**Snowflake Labs: End-to-End Hands-On**

**Lab 1: Create Warehouse – Configure Compute Sizing & Auto-Suspend**

**Objective:** Learn how to provision a compute warehouse with appropriate settings.

**Steps:**

1. Log in to Snowflake (Snowsight or Classic UI).
2. Ensure you have the correct role (e.g., ACCOUNTADMIN or SYSADMIN).
3. SELECT CURRENT\_ROLE();
4. USE ROLE SYSADMIN;
5. Create a **small warehouse** with auto-suspend and auto-resume:
6. CREATE OR REPLACE WAREHOUSE LAB\_WH
7. WITH
8. WAREHOUSE\_SIZE = 'SMALL'
9. AUTO\_SUSPEND = 120
10. AUTO\_RESUME = TRUE
11. INITIALLY\_SUSPENDED = TRUE;
12. Verify creation:
13. SHOW WAREHOUSES;

**Validation:** LAB\_WH should appear in the warehouses list with size SMALL and auto-suspend set.

**Lab 2: Database & Schema – Create Database and Schema for Project**

**Objective:** Organize project data in Snowflake.

**Steps:**

1. Create a database:
2. CREATE OR REPLACE DATABASE PROJECT\_DB;
3. Create a schema within the database:
4. CREATE OR REPLACE SCHEMA PROJECT\_DB.RAW\_DATA;
5. Switch context:
6. USE DATABASE PROJECT\_DB;
7. USE SCHEMA RAW\_DATA;

**Validation:**

SELECT CURRENT\_DATABASE(), CURRENT\_SCHEMA();

You should see PROJECT\_DB and RAW\_DATA.

**Lab 3: Load Data – Use COPY INTO to Load from Azure Blob**

**Objective:** Ingest external data into Snowflake tables.

**Pre-requisites:** You must have an **Azure Blob Storage container** with a sample CSV (e.g., sales.csv).

**Steps:**

1. Create an external stage pointing to Azure Blob:
2. CREATE OR REPLACE STAGE azure\_stage
3. URL='azure://<storage-account-name>.blob.core.windows.net/<container>'
4. CREDENTIALS=(AZURE\_SAS\_TOKEN='<sas-token>');
5. Verify stage:
6. LIST @azure\_stage;
7. Create a target table:
8. CREATE OR REPLACE TABLE SALES (
9. ORDER\_ID INT,
10. CUSTOMER\_NAME STRING,
11. PRODUCT STRING,
12. AMOUNT DECIMAL(10,2),
13. ORDER\_DATE DATE
14. );
15. Load data:
16. COPY INTO SALES
17. FROM @azure\_stage/sales.csv
18. FILE\_FORMAT = (TYPE = 'CSV' FIELD\_OPTIONALLY\_ENCLOSED\_BY='"' SKIP\_HEADER=1);
19. Validate:
20. SELECT COUNT(\*) FROM SALES;

**Expected Outcome:** The data from Azure Blob is loaded into the SALES table.

**Lab 4: Run Queries – Perform SELECT, JOIN, and Aggregation**

**Objective:** Query the ingested data.

**Steps:**

1. Basic SELECT:
2. SELECT \* FROM SALES LIMIT 10;
3. Aggregation – Total sales by customer:
4. SELECT CUSTOMER\_NAME, SUM(AMOUNT) AS TOTAL\_SPENT
5. FROM SALES
6. GROUP BY CUSTOMER\_NAME
7. ORDER BY TOTAL\_SPENT DESC;
8. Create another table (e.g., customer region):
9. CREATE OR REPLACE TABLE CUSTOMERS (
10. CUSTOMER\_NAME STRING,
11. REGION STRING
12. );
13. INSERT INTO CUSTOMERS VALUES
14. ('Alice','West'),
15. ('Bob','East'),
16. ('Charlie','South');
17. Perform JOIN:
18. SELECT S.CUSTOMER\_NAME, C.REGION, SUM(S.AMOUNT) AS TOTAL\_SALES
19. FROM SALES S
20. JOIN CUSTOMERS C ON S.CUSTOMER\_NAME = C.CUSTOMER\_NAME
21. GROUP BY S.CUSTOMER\_NAME, C.REGION;

**Expected Outcome:** You should see sales aggregated by **customer and region**.

**Lab 5: Time Travel – Restore Table to Previous State**

**Objective:** Use Snowflake’s **time travel** feature.

**Steps:**

1. Delete a record:
2. DELETE FROM SALES WHERE CUSTOMER\_NAME = 'Alice';
3. Check current table:
4. SELECT \* FROM SALES WHERE CUSTOMER\_NAME = 'Alice';

(No rows returned.)

1. Query the table **before deletion**:
2. SELECT \* FROM SALES
3. AT (OFFSET => -60\*5); -- 5 minutes ago
4. Restore table:
5. CREATE OR REPLACE TABLE SALES\_RESTORED CLONE SALES AT (OFFSET => -60\*5);
6. Verify restored table:
7. SELECT \* FROM SALES\_RESTORED WHERE CUSTOMER\_NAME = 'Alice';

**Expected Outcome:** The deleted rows are visible again in the restored table.