Mini Project: Advanced Data Governance and Security Using Unity Catalog

Task 1: Set Up Multi-Tenant Data Architecture 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

```
- - For Sales data
>>  CREATE TABLE corporate_data_catalog.sales_data.sales_table(
    SalesID STRING,
    CustomerID STRING,
    SalesAmount DECIMAL(10,2),
    SalesDate DATE
    );

- - For HR Data
>>  CREATE TABLE corporate_data_catalog.hr_data.hr_table(
    EmployeeID STRING,
    EmployeeName STRING,
    Department STRING,
    Salary DECIMAL(10,2)
    );
- - For Finance Data
>>  CREATE TABLE corporate_data_catalog.finance_data.finance_table(
    InvoiceID STRING,
    VendorID STRING,
    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.hr_table
   SET TAG 'sensitive' ON COLUMN Salary;

>> ALTER TABLE corporate_data_catalog.finance_data.finance_table
   SET TAG 'sensitive' ON COLUMN InvoiceAmount;
```

3. Data Profiling

```
>> SELECT AVG(SalesAmount), MIN(SalesAmount), MAX(SalesAmount) FROM
   corporate_data_catalog.sales_data.sales_table;
>> SELECT AVG(Salary), MIN(Salary), MAX(Salary) FROM
   corporate_data_catalog.hr_data.hr_table;
>> SELECT AVG(InvoiceAmount), MIN(InvoiceAmount), MAX(InvoiceAmount) FROM
   corporate_data_catalog.finance_data.finance_table;
```

Task 3: Implement Data Lineage and Data Auditing

1. Track Data Lineage

- - creating a reporting table that merges the sales and finance data.

```
>> CREATE TABLE corporate_data_catalog.reporting.sales_finance_report AS
   SELECT s.SalesID, s.CustomerID, s.SalesAmount, s.SalesDate, f.InvoiceID,
   f.InvoiceAmount
   FROM corporate_data_catalog.sales_data.sales_table s
   JOIN corporate_data_catalog.finance_data.finance_table f
   ON s.CustomerID = f.VendorID;
```

2. **Enable Data Audit Logs**

   - - Enabling audit logs for operations performed on the tables

   - Navigate to admin console in databricks
   - Go to audit logs tab and enable audit logs

Task 4: Data Access Control and Security

1. **Set Up Roles and Permissions**

```
>>  CREATE GROUP SalesTeam;

>>  GRANT USAGE ON SCHEMA corporate_data_catalog.sales_data TO SalesTeam;


>>  CREATE GROUP FinanceTeam;

>>  GRANT USAGE ON SCHEMA corporate_data_catalog.sales_data TO FinanceTeam;

>>  GRANT USAGE ON SCHEMA corporate_data_catalog.finance_data TO FinanceTeam;


>>  CREATE GROUP HRTeam;

>>  GRANT USAGE ON SCHEMA corporate_data_catalog.hr_data TO HRTeam;

>>  GRANT UPDATE ON TABLE corporate_data_catalog.hr_data.hr_table TO HRTeam;
```

2. **Implement Column-Level Security**

```
>>  GRANT SELECT ON COLUMN Salary TO HRManager;
```

3. **Row-Level Security**

```
>>  CREATE ROW ACCESS POLICY sales_rep_policy ON
    corporate_data_catalog.sales_data.sales_table
    FOR EACH ROW
    WHEN current_user = sales_rep_id;
```

Task 5: Data Governance Best Practices

1. **Define Data Quality Rules**
   - - Sales amounts are positive in the sales_data table.

```
>>  SELECT * FROM corporate_data_catalog.sales_data.sales_table
    WHERE SalesAmount <= 0;
```

- - Employee salaries are greater than zero in the hr_data table

```
>>  SELECT * FROM corporate_data_catalog.hr_data.hr_table

    WHERE Salary <= 0;
```

- - Invoice amounts in the finance_data table match payment records.

```
>>  SELECT * FROM corporate_data_catalog.finance_data.finance_table

    WHERE InvoiceAmount <> PaymentAmount;
```

2. Apply Time Travel for Data Auditing

```
>>  RESTORE TABLE corporate_data_catalog.finance_data.finance_table

    TO VERSION AS OF 5;
```

Task 6: Optimize and Clean Up Delta Tables

1. **Optimize Delta Tables**

```
>>  OPTIMIZE corporate_data_catalog.sales_data.sales_table;
>>  OPTIMIZE corporate_data_catalog.finance_data.finance_table;
```

2. **Vacuum Delta Tables**

```
>>  VACUUM corporate_data_catalog.sales_data.sales_table RETAIN 168 HOURS;

>>   VACUUM corporate_data_catalog.finance_data.finance_table RETAIN 168 HOURS;
```