Azure Databricks Coding Challenge

Setting up Unity Catalog in Databricks

Unity Catalog is Databricks' unified governance solution that helps you securely manage data and AI assets across all workspaces. This challenge walks you through the **end-to-end process of enabling Unity Catalog in Azure Databricks**, from prerequisites to validation.

Step 1: Prerequisites

Before you begin, make sure you have:

- **Databricks account admin** rights (workspace admin is not enough).
- **Azure Active Directory (AAD)** permissions to register applications and assign roles.
- A **Premium or Enterprise Databricks plan** (Unity Catalog is not available on Standard).

Step 2: Create a Unity Catalog Metastore

The **metastore** is the central governance layer that stores metadata for schemas, tables, and permissions.

1. Log in to the **Databricks Account Console** as an account admin.
2. Navigate to **Data** → **Metastores** in the left-hand panel.
3. Click **Create Metastore**.
4. Provide the required details:
    - **Name**: e.g., uc_metastore
    - **Region**: must match your Azure region

Storage Root: an ADLS Gen2 container, e.g.

abfss://ucroot@yourstorage.dfs.core.windows.net/

5. Assign a **Metastore Admin** who will manage access and governance.

Step 3: Configure Storage (External Location)

Unity Catalog requires a **root storage location** for managed tables and optional external data.

1. In the **Azure Portal**, create a new **ADLS Gen2 Storage Account**.
2. Inside it, create a container (e.g., uc-root).
3. Register a **Service Principal (SPN)** in Azure AD and note the:
    - Client ID
    - Client Secret
    - Tenant ID
4. Assign the **Storage Blob Data Contributor** role to the SPN on your ADLS account.
5. Back in Databricks, go to **External Locations** → **Create Location** and point to your ADLS path.

Step 4: Attach Metastore to Workspace

Now, link the newly created metastore with your Databricks workspace.

1. In the **Account Console**, go to **Workspaces**.
2. Select your workspace.
3. Click **Assign Metastore**, and choose uc_metastore.

From this point on, the workspace is governed by Unity Catalog.

Step 5: Create a 3-Level Namespace

Unity Catalog introduces a **three-level namespace**:

catalog.schema.table

Example in a Databricks notebook (SQL mode):

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

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

Step 6: Set Permissions (Data Governance)

Unity Catalog uses SQL-style **GRANT** and **REVOKE** commands for fine-grained access control.

```
-- Give analysts read-only access
GRANT SELECT ON TABLE sales_catalog.retail.orders TO `analyst_group`;

-- Give data engineers write access
GRANT MODIFY ON TABLE sales_catalog.retail.orders TO `data_engineers`;
```

Step 7: Validate Configuration

To confirm your workspace is attached to a Unity Catalog metastore, run:

```
SELECT CURRENT_METASTORE();
```

If a metastore ID is returned, Unity Catalog is successfully enabled.

Optional: Enable Unity Catalog During Workspace Creation

When creating a new workspace, you can enable Unity Catalog upfront:

1. Toggle **Enable Unity Catalog** during setup.

2. Select the metastore to associate with the workspace.

3. Click **Enable**, then provide configuration details and save.