**Mini Project: Data Governance Using Unity Catalog - Advanced Capabilities**

**Task 1: Set Up Unity Catalog Objects with Multiple Schemas**

CREATE CATALOG finance\_data\_catalog;

2. **Create Multiple Schemas**:

Transaction data:

CREATE SCHEMA finance\_data\_catalog.transaction\_data;

Customer data:

CREATE SCHEMA finance\_data\_catalog.customer\_data;

3. Create Tables in Each Schema:

Transaction\_data table:

CREATE TABLE finance\_data\_catalog.transaction\_data.transactions (

TransactionID STRING,

CustomerID STRING,

TransactionAmount DECIMAL(10, 2),

TransactionDate DATE

);

Customer\_data table:

CREATE TABLE finance\_data\_catalog.customer\_data.customers (

CustomerID STRING,

CustomerName STRING,

Email STRING,

Country STRING

);

**Task 2: Data Discovery Across Schemas**

1. Explore Metadata:

SHOW TABLES IN finance\_data\_catalog.transaction\_data;

SHOW TABLES IN finance\_data\_catalog.customer\_data;

1. Data Profiling:

Transaction\_data profile:

SELECT AVG(TransactionAmount), MAX(TransactionAmount), MIN(TransactionAmount)

FROM finance\_data\_catalog.transaction\_data.transactions;

Customer\_data profile:

SELECT Country, COUNT(CustomerID) AS CustomerCount

FROM finance\_data\_catalog.customer\_data.customers

GROUP BY Country;

Tagging Sensitive Data:

ALTER TABLE finance\_data\_catalog.customer\_data.customers ALTER COLUMN Email SET TAG 'sensitive';

ALTER TABLE finance\_data\_catalog.transaction\_data.transactions ALTER COLUMN TransactionAmount SET TAG 'sensitive';

**Task 3: Implement Data Lineage and Auditing**

Track Data Lineage:

CREATE OR REPLACE VIEW finance\_data\_catalog.transaction\_customer\_view AS

SELECT t.TransactionID, t.TransactionAmount, t.TransactionDate, c.CustomerName, c.Email, c.Country

FROM finance\_data\_catalog.transaction\_data.transactions t

JOIN finance\_data\_catalog.customer\_data.customers c

ON t.CustomerID = c.CustomerID;

**Task 4: Access Control and Permissions**

1. Set Up Roles and Groups:

CREATE GROUP DataEngineers;

CREATE GROUP DataAnalysts;

Assign permissions:

Data Engineers:

GRANT ALL PRIVILEGES ON SCHEMA finance\_data\_catalog.transaction\_data TO DataEngineers;

GRANT ALL PRIVILEGES ON SCHEMA finance\_data\_catalog.customer\_data TO DataEngineers;

Read-only for DataAnalysts on customer\_data and limited access on transaction\_data:

GRANT SELECT ON SCHEMA finance\_data\_catalog.customer\_data TO DataAnalysts;

GRANT SELECT (TransactionID, CustomerID, TransactionDate) ON finance\_data\_catalog.transaction\_data.transactions TO DataAnalysts;

1. Row-Level Security:

Implement row-level security for high-value transactions:

CREATE ROW ACCESS POLICY high\_value\_transactions

ON finance\_data\_catalog.transaction\_data.transactions

AS

CASE

WHEN current\_user() IN ('authorized\_user@example.com') AND TransactionAmount > 1000 THEN TRUE

ELSE FALSE

END;

**Task 5: Data Governance Best Practices**

1. Create Data Quality Rules:

Transaction amount should be non-negative:

ALTER TABLE finance\_data\_catalog.transaction\_data.transactions

ADD CONSTRAINT chk\_positive\_transaction\_amount CHECK (TransactionAmount >= 0);

Email should follow correct format:

ALTER TABLE finance\_data\_catalog.customer\_data.customers

ADD CONSTRAINT chk\_valid\_email CHECK (Email LIKE '%\_@\_%.\_%');

**Task 6: Data Lifecycle Management:**

**1. Implement Time Travel:**

**Access historical versions using Delta Time Travel:**

SELECT \* FROM finance\_data\_catalog.transaction\_data.transactions VERSION AS OF 5;

Restore a previous version:

RESTORE TABLE finance\_data\_catalog.transaction\_data.transactions VERSION AS OF 5;

**2. Run a Vacuum Operation:**

**Vacuum the Delta tables to clean up old files:**

VACUUM finance\_data\_catalog.transaction\_data.transactions;

VACUUM finance\_data\_catalog.customer\_data.customers;

**Mini Project: Advanced Data Governance and Security Using Unity Catalog**

1.Create a New Catalog:

CREATE CATALOG corporate\_data\_catalog;

2.Create Schemas for Each Department:

CREATE SCHEMA corporate\_data\_catalog.sales\_data;

CREATE SCHEMA corporate\_data\_catalog.hr\_data;

CREATE SCHEMA corporate\_data\_catalog.finance\_data;

3.Create Tables in Each Schema:

Sales Table:

CREATE TABLE corporate\_data\_catalog.sales\_data.sales (

SalesID INT,

CustomerID INT,

SalesAmount DECIMAL(10, 2),

SalesDate DATE

);

HR Employees Table:

CREATE TABLE corporate\_data\_catalog.hr\_data.employees (

EmployeeID INT,

EmployeeName STRING,

Department STRING,

Salary DECIMAL(10, 2)

);

Finance Invoices Table:

CREATE TABLE corporate\_data\_catalog.finance\_data.invoices (

InvoiceID INT,

VendorID INT,

InvoiceAmount DECIMAL(10, 2),

PaymentDate DATE

);

**Task 2: Enable Data Discovery for Cross-Departmental Data**

1. Search for Tables Across Departments:

SHOW TABLES IN corporate\_data\_catalog.sales\_data;

SHOW TABLES IN corporate\_data\_catalog.hr\_data;

SHOW TABLES IN corporate\_data\_catalog.finance\_data;

2.Tag Sensitive Information:

ALTER TABLE corporate\_data\_catalog.hr\_data.employees

SET TAG 'Sensitive' ON COLUMN Salary;

ALTER TABLE corporate\_data\_catalog.finance\_data.invoices

SET TAG 'Sensitive' ON COLUMN InvoiceAmount;

3. **Data Profiling:**

**Sales Trends:**

SELECT AVG(SalesAmount), MIN(SalesAmount), MAX(SalesAmount)

FROM corporate\_data\_catalog.sales\_data.sales;

Employee Salary Distribution:

SELECT AVG(Salary), MAX(Salary)

FROM corporate\_data\_catalog.hr\_data.employees;

Financial Transactions:

SELECT AVG(InvoiceAmount), MIN(InvoiceAmount), MAX(InvoiceAmount)

FROM corporate\_data\_catalog.finance\_data.invoices;

**Task 3: Implement Data Lineage and Data Auditing**

1.Track Data Lineage:

CREATE TABLE corporate\_data\_catalog.reports.sales\_finance\_report AS

SELECT s.SalesID, s.CustomerID, s.SalesAmount, s.SalesDate,

f.InvoiceID, f.InvoiceAmount, f.PaymentDate

FROM corporate\_data\_catalog.sales\_data.sales s

JOIN corporate\_data\_catalog.finance\_data.invoices f

ON s.CustomerID = f.VendorID;

**Task 4: Data Access Control and Security**

CREATE GROUP SalesTeam;

CREATE GROUP FinanceTeam;

CREATE GROUP HRTeam;

**Grant Permissions:**

SalesTeam Access:

GRANT SELECT ON SCHEMA corporate\_data\_catalog.sales\_data TO SalesTeam;

FinanceTeam Access:

GRANT SELECT ON SCHEMA corporate\_data\_catalog.sales\_data TO FinanceTeam;

GRANT SELECT, INSERT, UPDATE ON SCHEMA corporate\_data\_catalog.finance\_data TO FinanceTeam;

HRTeam Access:

GRANT SELECT, UPDATE ON SCHEMA corporate\_data\_catalog.hr\_data TO HRTeam;

**2. Implement Column-Level Security:**

**Create a Restricted View for HR Salary:**

CREATE VIEW corporate\_data\_catalog.hr\_data.salary\_restricted AS

SELECT EmployeeID, EmployeeName, Department

FROM corporate\_data\_catalog.hr\_data.employees;

**Grant Access to HRTeam:**

GRANT SELECT ON VIEW corporate\_data\_catalog.hr\_data.salary\_restricted TO HRTeam;

**3. Row-Level Security:**

**Create View for Sales Representatives:**

CREATE VIEW corporate\_data\_catalog.sales\_data.sales\_rep\_view AS

SELECT \*

FROM corporate\_data\_catalog.sales\_data.sales

WHERE SalesRepID = current\_user();

**Grant Access to Specific Sales Representative:**

GRANT SELECT ON VIEW corporate\_data\_catalog.sales\_data.sales\_rep\_view TO specific\_sales\_rep;

**Task 5: Data Governance Best Practices**

**1. Define Data Quality Rules:**

**Ensure Sales Amounts Are Positive:**

SELECT \* FROM corporate\_data\_catalog.sales\_data.sales

WHERE SalesAmount < 0;

**Ensure Employee Salaries Are Positive:**

SELECT \* FROM corporate\_data\_catalog.hr\_data.employees

WHERE Salary <= 0;

Ensure Invoice Amounts Are Valid:

SELECT \* FROM corporate\_data\_catalog.finance\_data.invoices

WHERE InvoiceAmount <= 0;

**2.Apply Time Travel for Data Auditing:**

**Access Historical Data Using Time Travel:**

SELECT \* FROM corporate\_data\_catalog.finance\_data.invoices

VERSION AS OF TIMESTAMP '2024-09-15';

**Task 6: Optimize and Clean Up Delta Tables:**

**1.Optimize Delta Tables:**

OPTIMIZE corporate\_data\_catalog.sales\_data.sales;

OPTIMIZE corporate\_data\_catalog.finance\_data.invoices;

**2.Vacuum Delta Tables:**

VACUUM corporate\_data\_catalog.sales\_data.sales;

VACUUM corporate\_data\_catalog.finance\_data.invoices;

**Mini Project: Building a Secure Data Platform with Unity Catalog**

**Task 1: Set Up Unity Catalog for Multi-Domain Data Management**

**1.Create a New Catalog:**

CREATE CATALOG enterprise\_data\_catalog;

**2.Create Domain-Specific Schemas:**

CREATE SCHEMA enterprise\_data\_catalog.marketing\_data;

CREATE SCHEMA enterprise\_data\_catalog.operations\_data;

CREATE SCHEMA enterprise\_data\_catalog.it\_data;

**3. Create Tables in Each Schema:**

* **Marketing Campaigns Table:**

CREATE TABLE enterprise\_data\_catalog.marketing\_data.campaigns (

CampaignID INT,

CampaignName STRING,

Budget DOUBLE,

StartDate DATE

);

* **Operations Orders Table:**

CREATE TABLE enterprise\_data\_catalog.operations\_data.orders (

OrderID INT,

ProductID INT,

Quantity INT,

ShippingStatus STRING

);

* **IT Incidents Table:**

CREATE TABLE enterprise\_data\_catalog.it\_data.incidents (

IncidentID INT,

ReportedBy STRING,

IssueType STRING,

ResolutionTime DOUBLE

);

**Task 2: Data Discovery and Classification**

**1. Search for Data Across Schemas:**

**List all tables in the catalog:**

SHOW TABLES IN enterprise\_data\_catalog;

**Search for specific columns across schemas:**

SELECT \*

FROM enterprise\_data\_catalog.INFORMATION\_SCHEMA.COLUMNS

WHERE column\_name = 'Budget';

**2. Tag Sensitive Information:**

**Tag the Budget column in the marketing table:**

ALTER TABLE enterprise\_data\_catalog.marketing\_data.campaigns

SET TAGS ('Budget' = 'Sensitive');

**Tag the ResolutionTime column in the IT table:**

ALTER TABLE enterprise\_data\_catalog.it\_data.incidents

SET TAGS ('ResolutionTime' = 'Sensitive');

**3. Data Profiling:**

**Profile the average budget of marketing campaigns:**

SELECT AVG(Budget)

FROM enterprise\_data\_catalog.marketing\_data.campaigns;

**Task 3: Data Lineage and Auditing**

1. **Track Data Lineage Across Schemas:**

**Join data between the campaigns and orders tables:**

SELECT m.CampaignID, m.CampaignName, o.OrderID, o.ProductID, o.Quantity

FROM enterprise\_data\_catalog.marketing\_data.campaigns m

JOIN enterprise\_data\_catalog.operations\_data.orders o

ON m.CampaignID = o.ProductID;

**2. Enable and Analyze Audit Logs:**

**Show audit logs for IT incidents:**

SHOW AUDIT LOGS FOR enterprise\_data\_catalog.it\_data;

**Task 4: Implement Fine-Grained Access Control**

**1. Create User Roles and Groups:**

* **Grant schema usage to groups:**
  + **MarketingTeam:**

GRANT USAGE ON SCHEMA enterprise\_data\_catalog.marketing\_data

TO GROUP MarketingTeam;

* OperationsTeam:

GRANT USAGE ON SCHEMA enterprise\_data\_catalog.operations\_data

TO GROUP OperationsTeam;

* ITSupportTeam:

GRANT USAGE ON SCHEMA enterprise\_data\_catalog.it\_data

TO GROUP ITSupportTeam;

**2. Implement Column-Level Security:**

* **Grant limited column access to OperationsTeam:**

GRANT SELECT (CampaignID, CampaignName, StartDate)

ON TABLE enterprise\_data\_catalog.marketing\_data.campaigns

TO GROUP OperationsTeam;

* **Grant full access to MarketingTeam:**

GRANT SELECT

ON TABLE enterprise\_data\_catalog.marketing\_data.campaigns

TO GROUP MarketingTeam;

**3. Row-Level Security:**

* **Implement row-level security for operations team:**

CREATE ROW ACCESS POLICY operations\_team\_policy

ON enterprise\_data\_catalog.operations\_data.orders

USING (User() = 'operations\_rep');

**Task 5: Data Governance and Quality Enforcement**

**1. Set Data Quality Rules:**

* Ensure no campaigns have a zero or negative budget:

SELECT \*

FROM enterprise\_data\_catalog.marketing\_data.campaigns

WHERE Budget <= 0;

**2. Apply Delta Lake Time Travel:**

* **View history of operations orders table:**

DESCRIBE HISTORY enterprise\_data\_catalog.operations\_data.orders;

* **Restore a previous version of the table:**

RESTORE enterprise\_data\_catalog.operations\_data.orders

TO VERSION AS OF <version-number>;

**Task 6: Performance Optimization and Data Cleanup**

**1. Optimize Delta Tables:**

**Optimize the orders table:**

OPTIMIZE enterprise\_data\_catalog.operations\_data.orders;

**Optimize the incidents table:**

OPTIMIZE enterprise\_data\_catalog.it\_data.incidents;

**2. Vacuum Delta Tables:**

**Clean up old files for orders table:**

VACUUM enterprise\_data\_catalog.operations\_data.orders;

**Clean up old files for incidents table:**

VACUUM enterprise\_data\_catalog.it\_data.incidents;