UNITY CATALOG

What's Unity Catalog?

- Unity Catalog is Databricks' unified governance solution for all data and AI assets. It provides centralized fine-grained access control, auditing, and data discovery across all Databricks workspaces.
- It provides a single point of administration for data access policies and simplifies data discovery and sharing.

Key features:

- Centralized metadata management
- Fine-grained access control (table, column level)
- Multi-workspace sharing & collaboration
- Support for SQL, Delta Lake, files, and more
- Auditing and lineage

Metastore

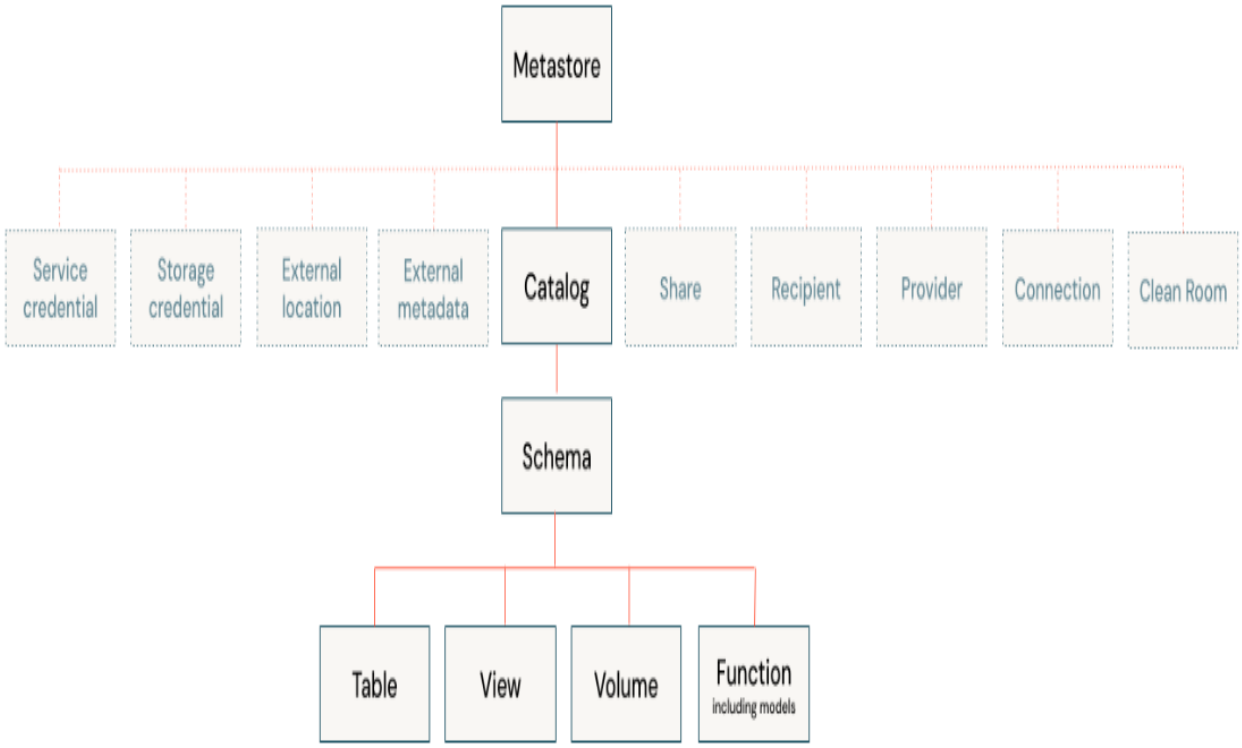
- The metastore is the top-level container for metadata in Unity Catalog.
- It registers metadata about data and AI assets and the permissions that govern access to them.
- For a workspace to use Unity Catalog, it must have a Unity Catalog metastore attached. You should have one metastore for each region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

The Unity Catalog object model

- In a Unity Catalog metastore, the three-level database object hierarchy consists of catalogs that contain schemas, which in turn contain data and AI objects, like tables and models.

    1) CATALOG
    2) SCHEMA
    3) TABLE

- This hierarchy is represented as a three-level namespace (catalog.schema.table-etc) when you reference tables, views, volumes, models, and function.



Catalogs :

- Catalogs are used to organize your data assets and are typically used as the top level in your data isolation scheme. Catalogs often mirror organizational units or software development lifecycle scopes

Schemas :

- Schemas (also known as databases) contain tables, views, volumes, AI models, and functions. Schemas organize data and AI assets into logical categories that are more granular than catalogs. Typically a schema represents a single use case, project, or team sandbox

Tables :

- Tables are collections of data organized by rows and columns. Tables can be either *managed*, with Unity Catalog managing the full lifecycle of the table, or *external*, with Unity Catalog managing access to the data from within Databricks, but not managing access to the data in cloud storage from other clients.

Views - are saved queries against one or more tables.

Volumes - represent logical volumes of data in cloud object storage. You can use volumes to store, organize, and access files in any format, including structured, semi-structured, and unstructured data. Typically they are used for non-tabular data.

Level	Description	Example
Catalog	Top-level container for schemas (similar to a database)	sales
Schema	Sub-container inside catalog (similar to schema or database)	marketing
Table	Actual table or view inside schema	customer_data

A fully qualified table name looks like:

catalog_name.schema_name.table_name

Example :

sales.marketing.customer_data

Creating Unity Catalog Objects

1) Create a Catalog

```
CREATE CATALOG sales;
```

This creates a new catalog named sales.

2) Create a Schema in the Catalog

```
CREATE SCHEMA sales.marketing;
```

This creates a schema named marketing inside the catalog sales.

3) Create a Table inside the Schema

```
CREATE TABLE sales.marketing.customer_data (
  customer_id INT,
  name STRING,
  email STRING,
  purchase_amount DOUBLE
);
```

Creates a table inside the schema and catalog.

4) Insert some data

```
INSERT INTO sales.marketing.customer_data
VALUES (1, 'Alice', 'alice@email.com', 120.50),
       (2, 'Bob', 'bob@email.com', 85.00);
```

5) Query the data

```
SELECT * FROM sales.marketing.customer_data;
```

6) Verifying & Managing

SHOW CATALOGS; → Check catalogs

SHOW SCHEMAS IN sales; → Check schemas in a catalog

SHOW TABLES IN sales.marketing; → Check tables in a schema