DAY1: SQL & Snowflake Foundations (Part 1)

1. Introduction to Relational Databases & Snowflake Basics

Concept Explanation

Relational databases store data in **tables** made of rows and columns. Data is structured and can be connected using relationships (keys).
Snowflake uses the same relational model but offers **cloud-native features** like instant scaling, separation of storage & compute, and zero-maintenance operations.
Objects in Snowflake follow a hierarchy:
Database → Schema → Table

Query: Show all databases

```
SHOW DATABASES;
```

Explanation: Lists all databases available to your Snowflake role.

Query: Select a database and schema

```
USE DATABASE DEMO_DB;
USE SCHEMA PUBLIC;
```

Explanation: Sets the active database and schema so Snowflake knows where to run your queries.

Query: Inspect tables in a schema

```
SHOW TABLES IN SCHEMA DEMO_DB.PUBLIC;
```

Explanation: Displays all tables you can work with in the current schema.

---

2. Snowflake Architecture Overview & Access Setup

Concept Explanation

Snowflake has a **multi-cluster shared architecture** with 3 key layers:

1. **Storage Layer** – Stores compressed, columnar data.
2. **Compute Layer (Warehouses)** – Executes SQL queries independently.
3. **Cloud Services Layer** – Authentication, query optimization, metadata.

Users need proper **roles**, **warehouses**, and **permissions** to run SQL.

Query: Check your current session settings

```
SELECT CURRENT_ROLE(), CURRENT_WAREHOUSE(), CURRENT_DATABASE(),
CURRENT_SCHEMA();
```

Explanation: Confirms your active role, compute warehouse, and working environment.

Query: Start (resume) a warehouse

```
ALTER WAREHOUSE COMPUTE_WH RESUME;
```

Explanation: Activates the compute resource needed to run queries.

Query: Create a warehouse (if permissions allow)

```
CREATE WAREHOUSE IF NOT EXISTS TRAINING_WH
  WAREHOUSE_SIZE = 'XSMALL'
  AUTO_SUSPEND = 60
  AUTO_RESUME = TRUE;
```

Explanation: Creates a cost-efficient warehouse for query execution with auto-suspend to save credits.

---

3. SQL Essentials: SELECT, WHERE

Concept Explanation

SELECT retrieves data from tables.
WHERE filters rows based on conditions.
These commands form the foundation of all SQL queries.

Query: Basic SELECT

```
SELECT *
FROM EMPLOYEES;
```

Explanation: Returns all rows and columns from the EMPLOYEES table.

Query: Select specific columns

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME
FROM EMPLOYEES;
```

Explanation: Retrieves only selected columns, improving readability and efficiency.

Query: Filtering data with WHERE

```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT = 'SALES';
```

Explanation: Returns employees only from the Sales department.

Query: Multiple conditions with WHERE

```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT = 'SALES'
  AND SALARY > 70000;
```

Explanation: Applies both filters to narrow down the dataset.

---

4. Hands-on: Extracting Simple Datasets

Concept Explanation

Dataset extraction involves selecting, filtering, sorting, and summarizing data.
Snowflake uses SQL to perform these operations quickly and efficiently due to its scalable compute.

Query: Sort results

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, SALARY
```

```
FROM EMPLOYEES
ORDER BY SALARY DESC;
```

Explanation: Sorts employees from highest to lowest salary.

Query: Limit rows

```
SELECT *
FROM EMPLOYEES
ORDER BY HIRE_DATE DESC
LIMIT 5;
```

Explanation: Returns the 5 most recently hired employees.

Query: Filter by date range

```
SELECT *
FROM SALES
WHERE ORDER_DATE BETWEEN '2025-01-01' AND '2025-12-31';
```

Explanation: Extracts sales data for the specified time period.

Query: Simple aggregation

```
SELECT DEPARTMENT, COUNT(*) AS TOTAL_EMPLOYEES
FROM EMPLOYEES
GROUP BY DEPARTMENT;
```

Explanation: Shows how many employees exist in each department.