Unity Catalog

Name: Priyeshwar

Mail: priyesh2664@gmail.com

Introduction

Databricks Unity Catalog (UC) is a centralized governance solution designed to manage data and AI assets within the Databricks ecosystem. It works seamlessly across multiple clouds and platforms, providing a unified approach to securing, organizing, and tracking data throughout its lifecycle. UC handles both structured and unstructured data, as well as machine learning models, notebooks, and dashboards, all under one governance framework. By offering fine-grained access control and integrated data lineage, it helps organizations meet compliance requirements while enabling secure collaboration.

Purpose & Benefits

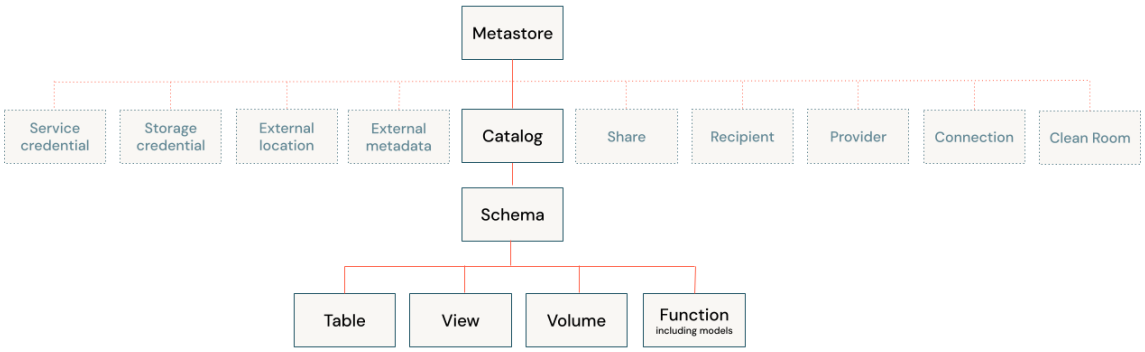
Unity Catalog addresses the challenges of fragmented data governance by creating a single control point for managing access policies across workspaces and data sources. It allows administrators to define permissions once and enforce them consistently, avoiding the complexity of maintaining separate tools or rules. This reduces security risks and ensures compliance with data privacy regulations. Additionally, UC enhances productivity by improving data discovery and providing built-in auditing and lineage features. Teams can quickly find the data they need, understand its origin, and see how it has been transformed or used. For organizations sharing data across departments or even external partners, Delta Sharing makes it possible to do so securely without compromising governance.

Architecture & Components

Unity Catalog is built around a clear hierarchical object model:

- Metastore – The top-level container that organizes metadata for all data assets.
- Catalog – A logical grouping of schemas, often aligned with a department, data domain, or environment (e.g., production vs. development).
- Schema – Also called a database, it contains tables, views, and volumes.

- Tables, Views, and Volumes – Tables hold structured data, views are virtual tables created with queries, and volumes store unstructured files.
- Models – Registered machine learning models that can be governed alongside other data assets.



On top of this structure, UC provides access control, which uses an inheritance model where permissions granted at a higher level flow down to lower-level objects. It also includes operational intelligence tools that give real-time visibility into how assets are being used, with AI-powered alerts and monitoring capabilities.

How to create Unity Catalog ?

Prerequisites

Before creating Unity Catalog in Azure, ensure the following:

- An **Azure subscription**.
- An **Azure Databricks Premium workspace**.
- Admin privileges in Databricks.
- An **Azure Active Directory (AAD) identity** with permission to create resources.
- An **Azure Data Lake Storage Gen2 (ADLS)** account with hierarchical namespace enabled.

Steps to Create Unity Catalog in Azure

Step 1: Create Unity Catalog Metastore

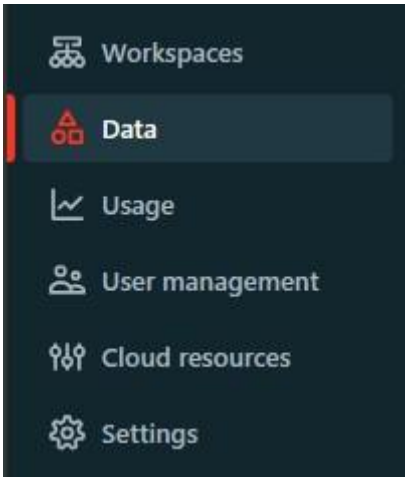
The metastore is like the central brain of Unity Catalog (stores all schemas, tables, permissions).

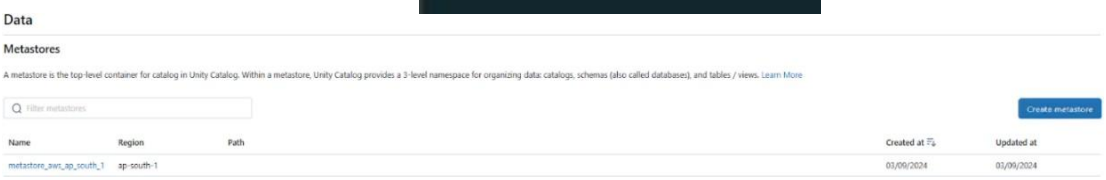
1. Go to the Databricks Account Console → Data → Metastores.

2. Log in as an account admin.

3. In the left-hand navigation panel, select **Data** to view available metastores

4. Click Create Metastore.





Step 2: Provide a name and select the region for the metastore

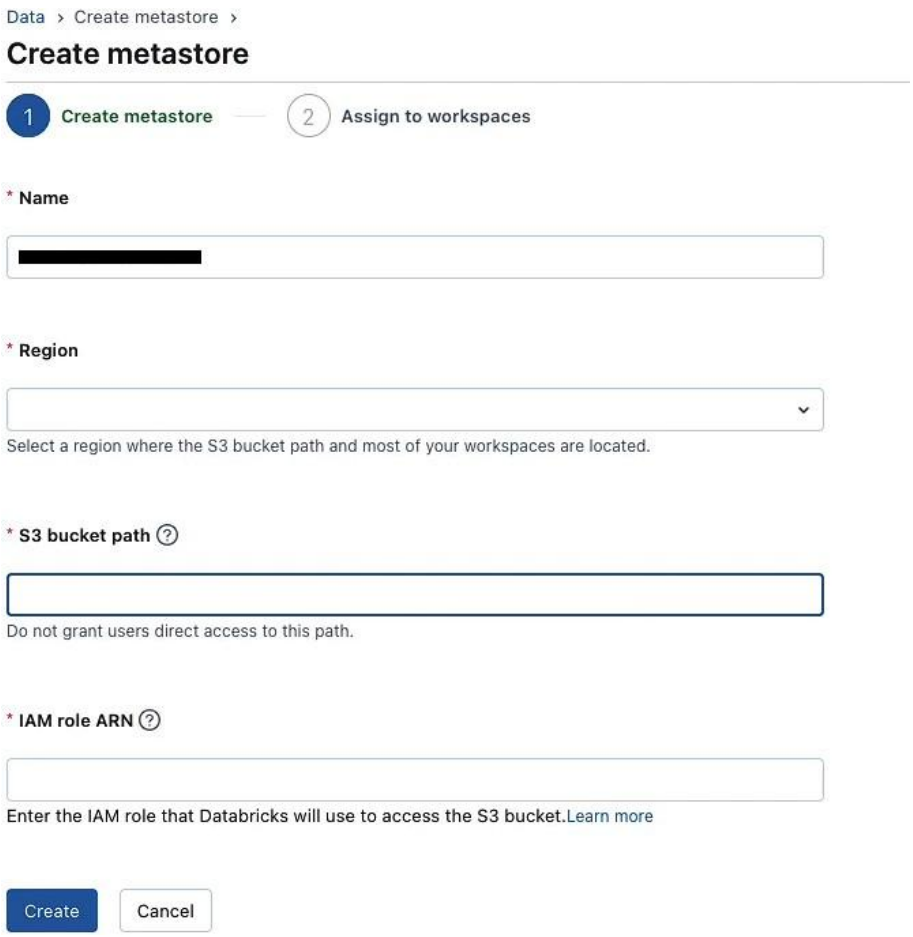
Note: same region as your workspaces.

Name: Example: uc_metastore

Region: Must match the Azure region of your Databricks workspace.

Storage Root: An ADLS Gen2 container path, e.g.:

abfss://uc-root@yourstorage.dfs.core.windows.net/



Step 3: Assign a Metastore Admin.
Designate an admin who will have full control over the Unity Catalog metastore.

Step 4: Storage Configuration

Unity Catalog requires a root storage for managed tables and may use external storage for additional datasets.

1. In **Azure Portal**, create a **Storage Account (ADLS Gen2)**.
2. Create a **Container** (e.g., uc-root).
3. Register a **Service Principal (SPN)** in Azure AD.

- Copy **Client ID**, **Secret**, and **Tenant ID**.

4. Assign the **Storage Blob Data Contributor** role to the SPN on the ADLS account.
5. In **Databricks**:
   - Navigate to **External Locations → Create Location**.
   - Point to the ADLS path and authenticate with the SPN.

Step 5: Assign Metastore to Workspace

1. Go to **Account Console → Workspaces**.
2. Select your workspace.
3. Click **Assign Metastore**, then choose uc_metastore.

At this point, the workspace is governed by Unity Catalog

Step 6: Create 3-Level Namespace

Unity Catalog introduces a **three-level namespace**: catalog.schema.table.

In a Databricks Notebook (SQL mode), run:

```
CREATE CATALOG sales_catalog;
CREATE SCHEMA sales_catalog.retail;

CREATE TABLE sales_catalog.retail.orders (
   order_id INT,
   product STRING,
   amount DOUBLE
);
```

Step 7: Set Permissions

Unity Catalog supports **GRANT** and **REVOKE** commands for access control.

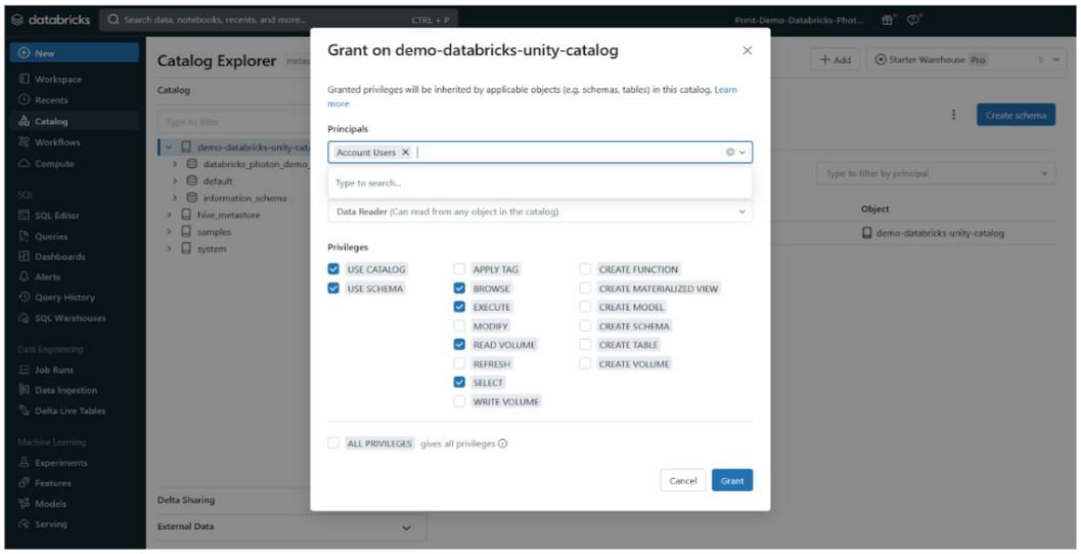
Examples:

-- Grant read access to analysts

GRANT SELECT ON TABLE sales_catalog.retail.orders TO `analyst_group`;

-- Grant write access to data engineers

GRANT MODIFY ON TABLE sales_catalog.retail.orders TO `data_engineers`;

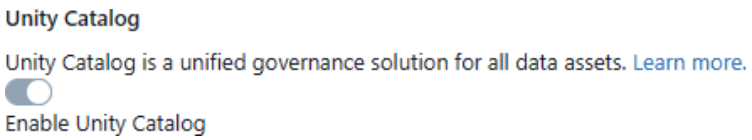


Access Control and Security - Databricks Unity Catalog

Step 8: Assign and Confirm

Click **Assign**, then confirm by selecting **Enable** in the dialog.

(Optional) If creating a new workspace, enable Unity Catalog by toggling **Enable Unity Catalog** during setup, choosing a metastore, and saving.



Step 9: Verify Setup

To check if Unity Catalog is enabled, run the following SQL in a notebook:

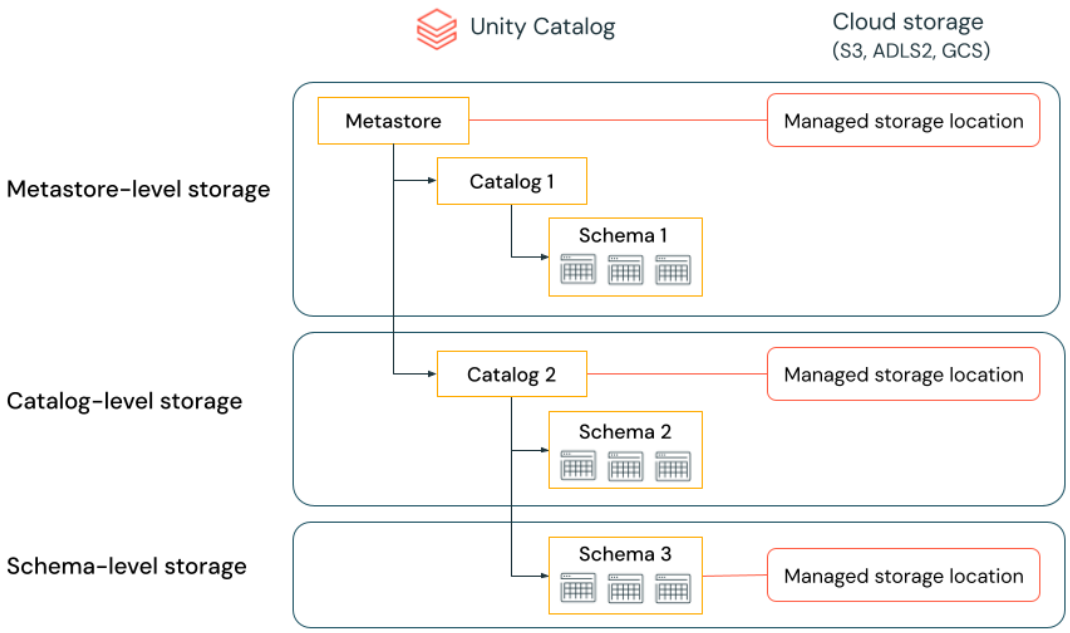
SELECT CURRENT_METASTORE();

Managed Storage Location Hierarchy

Unity Catalog provides flexibility in defining managed storage locations depending on organizational requirements for data isolation. Locations can be assigned at the metastore, catalog, or schema level.

Example: A compliance policy requires HR production data to be stored in a secure container:
abfss://mycompany-hr-prod@storage-account.dfs.core.windows.net/unity-catalog

By creating a catalog 'hr_prod' and assigning this location, all managed tables in the catalog are stored in the specified container. Optionally, schema-level locations can be used for finer control.



Evaluation order of storage locations:
1. Schema-level location (highest priority).
2. Catalog-level location.
3. Metastore-level location (default if none defined above).

Key Features in Azure Unity Catalog

- **Centralized Governance**: Policies apply across all Azure Databricks workspaces.
- **Integration with Azure AD**: Access control is based on Azure identities.
- **Data Lineage**: Track data flows across pipelines and notebooks.
- **Cross-Cloud Sharing**: Use Delta Sharing to securely share data across Azure and other clouds.
