

Unity Catalog – Mini Project Summary

1. Introduction

Data is everywhere and plays a critical role in daily life. With such large volumes of information, ensuring that data is secure, accurate, and well-organized is essential.

Data governance provides the rules, processes, and tools to achieve this by controlling who can access data, tracking where it comes from, and ensuring it is protected.

Databricks Unity Catalog is a unified governance layer for data and AI within the Databricks platform. It allows organizations to centrally manage all assets — structured/unstructured data, machine learning models, notebooks, dashboards, and files — across any cloud.

2. Purpose

- Centralize governance for all data assets.
- Provide fine-grained access control.
- Enable auditing and data lineage.
- Simplify secure data sharing.

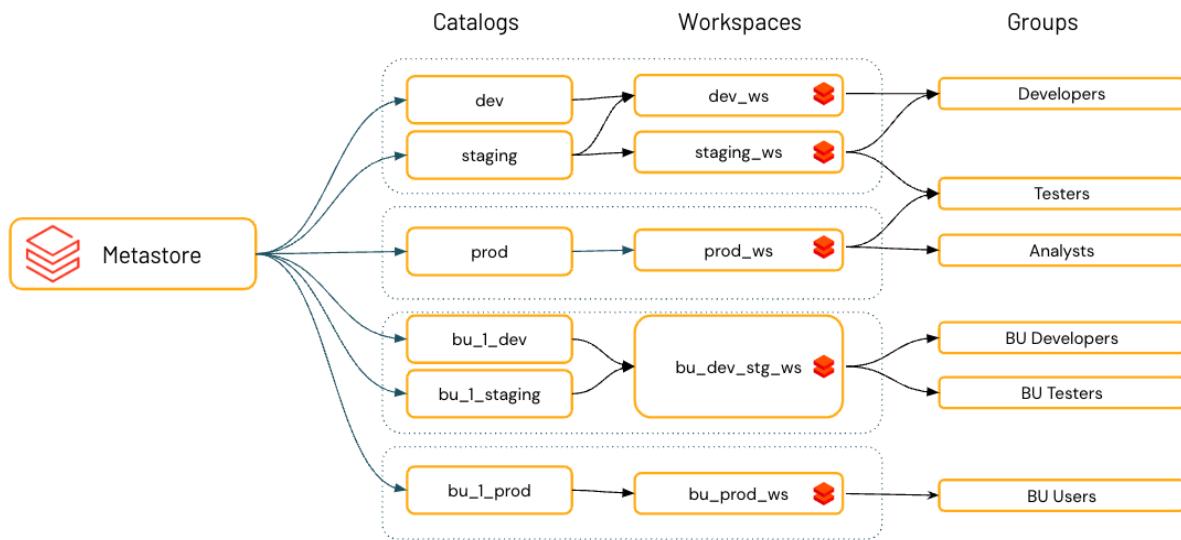
3. Architecture Overview

Unity Catalog uses a three-level namespace:

Metastore → Catalog → Schema → Tables / Views / Models

- Metastore – Top-level container.
- Catalog – Logical grouping of schemas.
- Schema – Collection of tables/views.
- Table/View – Actual data objects.

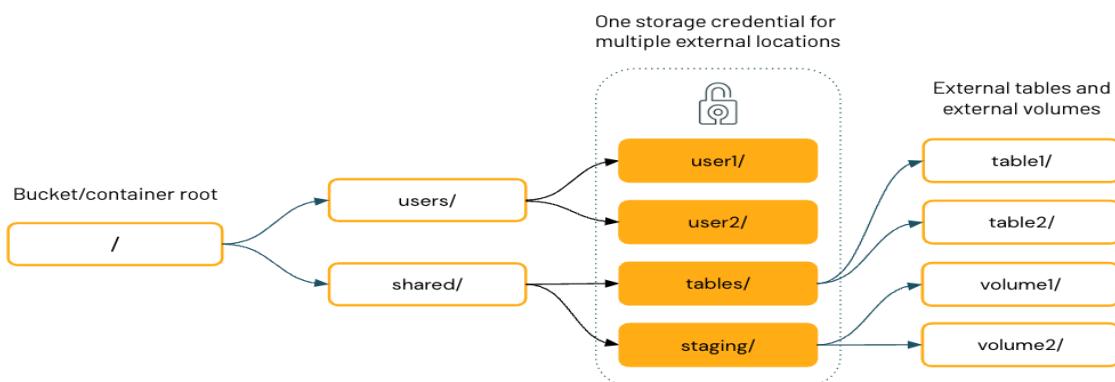
Architecture Diagram



4. Key Features

- **Centralized Policy Management** – One place to manage access for all workspaces.
- **ANSI SQL-Based Security** – Use familiar SQL syntax for granting/revoking permissions.
- **Built-in Auditing & Lineage** – Track data usage and dependencies.
- **Data Discovery** – Tag and search for data.
- **Managed & External Storage** – Store data in Databricks-managed or external sources.
- **Delta Sharing** – Securely share data across clouds/platforms.

Governance Overview

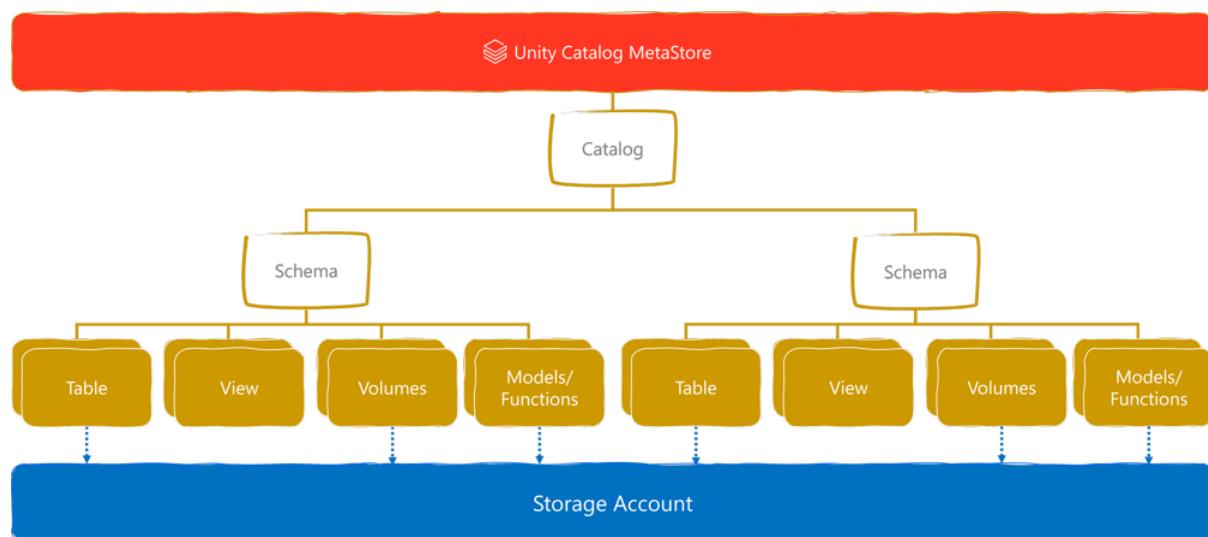


5. Access Control

Unity Catalog uses:

- **Privilege Inheritance** – Permissions set at a higher level flow down to lower levels.
- **Object Ownership Model** – The creator of an object is its owner by default.
- **Dynamic Views** – For row/column-level security.
- **External Locations & Storage Credentials** – To securely access data in external sources.

Catalog Explorer & Permissions



6. Setup Steps

Step 1 – Create Metastore

1. Navigate to Data → Unity Catalog → Metastores in Databricks.
2. Click Create Metastore, enter name, region, and storage location.
3. Assign to workspace.

Metastore Creation UI

The screenshot shows the Microsoft Azure Databricks Data Explorer interface. On the left, there's a sidebar with various icons and a 'Data' section containing a tree view of catalogs. A red box highlights the 'dev_procurement' catalog under 'dev_procurement'. Another red box highlights the 'prod_procurement' catalog under 'prod_procurement'. On the right, the 'Catalogs' page lists 24 catalogs. A third red box highlights the 'Owner' column, which lists email addresses for each catalog. The columns are 'Name', 'Created at', and 'Owner'.

Name	Created at	Owner
catalog_br	2023-05-22 16:23:51	biraj.roy@tredence.com
catalog_test01	2023-05-22 23:01:17	prerna.bhogaparam@tredence.com
catalog1	2023-05-20 15:20:56	chandrakanth.v@tredence.com
demo_forecasting	2023-05-15 16:27:58	chandrakanth.v@tredence.com
democatalog	2023-05-22 16:03:05	rehana.mushtaq@tredence.com
democatalog1	2023-05-31 12:40:39	rehana.mushtaq@tredence.com
dev_procurement	2023-05-18 10:57:22	maulik.dixit@tredence.com
hive_metastore		chandrakanth.v@tredence.com
lineage_cat	2023-05-19 19:12:21	chandrakanth.v@tredence.com
lineage_data	2023-05-19 19:08:22	chandrakanth.v@tredence.com
lineage_gen	2023-05-12 16:40:50	tre10002@tredence.com
main	2023-02-09 12:59:53	chandrakanth.v@tredence.com
poc	2023-05-20 11:38:44	chandrakanth.v@tredence.com
poc_catalog	2023-05-20 11:35:00	chandrakanth.v@tredence.com
poccatalog	2023-05-20 11:35:33	chandrakanth.v@tredence.com
prod_procurement	2023-05-18 11:26:23	maulik.dixit@tredence.com
qa_procurement	2023-05-18 11:25:53	maulik.dixit@tredence.com

Step 2 – Configure Storage Credentials

- Create a service principal with storage access.
- Add credentials in Unity Catalog.

Storage Credential Config

The screenshot shows the 'Mount your Hive Metastore as a catalog.' step. It displays a list of catalogs in the 'In my org' section, with 'hive_as_unity' selected. A red box highlights 'hive_as_unity'. Step 1 points to this section. Step 2 points to the 'Manage it alongside other catalogs.' section, which shows the catalog list again. Step 3 points to the 'Use a unified governance model.' section, which includes a 'Grant on hive_as_unity' dialog. The dialog shows 'All account users' selected in the 'Principals' dropdown. Under 'Privilege presets', 'Custom' is selected. In the 'Prerequisite' section, 'USE CATALOG' and 'USE SCHEMA' are checked. In the 'Read' section, 'EXECUTE' and 'READ VOLUME' are unchecked.

1 Mount your Hive Metastore as a catalog.

2 Manage it alongside other catalogs.

3 Use a unified governance model.

Step 3 – Create Catalog & Schema

CREATE CATALOG my_catalog;

USE CATALOG my_catalog;

CREATE SCHEMA my_schema;

USE SCHEMA my_schema;

Catalog & Schema Creation UI

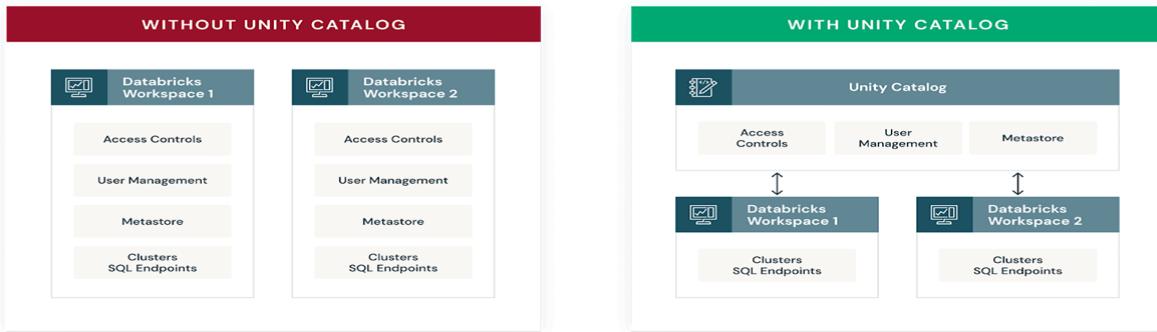
The screenshot shows the 'Create metastore' process in the Databricks UI. The current step is 'Assign to workspaces'. A table lists existing workspaces, with 'unitycatalogdemo' selected. At the bottom, there are 'Assign' and 'Skip' buttons.

Name	Status	Pricing tier	Resource group	Region	Created	Metastore
unitycatalogdemo	Running	Premium	sk03	westindia	today at 12:36 PM	-
ucdemo1	Running	Premium	sk02	centralindia	last Friday at 10:07 AM	metastore1
ucdemo8	Running	Premium	sk01	westindia	last Thursday at 7:31 PM	-
UnityCatalogDemo	Running	Premium	sk01	eastus	last Thursday at 11:10 AM	-
test00	Running	Premium	sk01	eastus	last Wednesday at 3:14 PM	-
Delta_Live_tables_2	Running	Premium	sk01	centralindia	08/17/2023	-
databricks_01	Running	Premium	sk01	eastus	08/17/2023	-

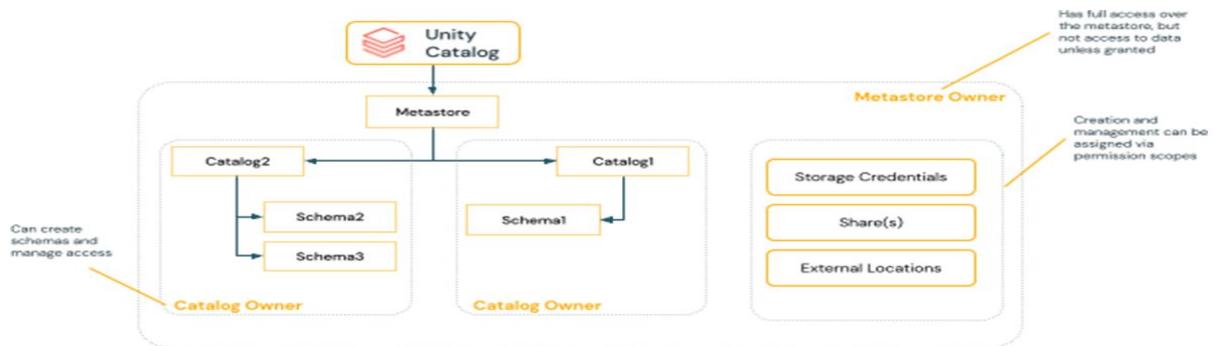
7. Best Practices

1. Use catalogs for isolation (prod/dev, sensitive/non-sensitive).
2. Group-based ownership instead of individual ownership.
3. Enable full audit logging.
4. Implement least privilege access.
5. Use Delta Sharing for external collaboration.

Delta Sharing Interface



Data Lineage View



8. Limitations

- Requires Databricks Runtime 11.3 LTS+.
- No bucketing or custom partition schemes.
- Limited R workload security.
- Naming restrictions on objects & columns.
- Workspace-level groups not supported in GRANT statements.

9. Conclusion

Unity Catalog is a centralized governance solution that unifies data security, discovery, auditing, and sharing across the Databricks platform. By following best practices and understanding its limitations, organizations can ensure secure, compliant, and efficient data management.