**Part 1: Setting Up the Environment**

**Task 1: Create a Metastore**

CREATE CATALOG IF NOT EXISTS my\_catalog;

USE CATALOG my\_catalog;

CREATE SCHEMA IF NOT EXISTS my\_schema;

**Task 2: Create Department-Specific Catalogs**

**Marketing**

CREATE CATALOG IF NOT EXISTS Marketing;

**Engineering**

CREATE CATALOG IF NOT EXISTS Engineering;

**Operations**

CREATE CATALOG IF NOT EXISTS Operations;

**Permissions(Example)**

GRANT ALL PRIVILEGES ON CATALOG Marketing TO `marketing\_team`;

GRANT SELECT, MODIFY ON CATALOG Engineering TO `engineering\_team`;

GRANT SELECT ON CATALOG Operations TO `operations\_team`;

**Task 3: Create Schemas for Each Department**

**Marketing Catalog:**

USE CATALOG Marketing;

CREATE SCHEMA IF NOT EXISTS ads\_data;

CREATE SCHEMA IF NOT EXISTS customer\_data;

**Engineering Catalog:**

USE CATALOG Engineering;

CREATE SCHEMA IF NOT EXISTS projects;

CREATE SCHEMA IF NOT EXISTS development\_data;

**Operations Catalog:**

USE CATALOG Operations;

CREATE SCHEMA IF NOT EXISTS logistics\_data;

CREATE SCHEMA IF NOT EXISTS supply\_chain;

**Verify schemas**

SHOW SCHEMAS IN CATALOG Marketing;

SHOW SCHEMAS IN CATALOG Engineering;

SHOW SCHEMAS IN CATALOG Operations;

**Permissions:**

GRANT USAGE ON SCHEMA Marketing.ads\_data TO `marketing\_team`;

GRANT USAGE ON SCHEMA Marketing.customer\_data TO `marketing\_team`;

**Part 2: Loading Data and Creating Tables**

**Task 4: Prepare Datasets**

**Sample Dataset:**

**Marketing - Ads Data**

ad\_id,impressions,clicks,cost\_per\_click

1,1000,50,0.5

2,1500,60,0.4

3,2000,70,0.6

4,1200,45,0.55

**Engineering - Projects Data**

project\_id,project\_name,start\_date,end\_date

101,Website Redesign,2023-01-15,2023-04-15

102,Mobile App Development,2023-02-10,2023-05-30

103,Database Migration,2023-03-05,2023-06-20

104,AI Model Development,2023-04-01,2023-07-15

**Operations - Logistics Data**

shipment\_id,origin,destination,status

201,New York,Los Angeles,In Transit

202,Chicago,Houston,Delivered

203,San Francisco,Seattle,Pending

204,Miami,Dallas,In Transit

**Creating CSV Files**

import pandas as pd

marketing\_data = {

'ad\_id': [1, 2, 3, 4],

'impressions': [1000, 1500, 2000, 1200],

'clicks': [50, 60, 70, 45],

'cost\_per\_click': [0.5, 0.4, 0.6, 0.55]

}

marketing\_df = pd.DataFrame(marketing\_data)

marketing\_df.to\_csv('marketing\_ads\_data.csv', index=False)

engineering\_data = {

'project\_id': [101, 102, 103, 104],

'project\_name': ['Website Redesign', 'Mobile App Development', 'Database Migration', 'AI Model Development'],

'start\_date': ['2023-01-15', '2023-02-10', '2023-03-05', '2023-04-01'],

'end\_date': ['2023-04-15', '2023-05-30', '2023-06-20', '2023-07-15']

}

engineering\_df = pd.DataFrame(engineering\_data)

engineering\_df.to\_csv('engineering\_projects\_data.csv', index=False)

operations\_data = {

'shipment\_id': [201, 202, 203, 204],

'origin': ['New York', 'Chicago', 'San Francisco', 'Miami'],

'destination': ['Los Angeles', 'Houston', 'Seattle', 'Dallas'],

'status': ['In Transit', 'Delivered', 'Pending', 'In Transit']

}

operations\_df = pd.DataFrame(operations\_data)

operations\_df.to\_csv('operations\_logistics\_data.csv', index=False)

**Upload**

databricks fs cp local\_path/marketing\_ads\_data.csv dbfs:/mnt/your\_path/marketing\_ads\_data.csv

**Task 5: Create Tables from the Datasets**

**Marketing - ads\_data Table:**

USE CATALOG Marketing;

USE SCHEMA ads\_data;

CREATE TABLE ads\_data (

ad\_id INT,

impressions INT,

clicks INT,

cost\_per\_click FLOAT

)

USING csv

OPTIONS (path 'dbfs:/mnt/your\_path/marketing\_ads\_data.csv', header = true);

**Engineering - projects Table:**

USE CATALOG Engineering;

USE SCHEMA projects;

CREATE TABLE projects (

project\_id INT,

project\_name STRING,

start\_date DATE,

end\_date DATE

)

USING csv

OPTIONS (path 'dbfs:/mnt/your\_path/engineering\_projects\_data.csv', header = true);

**Operations - logistics\_data Table:**

USE CATALOG Operations;

USE SCHEMA logistics\_data;

CREATE TABLE logistics\_data (

shipment\_id INT,

origin STRING,

destination STRING,

status STRING

)

USING csv

OPTIONS (path 'dbfs:/mnt/your\_path/operations\_logistics\_data.csv', header = true);

**Verifying the Tables**

SHOW TABLES IN Marketing.ads\_data;

SHOW TABLES IN Engineering.projects;

SHOW TABLES IN Operations.logistics\_data;

**Part 3: Data Governance Capabilities Data Access Control**

**Task 6: Create Roles and Grant Access**

CREATE ROLE IF NOT EXISTS marketing\_role;

CREATE ROLE IF NOT EXISTS engineering\_role;

CREATE ROLE IF NOT EXISTS operations\_role;

**Grant Access**

**Marketing Role:**

GRANT USAGE ON CATALOG Marketing TO ROLE marketing\_role;

GRANT USAGE ON SCHEMA Marketing.ads\_data TO ROLE marketing\_role;

GRANT USAGE ON SCHEMA Marketing.customer\_data TO ROLE marketing\_role;

**Engineering Role:**

GRANT USAGE ON CATALOG Engineering TO ROLE engineering\_role;

GRANT USAGE ON SCHEMA Engineering.projects TO ROLE engineering\_role;

GRANT USAGE ON SCHEMA Engineering.development\_data TO ROLE engineering\_role;

**Operations Role:**

GRANT USAGE ON CATALOG Operations TO ROLE operations\_role;

GRANT USAGE ON SCHEMA Operations.logistics\_data TO ROLE operations\_role;

GRANT USAGE ON SCHEMA Operations.supply\_chain TO ROLE operations\_role;

**Assign Users to Roles:**

GRANT ROLE marketing\_role TO `user1@databricks.com`;

GRANT ROLE engineering\_role TO `user2@databricks.com`;

GRANT ROLE operations\_role TO `user3@databricks.com`;

**Verifying Roles and Permissions**

SHOW ROLES;

SHOW GRANTS ON CATALOG Marketing;

SHOW GRANTS ON CATALOG Engineering;

SHOW GRANTS ON CATALOG Operations;

**Task 7: Configure Fine-Grained Access Control**

**Marketing Department:**

USE CATALOG Marketing;

GRANT USAGE ON SCHEMA customer\_data TO ROLE marketing\_role;

GRANT SELECT ON TABLE Marketing.customer\_data TO ROLE marketing\_role;

REVOKE SELECT ON TABLE Marketing.ads\_data FROM ROLE marketing\_role;

**Engineering Department:**

USE CATALOG Engineering;

GRANT USAGE ON SCHEMA projects TO ROLE engineering\_role;

GRANT SELECT ON TABLE Engineering.projects TO ROLE engineering\_role;

REVOKE SELECT ON TABLE Engineering.development\_data FROM ROLE engineering\_role;

GRANT SELECT(ad\_id, impressions, clicks) ON TABLE Marketing.ads\_data TO ROLE marketing\_role;

REVOKE SELECT(cost\_per\_click) ON TABLE Marketing.ads\_data FROM ROLE marketing\_role;

**Assign Specific Users to Roles:**

GRANT ROLE marketing\_role TO `marketing\_user1@databricks.com`;

GRANT ROLE engineering\_role TO `engineering\_user1@databricks.com`;

**Verify**

SHOW GRANTS ON TABLE Marketing.customer\_data;

SHOW GRANTS ON TABLE Engineering.projects;

SHOW GRANTS ON TABLE Marketing.ads\_data;

SHOW POLICIES ON TABLE Marketing.customer\_data;

**Data Lineage**

**Task 8: Enable and Explore Data Lineage**

**Marketing:**

USE CATALOG Marketing;

USE SCHEMA ads\_data;

SELECT ad\_id, SUM(impressions) AS total\_impressions, SUM(clicks) AS total\_clicks

FROM ads\_data

GROUP BY ad\_id;

**Engineering:**

USE CATALOG Engineering;

USE SCHEMA projects;

SELECT p.project\_id, p.project\_name, d.development\_status

FROM projects p

JOIN development\_data d

ON p.project\_id = d.project\_id;

**Operations:**

USE CATALOG Operations;

USE SCHEMA logistics\_data;

SELECT status, COUNT(shipment\_id) AS total\_shipments

FROM logistics\_data

WHERE origin = 'New York'

GROUP BY status;

**Data Audit**

**Task 9: Monitor Data Access and Modifications**

CREATE TABLE audit\_log (

log\_id INT IDENTITY(1,1),

user\_name VARCHAR(100),

action\_type VARCHAR(50),

table\_name VARCHAR(100),

query\_text VARCHAR(MAX),

action\_time DATETIME DEFAULT CURRENT\_TIMESTAMP

);

CREATE TRIGGER trg\_ads\_data\_insert

ON ads\_data

AFTER INSERT

AS

BEGIN

DECLARE @user\_name VARCHAR(100) = SYSTEM\_USER; -- Gets the current user

DECLARE @query\_text VARCHAR(MAX) = 'INSERT'; -- Logs the action type

-- Insert action into audit log

INSERT INTO audit\_log (user\_name, action\_type, table\_name, query\_text)

VALUES (@user\_name, 'INSERT', 'ads\_data', @query\_text);

END;

CREATE TRIGGER trg\_ads\_data\_update

ON ads\_data

AFTER UPDATE

AS

BEGIN

DECLARE @user\_name VARCHAR(100) = SYSTEM\_USER;

DECLARE @query\_text VARCHAR(MAX) = 'UPDATE';

INSERT INTO audit\_log (user\_name, action\_type, table\_name, query\_text)

VALUES (@user\_name, 'UPDATE', 'ads\_data', @query\_text);

END;

CREATE TRIGGER trg\_ads\_data\_delete

ON ads\_data

AFTER DELETE

AS

BEGIN

DECLARE @user\_name VARCHAR(100) = SYSTEM\_USER;

DECLARE @query\_text VARCHAR(MAX) = 'DELETE';

INSERT INTO audit\_log (user\_name, action\_type, table\_name, query\_text)

VALUES (@user\_name, 'DELETE', 'ads\_data', @query\_text);

END;

**-- View all actions logged in the audit log**

SELECT \*

FROM audit\_log

ORDER BY action\_time DESC;

**Data Discovery**

**Task 10: Explore Metadata in Unity Catalog**

SHOW CATALOGS;

SHOW SCHEMAS IN < Marketing>;

SHOW TABLES IN <catalog\_name>.<schema\_name>;

DESCRIBE <catalog\_name>.<schema\_name>.<table\_name>;

SELECT COUNT(\*) AS num\_rows

FROM <catalog\_name>.<schema\_name>.<table\_name>;

**Descriptions:**

ALTER CATALOG <catalog\_name>

SET COMMENT 'Description of the catalog';

ALTER SCHEMA <catalog\_name>.<schema\_name>

SET COMMENT 'Description of the schema';

ALTER TABLE <catalog\_name>.<schema\_name>.<table\_name>

SET COMMENT 'Description of the table';

ALTER TABLE <catalog\_name>.<schema\_name>.<table\_name>

SET TBLPROPERTIES ('property\_name' = 'property\_value');