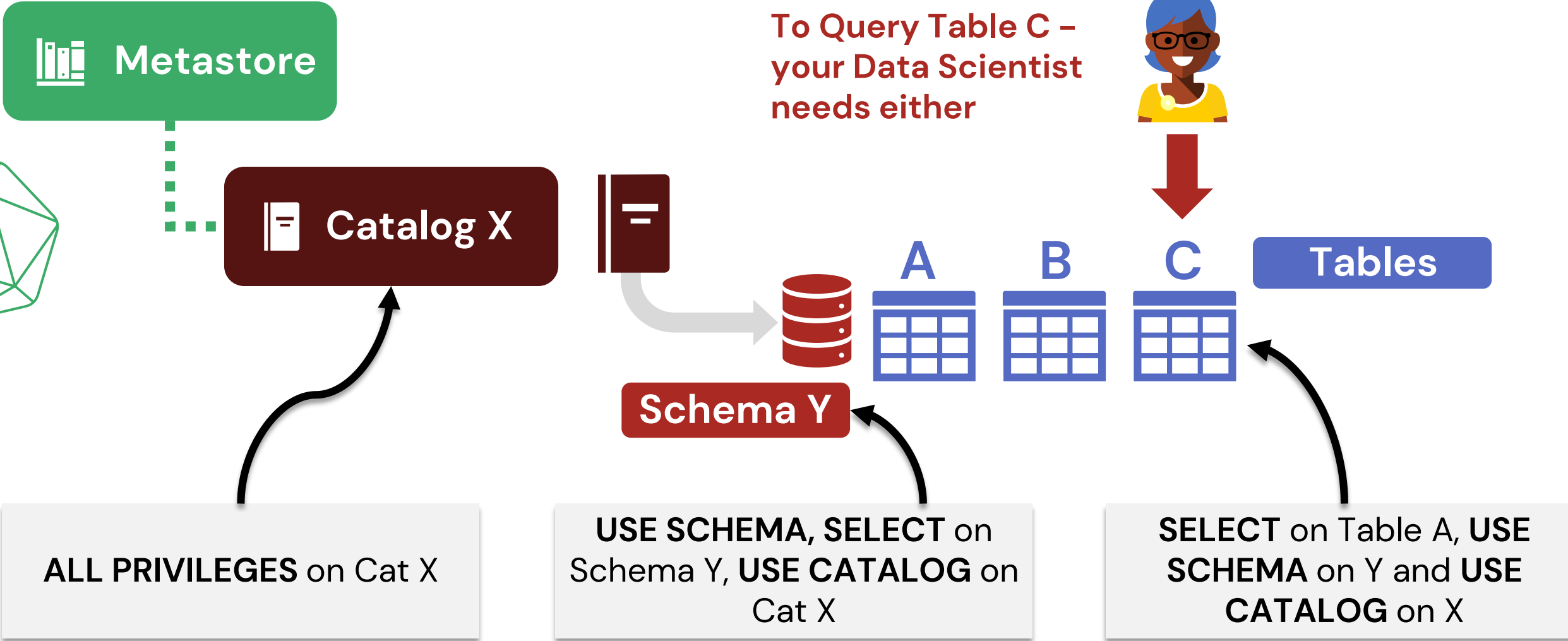
Securable	Privileges
Metastore	CREATE CATALOG CREATE EXTERNAL LOCATION CREATE RECIPIENT CREATE SHARE
Catalog	ALL PRIVILEGES BROWSE CREATE SCHEMA USE CATALOG
Schema	ALL PRIVILEGES CREATE FUNCTION CREATE TABLE CREATE VIEW USE SCHEMA
Table / View	ALL PRIVILEGES SELECT MANAGE MODIFY

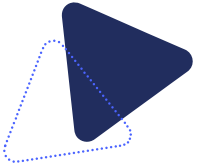
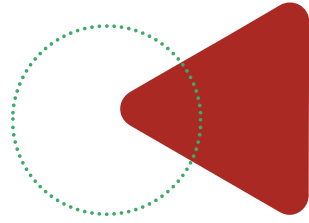
You cannot access child objects unless you have “USAGE” on the parent object

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

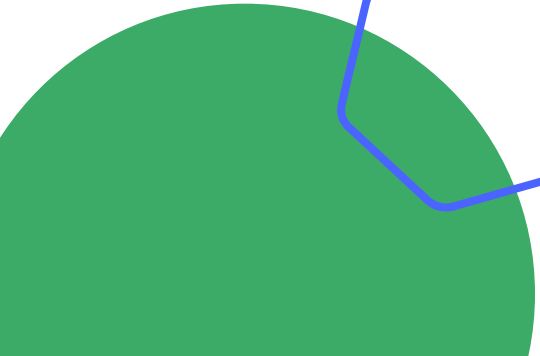
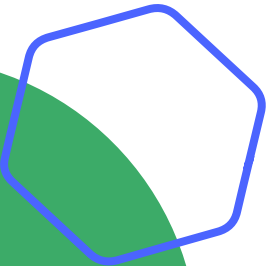
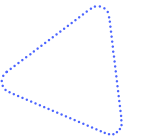


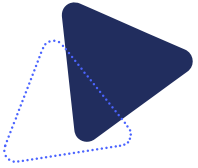
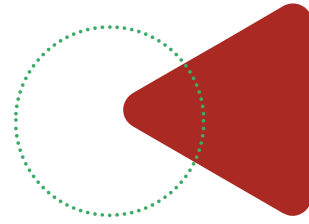
Unity Catalog – Privileges Example



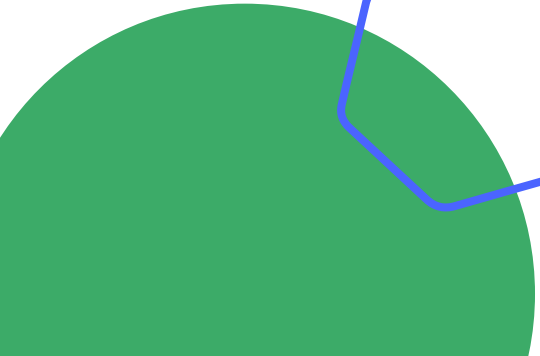
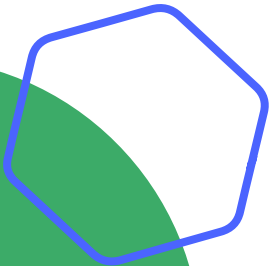
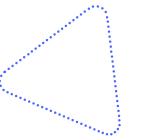


Demo: *Managing Permissions*





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

Add/Edit tags for zach_catalog.warehouse.address

Key	Value (optional)
Type a key	Type a value
scd: 1 X	
Cancel Save tags	

Cmd 1

```
1 SELECT *
2 FROM zach_catalog.information_schema.table_tags
```

▶ (1) Spark Jobs

Table v +

	catalog_name	schema_name	table_name	tag_name	tag_value
1	zach_catalog	warehouse	address	scd	1

1 row | 2.97 seconds runtime

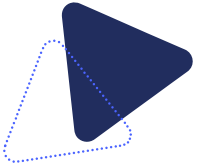
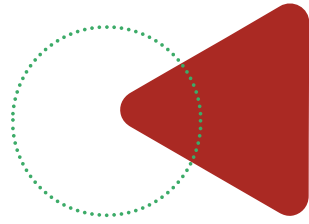
Command took 2.97 seconds -- by zach@advancinganalytics.co.uk at 31/01/2024, 14:38:31 on cluster01

Comments

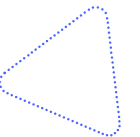
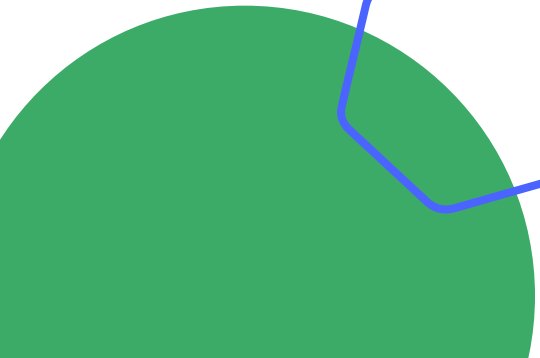
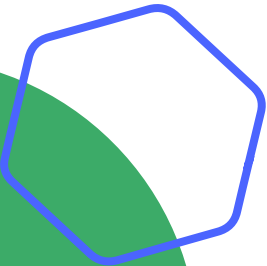
- Descriptive comments to data objects
- Insights into dataset usage, assumptions, and quality
- Improves knowledge sharing & collaboration
- Added to catalogs, schemas, tables, and columns
- View in Catalog Explorer or via SQL commands
- Maintains history of updates

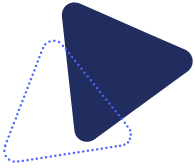
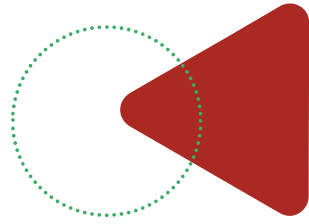
```
COMMENT ON CATALOG catalog IS 'comment';  
COMMENT ON SCHEMA schema IS 'comment';  
COMMENT ON TABLE table IS 'comment';  
COMMENT ON COLUMN table.c1 IS 'comment';
```

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.	



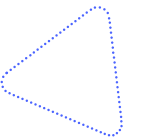
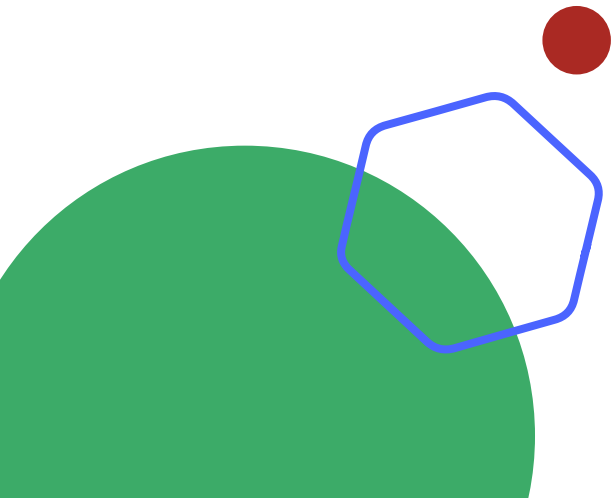
Demo: Data Discovery

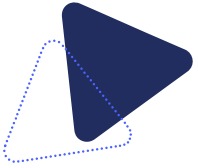
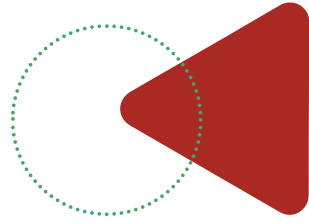




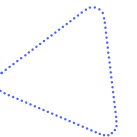
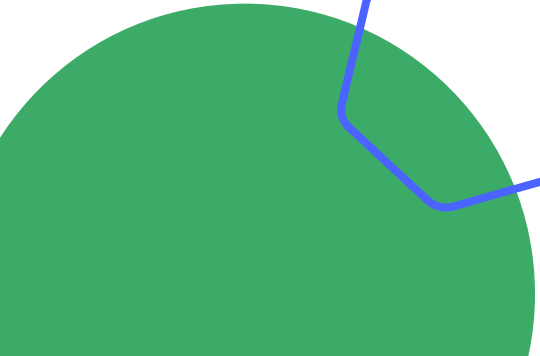
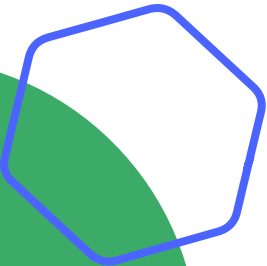
LABO1 – Unity Catalog Objects

Creating and securing UC objects



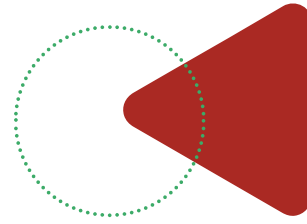


UC Limitations

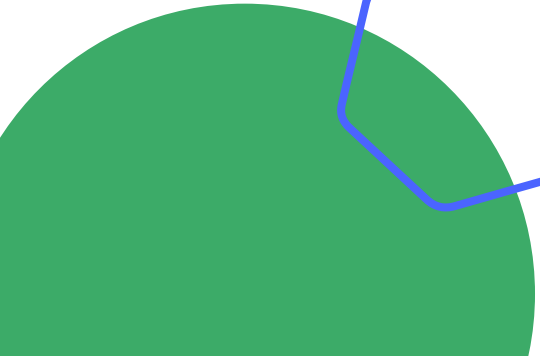
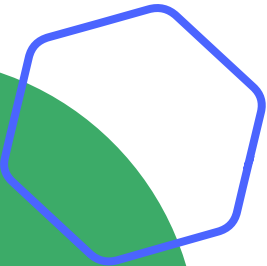
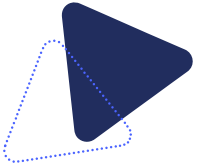


Unity Catalog Quotas

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

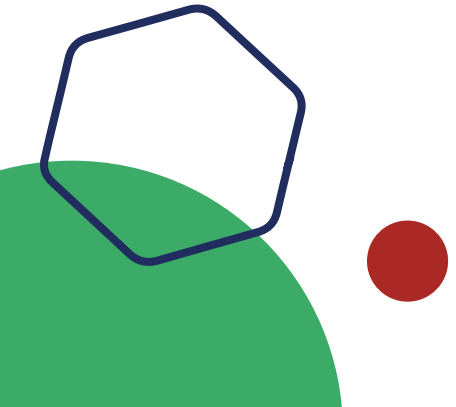
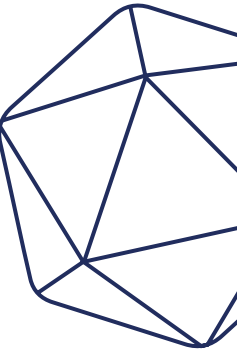
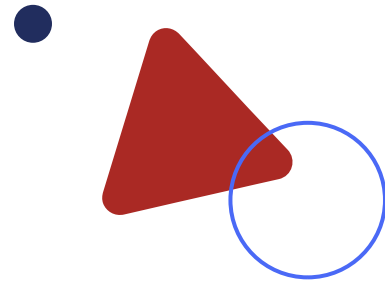


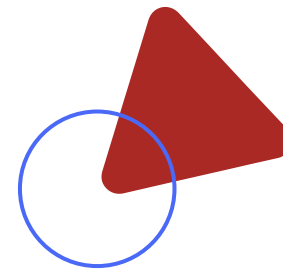
Recap



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

