# SQL SELECT Queries Practical Report

**Name:** Samridh Srivastava

**UID:** 24BDA702028

**Section:** 24_AIT_KRG_G2

## AIM

To understand and implement SQL SELECT queries using various clauses such as WHERE, ORDER BY, GROUP BY, and HAVING to retrieve and manipulate data efficiently from relational database tables.

## Software Requirements

### Database Management System

- PostgreSQL

### Database Administration Tool

- pgAdmin

## Objective

- To practice writing SQL SELECT statements
- To apply filtering conditions using the WHERE clause
- To sort query results using the ORDER BY clause
- To group records using the GROUP BY clause
- To filter grouped data using the HAVING clause
- To analyze data using aggregate functions like COUNT(), SUM(), AVG(), MIN(), and MAX()

## Actual Implementation Steps / Logic / Tasks Performed

## Table Setup

Created a table named Students to store student information. The table contains the following fields:

- **id**: Unique student ID (Primary Key)
- **name**: Student name
- **city**: City of the student
- **marks**: Marks obtained by the student

## Data Insertion

Inserted multiple student records into the Students table. Each record includes values for id, name, city, and marks.

## City-wise Student Count

Counted the number of students in each city using the COUNT() function. Grouping was performed using GROUP BY city so each city gets its own count.

Both methods were used:

- COUNT(*) to count total rows
- COUNT(id) to count non-null IDs (same result here since ID is always present)

## Sorting Based on Count

Sorted the cities based on the number of students in each city. Used ORDER BY to arrange the results in ascending order.

## Filtering Cities with Minimum Students

Identified cities having at least 3 students. Used the HAVING clause because it filters results after grouping.

## Average Marks per City

Calculated the average marks for each city using AVG(marks). Used GROUP BY city to calculate separate averages for each city. Displayed average marks with precision up to 2 decimal places.

# Procedure of the Practical (Step-by-Step)

1. Open the SQL environment (such as PostgreSQL / MySQL / SQL Server) and connect to the required database.
2. Create a table named Students with columns for:
    - Student ID

- Student Name
- City
- Marks

3. Also set the id column as the Primary Key.
4. Insert the given student records into the Students table by providing values for each column.
5. Execute a query to count the number of students in each city using COUNT() along with GROUP BY city.
6. Execute a query to sort the city-wise student count results using ORDER BY (ascending order).
7. Execute a query to find cities having at least 3 students using HAVING COUNT() >= 3.
8. Execute a query to calculate the average marks of students in each city using AVG(marks) and group the results using GROUP BY city.

## Input/Output Analysis

### Input

Student data with ID, Name, City, and Marks fields to be inserted into the database.

### Output

Query results showing:

- City-wise student counts
- Sorted city counts in ascending order
- Cities with at least 3 students
- Average marks per city with 2 decimal precision

## Learning Outcomes

- Filter records using the WHERE clause
- Group records using GROUP BY
- Apply conditions on grouped data using HAVING
- Sort query results using ORDER BY

## Code Implementation

### Table Creation

```
CREATE TABLE Students (
id NUMERIC PRIMARY KEY,
name VARCHAR(50),
city VARCHAR(30),
marks NUMERIC(10,0)
);
```

## Data Insertion

```
INSERT INTO Students VALUES (1, 'Aman', 'Mohali', 85);
INSERT INTO Students VALUES (2, 'Rohit', 'Mohali', 78);
INSERT INTO Students VALUES (3, 'Neha', 'Mohali', 92);
INSERT INTO Students VALUES (4, 'Simran', 'Amritsar', 88);
INSERT INTO Students VALUES (5, 'Karan', 'Amritsar', 75);
```

## COUNT Number of Students in Each City

**Method I:**

```
SELECT CITY, COUNT(*) AS COUNT_STUDENTS
FROM STUDENTS
GROUP BY CITY;
```

**Method II:**

```
SELECT CITY, COUNT(ID) AS COUNT_STUDENTS
FROM STUDENTS
GROUP BY CITY;
```

## Sort Based on Count of Students in Each City

**Method I:**

```
SELECT CITY, COUNT(ID) AS COUNT_STUDENTS
FROM STUDENTS
GROUP BY CITY
ORDER BY COUNT_STUDENTS ASC;
```

**Method II:**

```
SELECT CITY, COUNT(
```

*) AS COUNT_STUDENTSFROM STUDENTSGROUP BY CITYORDER BY COUNT(*) ASC;

## Find Cities Having Count at Least 3

```
SELECT CITY, COUNT(ID) AS COUNT_STUDENTS
FROM STUDENTS
GROUP BY CITY
HAVING COUNT(ID) >= 3;
```

## Find Average Marks of Each City

```
SELECT CITY, AVG(MARKS)::NUMERIC(10,2) AS AVERAGE_MARKS
FROM STUDENTS
GROUP BY CITY;
```

## Conclusion

This practical exercise demonstrates the effective use of SQL SELECT queries with various clauses to manipulate and analyze data. The implementation covered fundamental database operations including table creation, data insertion, data aggregation, and filtering using GROUP BY and HAVING clauses. These queries form the foundation for efficient data retrieval and analysis in relational database management systems.