UNITY CATALOG

Unity Catalog in Azure Databricks is a unified governance solution for all your data and AI assets across workspaces in a Databricks account.
It lets you centrally manage permissions, data lineage, and auditing for files, tables, machine learning models, and notebooks all from one place.

Before Unity Catalog:

- Permissions were workspace-specific.
- Managing security across multiple workspaces meant duplicated configurations.
- No unified view of data lineage or governance.

With Unity Catalog:

- You have one central metastore for all workspaces in a region.
- Permissions and policies apply consistently across workspaces.
- Storage is linked to Azure Data Lake Storage Gen2 with secure, governed access.

Key Components :

a. Metastore

- The top-level container for all Unity Catalog objects.
- One per region per Databricks account.
- Linked to one default storage location (ADLS Gen2 container) for managed tables.

b. Catalog

- Organizational layer grouping schemas.
- Comparable to a “database” in traditional SQL systems.
- Example: finance, sales, engineering.

c. Schema

- Groups related tables, views, and functions within a catalog.
- Example: sales.orders schema containing tables for orders, customers.

d. Tables

- Managed tables: Data stored in the default storage location for the metastore.
- External tables: Data stored in external locations (ADLS, Blob Storage) linked to Unity Catalog.

e. Volumes

- Governed storage for unstructured or semi-structured files (images, PDFs, audio)

f. Delta Sharing

- Open protocol for secure cross-organization sharing of data without replication.

Naming Convention

Assets are referenced in three-part names:

catalog.schema.table

Metastore :

The metastore is the top-level container for metadata in Unity Catalog. It registers metadata about data and AI assets and the permissions that govern access to them. For a workspace to use Unity Catalog, it must have a Unity Catalog metastore attached. You should have one metastore for each region in which you have workspaces.

How It Works in Azure :

1. Metastore Creation:
    - Unity Catalog metastores are created per region in your Azure Databricks account.
    - You link it to an Azure Data Lake Storage Gen2 container for managed table storage.
2. Workspace Binding:
    - A single workspace can be bound to one Unity Catalog metastore at a time.
3. Data Access via ABAC & ACLs:
    - Uses Azure service principals or managed identities for secure access to data storage.
    - Permissions are granted with SQL GRANT statements or the Databricks UI.
4. Integration with Azure Services:
    - Azure AD: For user authentication & group management.
    - Azure Key Vault: For secret storage.
    - Azure Data Lake: For underlying storage.
    - Purview: For enterprise-wide cataloging and compliance.

5. Security & Access Control

- Object-level permissions – Control access to catalogs, schemas, tables, and views using SQL GRANT/REVOKE.
- Row-level security – Restrict records visible to users based on their identity or group.
- Column masking – Hide or mask sensitive fields like personal identifiers or financial data.
- Attribute-Based Access Control (ABAC) – Enforce policies using tags/labels for dynamic, context-based access.
- Ownership model – Each object has an owner who manages permissions and can delegate access rights.

```
CREATE CATALOG finance_catalog;

CREATE SCHEMA finance_catalog.sales_schema;

CREATE TABLE finance_catalog.sales_schema.transactions (
   transaction_id INT,
   customer_id INT,
   amount DECIMAL(10,2),
   transaction_date DATE
);

-- Insert

INSERT INTO finance_catalog.sales_schema.transactions
VALUES (1, 101, 250.75, '2025-08-18');


-- Query

SELECT * FROM finance_catalog.sales_schema.transactions; -- Grant SELECT permission to a user

GRANT SELECT ON TABLE finance_catalog.sales_schema.transactions TO `user@example.com`;


-- Revoke access

REVOKE SELECT ON TABLE finance_catalog.sales_schema.transactions FROM `user@example.com`;

-- Create a row filter policy

CREATE ROW FILTER filter_sales_amount
AS (amount > 100);


-- Apply policy

ALTER TABLE finance_catalog.sales_schema.transactions
SET ROW FILTER filter_sales_amount;
```

```
ALTER TABLE finance_catalog.sales_schema.transactions
OWNER TO `steward@example.com`;

-- Allow a user to create tables in the schema
GRANT CREATE ON SCHEMA finance_catalog.sales_schema TO `developer@example.com`;

-- Allow a group to use the schema
GRANT USAGE ON SCHEMA finance_catalog.sales_schema TO `finance_team`;
CREATE VIEW finance_catalog.sales_schema.recent_transactions AS
SELECT *
FROM finance_catalog.sales_schema.transactions
WHERE transaction_date >= DATE'2025-08-01';

DROP VIEW IF EXISTS finance_catalog.sales_schema.recent_transactions;
DROP TABLE IF EXISTS finance_catalog.sales_schema.transactions;
DROP SCHEMA IF EXISTS finance_catalog.sales_schema CASCADE;
DROP CATALOG IF EXISTS finance_catalog CASCADE;
```

Unity Catalog provides a centralized, fine-grained security model that governs data access across all workspaces in your Databricks account. By combining object-level permissions, row and column-level security, ABAC, and a clear ownership model, it ensures that only authorized users can view or modify data while maintaining compliance with privacy and regulatory standards. This layered approach makes it possible to balance accessibility with strong governance.