**Snowflake Hands-On Labs**

**Lab 1: Snowflake Cloud Data Platform Overview**

**Objective:** Get familiar with the Snowflake UI, roles, and navigation.  
**Steps:**

1. Log in to your Snowflake account (web UI or Snowsight).
2. Explore the **Databases**, **Warehouses**, and **Worksheets** tabs.
3. Identify the current **role** and **warehouse** assigned.
4. SELECT CURRENT\_ROLE(), CURRENT\_WAREHOUSE(), CURRENT\_DATABASE(), CURRENT\_SCHEMA();
5. Switch to another role (e.g., SYSADMIN).
6. USE ROLE SYSADMIN;
7. Validate permissions by creating a dummy database:
8. CREATE DATABASE LAB\_DB;
9. SHOW DATABASES;

**Validation:** You should see LAB\_DB listed under databases.

**Lab 2: Virtual Warehouses – Compute Separation from Storage**

**Objective:** Understand Snowflake’s compute/storage architecture.  
**Steps:**

1. Create a new virtual warehouse:
2. CREATE WAREHOUSE LAB\_WH
3. WITH WAREHOUSE\_SIZE = 'SMALL'
4. AUTO\_SUSPEND = 60
5. AUTO\_RESUME = TRUE;
6. Start the warehouse and run a query:
7. USE WAREHOUSE LAB\_WH;
8. SELECT CURRENT\_WAREHOUSE();
9. Suspend the warehouse manually:
10. ALTER WAREHOUSE LAB\_WH SUSPEND;
11. Resume it again:
12. ALTER WAREHOUSE LAB\_WH RESUME;

**Validation:** The warehouse resumes only when queries run.

**Lab 3: Databases, Schemas, and Tables**

**Objective:** Work with standard and transient tables.  
**Steps:**

1. Create a database and schema:
2. CREATE DATABASE SALES\_DB;
3. CREATE SCHEMA SALES\_DB.PUBLIC;
4. Create a **permanent table**:
5. CREATE OR REPLACE TABLE SALES\_DB.PUBLIC.ORDERS (
6. ORDER\_ID INT,
7. CUSTOMER\_NAME STRING,
8. AMOUNT DECIMAL(10,2)
9. );
10. Create a **transient table**:
11. CREATE OR REPLACE TRANSIENT TABLE SALES\_DB.PUBLIC.TEMP\_ORDERS (
12. ORDER\_ID INT,
13. STATUS STRING
14. );

**Validation:** Run SHOW TABLES; and note the difference in type.

**Lab 4: File Formats – CSV, JSON, Parquet, ORC**

**Objective:** Define and use file formats for staged data.  
**Steps:**

1. Create file formats:
2. CREATE FILE FORMAT my\_csv TYPE = 'CSV' FIELD\_OPTIONALLY\_ENCLOSED\_BY='"';
3. CREATE FILE FORMAT my\_json TYPE = 'JSON';
4. CREATE FILE FORMAT my\_parquet TYPE = 'PARQUET';
5. CREATE FILE FORMAT my\_orc TYPE = 'ORC';
6. Validate:
7. SHOW FILE FORMATS;

**Lab 5: Data Loading with COPY INTO**

**Objective:** Load data into Snowflake tables from an internal stage.  
**Steps:**

1. Upload a CSV file (orders.csv) to a stage:
2. CREATE OR REPLACE STAGE mystage;
3. PUT file://orders.csv @mystage;
4. LIST @mystage;
5. Load data:
6. COPY INTO SALES\_DB.PUBLIC.ORDERS
7. FROM @mystage/orders.csv
8. FILE\_FORMAT = (FORMAT\_NAME = my\_csv);
9. Verify:
10. SELECT \* FROM SALES\_DB.PUBLIC.ORDERS;

**Lab 6: Querying Data with SQL**

**Objective:** Run SQL queries against Snowflake tables.  
**Steps:**

1. Simple query:
2. SELECT CUSTOMER\_NAME, SUM(AMOUNT) AS TOTAL\_SPENT
3. FROM SALES\_DB.PUBLIC.ORDERS
4. GROUP BY CUSTOMER\_NAME
5. ORDER BY TOTAL\_SPENT DESC;
6. Filtering:
7. SELECT \* FROM SALES\_DB.PUBLIC.ORDERS WHERE AMOUNT > 500;

**Lab 7: Time Travel**

**Objective:** Query historical versions of data.  
**Steps:**

1. Delete a row:
2. DELETE FROM SALES\_DB.PUBLIC.ORDERS WHERE ORDER\_ID = 1;
3. Check current state:
4. SELECT \* FROM SALES\_DB.PUBLIC.ORDERS;
5. Query using **time travel**:
6. SELECT \* FROM SALES\_DB.PUBLIC.ORDERS AT (OFFSET => -60\*5); -- 5 minutes ago

**Lab 8: Cloning Basics**

**Objective:** Use zero-copy cloning.  
**Steps:**

1. Clone an existing table:
2. CREATE OR REPLACE TABLE SALES\_DB.PUBLIC.ORDERS\_CLONE CLONE SALES\_DB.PUBLIC.ORDERS;
3. Query cloned table:
4. SELECT \* FROM SALES\_DB.PUBLIC.ORDERS\_CLONE;

**Validation:** Data is instantly available without extra storage.