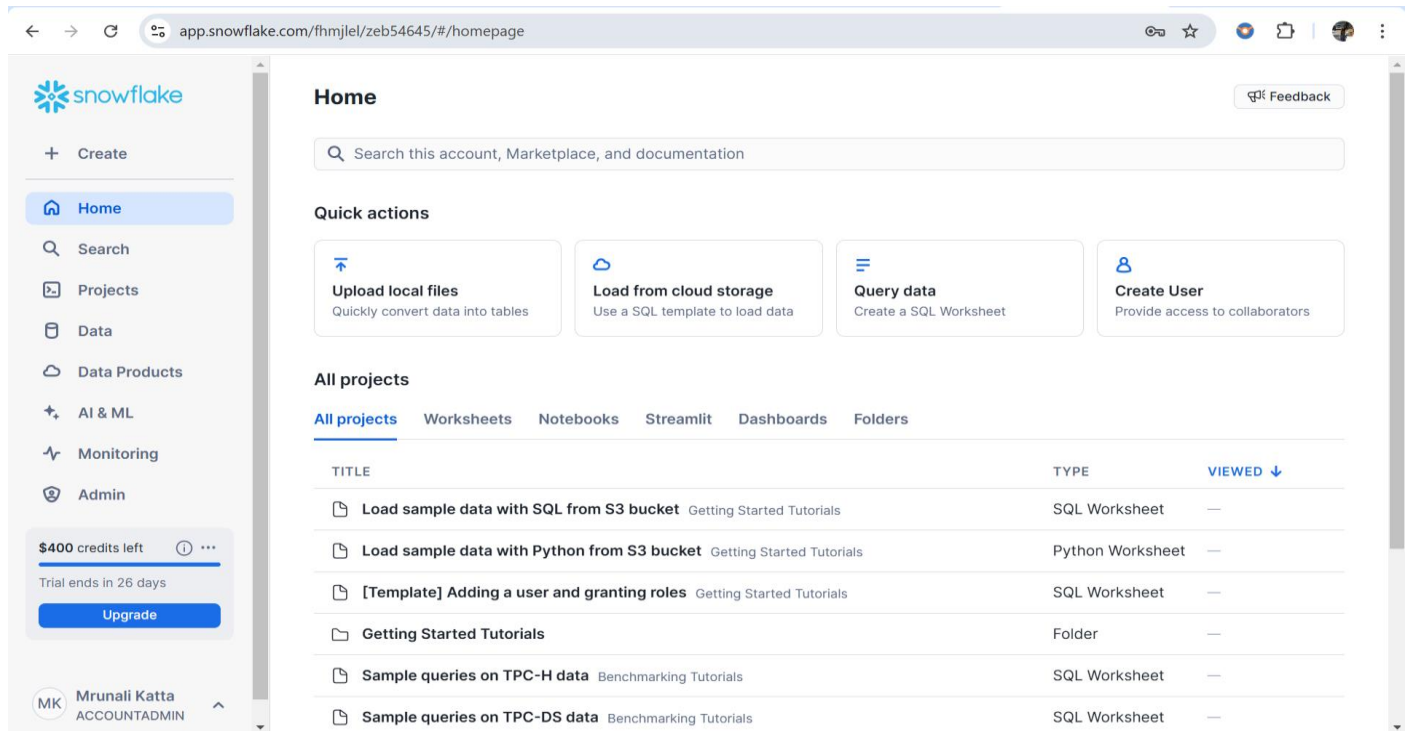


Name: Mrunali Katta

ID : 017516785

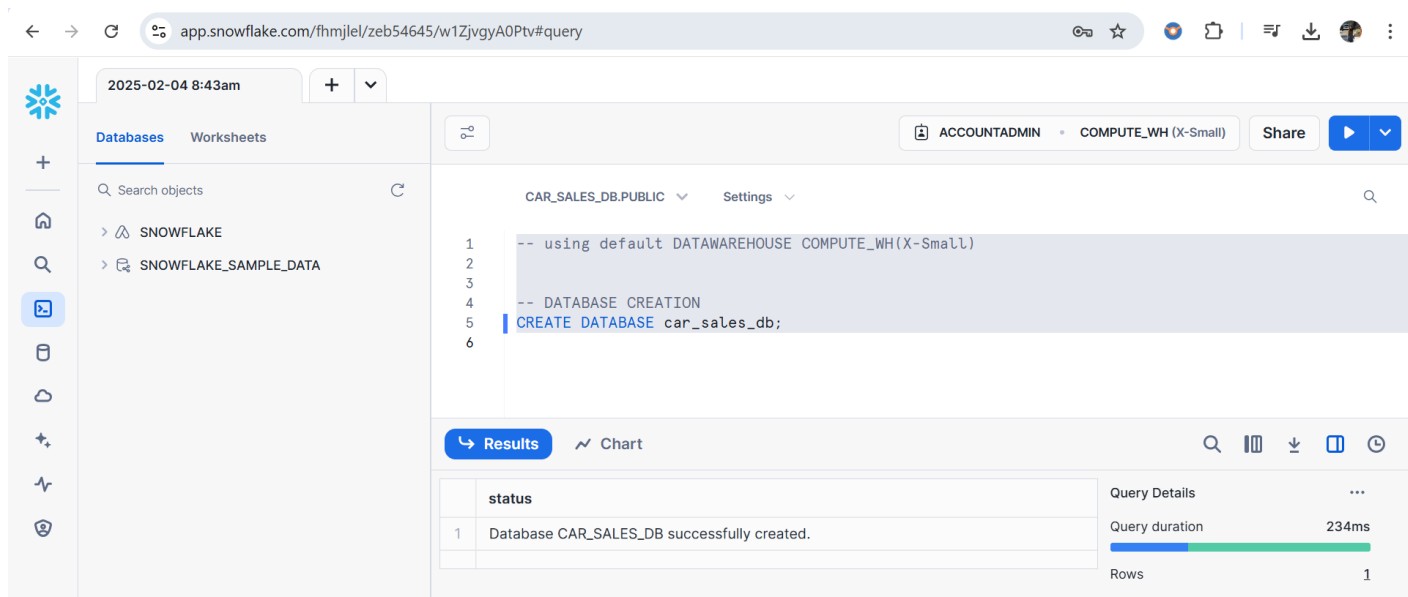
DATA 226 - HOMEWORK 1

1. Created a Snowflake account.



The screenshot shows the Snowflake web interface. The left sidebar contains navigation options: Create, Home (selected), Search, Projects, Data, Data Products, AI & ML, Monitoring, and Admin. A credit balance of \$400 is shown, along with a trial end date of 26 days and an Upgrade button. The main content area is titled 'Home' and includes a search bar, 'Quick actions' (Upload local files, Load from cloud storage, Query data, Create User), and 'All projects'. The 'All projects' section lists several projects, including 'Load sample data with SQL from S3 bucket', 'Load sample data with Python from S3 bucket', '[Template] Adding a user and granting roles', 'Getting Started Tutorials', 'Sample queries on TPC-H data', and 'Sample queries on TPC-DS data'.

2. Created a Database 'car_sales_db', Schema 'car_sales_schema', Table 'car_sales_tbl' using the default warehouse **"COMPUTE_WH"** of size XSMALL.



The screenshot shows the Snowflake Query Editor interface. The top bar displays the date and time (2025-02-04 8:43am) and the user (ACCOUNTADMIN). The left sidebar shows the 'Databases' tab with a search bar and a list of objects: SNOWFLAKE and SNOWFLAKE_SAMPLE_DATA. The main area shows the query editor with the following SQL code:

```
1 -- using default DATAWAREHOUSE COMPUTE_WH(X-Small)
2
3
4 -- DATABASE CREATION
5 CREATE DATABASE car_sales_db;
6
```

The 'Results' tab is selected, showing a table with one row:

status
Database CAR_SALES_DB successfully created.

Query Details are shown on the right, including Query duration (234ms) and Rows (1).

CODE:

```
CREATE DATABASE car_sales_db;
```

2025-02-04 8:43am

Databases Worksheets

Search objects

SNOWFLAKE

SNOWFLAKE_SAMPLE_DATA

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

CAR_SALES_DB.CAR_SALES_SCHEMA Settings

```
1 -- using default DATAWAREHOUSE COMPUTE_WH(X-Small)
2
3
4 -- DATABASE CREATION
5 CREATE DATABASE car_sales_db;
6
7 --schema creation
8 CREATE SCHEMA car_sales_schema;
9
```

Results Chart

status
1 Schema CAR_SALES_SCHEMA successfully created.

Query Details

Query duration 95ms

Rows 1

Query ID 01ba2bcf-0004-8b2b-0...

CODE:

```
CREATE SCHEMA car_sales_schema;
```

2025-02-04 8:43am

Databases Worksheets

Search objects

CAR_SALES_DB

CAR_SALES_SCHEMA

Tables

CAR_SALES_TBL

INFORMATION_SCHEMA

CAR_SALES_TBL

CARS_NAME VARCHAR(100)

CAR_PRICE NUMBER(38,0)

PURCHASE_DATE DATE

BRAND_NAME VARCHAR(100)

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

CAR_SALES_DB.CAR_SALES_SCHEMA Settings

```
7 --schema creation
8 CREATE SCHEMA car_sales_schema;
9
10
11 --table creation
12 CREATE TABLE car_sales_tbl (
13   cars_name VARCHAR(100),
14   car_price INTEGER,
15   purchase_date DATE,
16   brand_name VARCHAR(100)
17 );
18
```

Results Chart

status
1 Table CAR_SALES_TBL successfully created.

Query Details

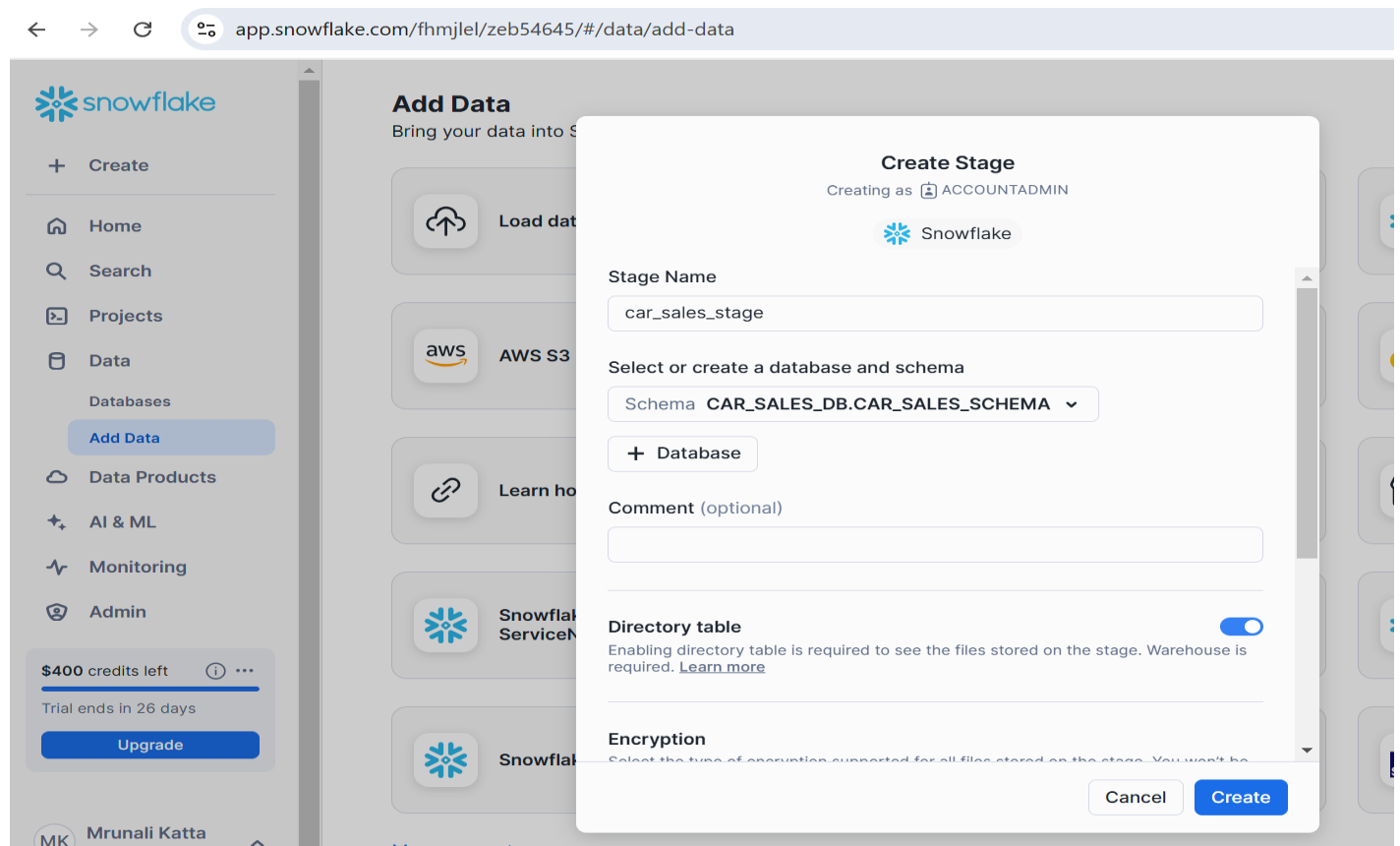
Query duration 271ms

Rows 1

CODE:

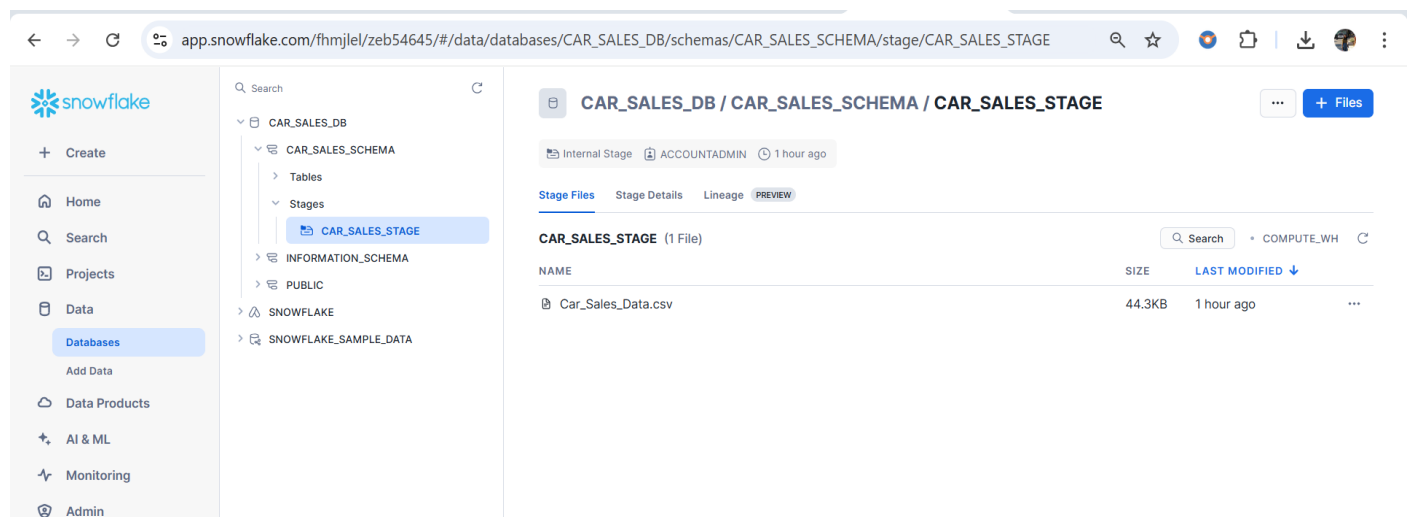
```
CREATE TABLE car_sales_tbl (
  cars_name VARCHAR(100),
  car_price INTEGER,
  purchase_date DATE,
  brand_name VARCHAR(100)
);
```

3. Stage creation Created a stage 'car_sales_stage' in Snowflake and load the csv.



The screenshot shows the Snowflake 'Create Stage' dialog box. The 'Stage Name' is 'car_sales_stage'. The 'Select or create a database and schema' dropdown is set to 'Schema CAR_SALES_DB.CAR_SALES_SCHEMA'. The 'Directory table' toggle is turned on. The 'Encryption' section is partially visible. The 'Create' button is highlighted.

Loaded the **Car_Sales_Data.csv** file into the earlier created stage 'car_sales_stage'



The screenshot shows the Snowflake interface with the 'CAR_SALES_STAGE' stage selected. The breadcrumb path is 'CAR_SALES_DB / CAR_SALES_SCHEMA / CAR_SALES_STAGE'. The 'Stage Files' tab is active, showing a table with one file: 'Car_Sales_Data.csv', which is 44.3KB and was modified 1 hour ago. The left sidebar shows the database structure with 'CAR_SALES_STAGE' highlighted under 'Stages'.

4. Created a file format called 'car_sales_format' and Loaded Data into the 'car_sales_tbl' table using COPY INTO query.

The screenshot shows the Snowflake web interface. On the left, the 'Databases' tab is active, showing a tree view of the database structure. The 'CAR_SALES_DB' database is selected, and the 'CAR_SALES_SCHEMA' is expanded. Under 'Tables', 'CAR_SALES_TBL' is highlighted. The main panel displays the SQL code for creating the schema, table, and file format. The 'Results' tab at the bottom shows a single message: 'File format CAR_SALES_FILEFORMAT successfully created.' The query details on the right indicate a duration of 108ms.

```
--schema creation
CREATE SCHEMA car_sales_schema;

--table creation
CREATE TABLE car_sales_tbl (
  cars_name VARCHAR(100),
  car_price INTEGER,
  purchase_date DATE,
  brand_name VARCHAR(100)
);

-- file format
CREATE OR REPLACE FILE FORMAT car_sales_fileformat
TYPE = 'CSV'
FIELD_OPTIONALLY_ENCLOSED_BY = ''
SKIP_HEADER = 1
NULL_IF = ('NULL', '');
```

status
1 File format CAR_SALES_FILEFORMAT successfully created.

Query Details
Query duration 108ms

CODE:

```
CREATE OR REPLACE FILE FORMAT car_sales_fileformat
TYPE = 'CSV'
FIELD_OPTIONALLY_ENCLOSED_BY = ''
SKIP_HEADER = 1
NULL_IF = ('NULL', '');
```

Now loaded data into the 'car_sales_tbl' table using COPY INTO query

The screenshot shows the Snowflake web interface with the 'COPY INTO' query executed. The main panel displays the SQL code for loading data from the '@car_sales_stage' table into the 'car_sales_tbl' table using the 'car_sales_fileformat' file format. The 'Results' tab at the bottom shows a table with columns: file, status, rows_parsed, rows_loaded, error_limit, errors_seen, first_error, and first_error_line. The first row shows that 1200 rows were successfully loaded from 'car_sales_stage/Car_Sales_Data.csv' with no errors. The query details on the right indicate a duration of 1.7s and 1 row loaded.

```
--Load data into new table 'car_sales_tbl' of the newly created
COPY INTO car_sales_tbl
FROM @car_sales_stage
FILE_FORMAT = (FORMAT_NAME = car_sales_fileformat);
```

file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line
car_sales_stage/Car_Sales_Data.csv	LOADED	1200	1200	1	0	null	

Query Details
Query duration 1.7s
Rows 1

CODE:

```
COPY INTO car_sales_tbl
FROM @car_sales_stage
FILE_FORMAT = (FORMAT_NAME = car_sales_fileformat);
```

5. Write an SQL queries for each that shows the following insights

1. The average number of cars sold per month in 2024.

The screenshot shows the Snowflake web interface. The left sidebar displays the database structure: CAR_SALES_DB, CAR_SALES_SCHEMA, Tables (CAR_SALES_TBL), Stages, File Formats, INFORMATION_SCHEMA, Views, and APPLICABLE_ROLES. The main editor shows the following SQL query:

```
-- SQL QUERIES
-- 1) The average number of cars sold per month in 2024.
SELECT EXTRACT(MONTH FROM purchase_date) AS month, COUNT(*) AS avg_cars_sold
FROM car_sales_tbl
WHERE EXTRACT(YEAR FROM purchase_date) = 2024
GROUP BY EXTRACT(MONTH FROM purchase_date)
ORDER BY month;
```

The 'Results' tab shows a table with 12 rows, representing the months of 2024. The columns are MONTH and AVG_CARS_SOLD. The values for AVG_CARS_SOLD range from 4 to 116.

MONTH	AVG_CARS_SOLD
1	4
2	116
3	94
4	106
5	89
6	99
7	103
8	91
9	103
10	93
11	96
12	104

Query Details: Query duration 82ms, Rows 12, Query ID 01ba2cce-0004-8b2d-0...

CODE:

```
SELECT EXTRACT(MONTH FROM purchase_date) AS month, COUNT(*) AS avg_cars_sold
FROM car_sales_tbl
WHERE EXTRACT(YEAR FROM purchase_date) = 2024
GROUP BY EXTRACT(MONTH FROM purchase_date)
ORDER BY month;
```

2) The top 5 best-selling cars along with the count in 2024.

The screenshot shows the Snowflake web interface. The left sidebar displays the database structure: CAR_SALES_DB, CAR_SALES_SCHEMA, Tables (CAR_SALES_TBL), Stages, File Formats, INFORMATION_SCHEMA, Views, and APPLICABLE_ROLES. The main editor shows the following SQL query:

```
-- The top 5 best-selling cars along with the count in 2024.
SELECT cars_name, COUNT(*) AS total_sold
FROM car_sales_tbl
WHERE EXTRACT(YEAR FROM purchase_date) = 2024
GROUP BY cars_name
ORDER BY total_sold DESC
LIMIT 5;
```

The 'Results' tab shows a table with 5 rows, representing the top 5 best-selling cars. The columns are CARS_NAME and TOTAL_SOLD. The values for TOTAL_SOLD are 140, 126, 115, 114, and 112.

CARS_NAME	TOTAL_SOLD
Mercedes C-Class	140
Hyundai Elantra	126
Chevrolet Silverado	115
BMW 3 Series	114
Toyota Camry	112

Query Details: Query duration 346ms, Rows 5, Query ID 01ba2cc2-0004-8b2c-0...

CODE:

```
SELECT cars_name, COUNT(*) AS total_sold
FROM car_sales_tbl
WHERE EXTRACT(YEAR FROM purchase_date) = 2024
GROUP BY cars_name
ORDER BY total_sold DESC
LIMIT 5;
```

3) The top 3 least-selling cars along with the count in 2024.

The screenshot shows a database query editor interface. On the left, a sidebar displays the database structure: 'CAR_SALES_DB' containing 'CAR_SALES_SCHEMA', which has a table 'CAR_SALES_TBL' with 1.2K rows. The main editor area shows a SQL query with line numbers 48 to 61. The query is a continuation of the previous one, filtering for the year 2024 and limiting the results to 3. Below the query editor, the 'Results' tab is active, showing a table with 3 rows and 2 columns: 'CARS_NAME' and 'TOTAL_SOLD'. The results are: Audi A4 (79), Nissan Altima (98), and Honda Civic (99). To the right of the results table, a 'Query Details' panel shows a query duration of 372ms and 3 rows returned.

```
48 WHERE EXTRACT(YEAR FROM purchase_date) = 2024
49 GROUP BY cars_name
50 ORDER BY total_sold DESC
51 LIMIT 5;
52
53
54 -- 3) The top 3 least-selling cars along with the count in 2024.
55 SELECT cars_name, COUNT(*) AS total_sold
56 FROM car_sales_tbl
57 WHERE EXTRACT(YEAR FROM purchase_date) = 2024
58 GROUP BY cars_name
59 ORDER BY total_sold
60 LIMIT 3;
61
```

	CARS_NAME	TOTAL_SOLD
1	Audi A4	79
2	Nissan Altima	98
3	Honda Civic	99

Query Details
Query duration: 372ms
Rows: 3
Query ID: 01ba2cc6-0004-8b26-0...

CODE:

```
SELECT cars_name, COUNT(*) AS total_sold
FROM car_sales_tbl
WHERE EXTRACT(YEAR FROM purchase_date) = 2024
GROUP BY cars_name
ORDER BY total_sold
LIMIT 3;
```