Task-1 : Creating a meta Store from the admin console

Task-2: Create Department Specific Catalogs

- CREATE CATALOG Marketing;
- CREATE CATALOG Engineering;
- CREATE CATALOG Operations; Task 3: Create Schemas

  for each Catalog

- CREATE SCHEMA Marketing.ads_data;
- CREATE SCHEMA Marketing.customer_data; For

  Engineering Catalog:

- CREATE SCHEMA Engineering.projects;
- CREATE SCHEMA Engineering.development_data; For

  Operations Catalog:

- CREATE SCHEMA Operations.logistics_data;
- CREATE SCHEMA Operations.supply_chain; Task 4 & 5:

  Creating Tables and loading dataset:

For Marketing Catalog:

- CREATE TABLE Marketing.ads_data.ad_details (ad_id
  int, impressions int,
  clicks int, cost_per_click
  double);
- CREATE TABLE
  Marketing.customer_data.customer_detail(cust_id int,
  ad_id int);

For Engineering Catalog:

-
  ```
  CREATE TABLE Engineering.projects.project_data(
  project_id int, project_name string);
  ```

-
  ```
  CREATE TABLE Engineering.projects.development_data(
  dev_id int, project_id int, start_data date, end_date date);
  ```

For Operations Catalog:

-
  ```
  CREATE TABLE Operations.logistics_data.logistics (
  shipment_id int, status string);
  ```

-
  ```
  CREATE TABLE
  Operations.supply_chain.supply_chain_data( Id_no int,
  origin string, destination string, shipment_id int);
  ```

Inserting Data:

-
  ```
  INSERT INTO Marketing.ads_data.ad_details (ad_id,
  impressions, clicks, cost_per_click)
  VALUES
  (1, 1000, 700, 0.25),
  (2, 1500, 750, 0.30),
  (3, 1200, 630, 0.20);
  ```

-
  ```
  INSERT INTO Marketing.customer_data.customer_detail
  (cust_id, ad_id)
  VALUES
  (101, 1),
  (102, 2),
  (103, 3);
  ```

-
  ```
  INSERT INTO Engineering.projects.project_data
  (project_id, project_name)
  VALUES
  (1, 'Website Redesign'),
  (2, 'Mobile App Development'),
  (3, 'Database Optimization');
  ```

-
  ```
  INSERT INTO Engineering.projects.development_data
  (dev_id, project_id, start_data, end_date)
  VALUES
  (1, 1, '2024-01-01', '2024-06-30'),
  ```

```
(2, 2, '2024-03-15', '2024-12-31'),
(3, 3, '2024-02-01', '2024-04-30');
```

-
  ```
  INSERT INTO Operations.logistics_data.logistics
  (shipment_id, status)
  VALUES
  (1001, 'Delivered'),
  (1002, 'In Transit'),
  (1003, 'Processing');
  ```

-
  ```
  INSERT INTO
  Operations.supply_chain.supply_chain_data (Id_no,
  origin, destination, shipment_id)
  VALUES
  (1, 'Chennai', 'Bangalore', 1001),
  (2, 'Chennai', 'Hyderabad', 1002),
  (3, 'Chennai', 'Mumbai', 1003);
  ```

Task 6: Create Roles and Grant Access:

CREATE ROLE marketing_role;

CREATE ROLE engineering_role;

CREATE ROLE operations_role; Task 7:

Configure Fine Grained Access:

For Marketing role:

GRANT SELECT ON TABLE Marketing.customer_data.customer_detail TO marketing_role;

GRANT SELECT ON TABLE Marketing.ads_data.ad_details TO marketing_role;

For Engineering role:

GRANT SELECT ON TABLE Engineering.projects.project_data TO engineering_role; GRANT

SELECT ON TABLE Engineering.projects.development_data TO engineering_role;

For Operations role:

```
GRANT SELECT ON TABLE operations.logistics_data.logistics TO operations_role; GRANT

SELECT ON TABLE operations.supply_chain.supply_chain_data TO operations_role; Task 8:
```

Enable and Explore Data Lineage:

Navigate to the databricks UI to Catalog Explorer to check the lineage of the tables we created

Task 9: Monitor Data Access and Modifications:

In the Admin Console, we can view the Audit logs for the operations performed.

Task 10: Explore Metadata in unity catalog:

For Marketing Tables:

```
DESCRIBE TABLE Marketing.ads_data.ad_details;

DESCRIBE TABLE Marketing.customer_data.customer_detail;

SELECT COUNT(*) FROM marketing.ads_data.ad_details;

SELECT COUNT(*) FROM marketing.customer_data.customer_detail;
```

For Engineering Tables:

```
DESCRIBE TABLE Engineering.projects.project_data;

DESCRIBE TABLE Engineering.projects.development_data;

SELECT COUNT(*) FROM engineering.projects.project_data;

SELECT COUNT(*) FROM engineering.projects.development_data;
```

For Operations Tables:

```
DESCRIBE TABLE Operations.logistics_data.logistics;

DESCRIBE TABLE Operations.supply_chain.supply_chain_data;

SELECT COUNT(*) FROM Operations.logistics_data.logistics;

SELECT COUNT(*) FROM Operations.supply_chain.supply_chain_data;
```