UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+ ACCREDITED UNIVERSITY

Experiment No. 2

Student Name: Roshan Kumar Singh
Branch: MCA(GEN)
Semester: 2nd
Subject Name: Technical Training-1

UID: 25MCA20067
Section/Group: 25MCA-1_A
Date of Performance: 13/01/26
Subject Code: 25CAP-652

Aim:

To implement and analyze SQL SELECT queries using filtering, sorting, grouping, and aggregation concepts in PostgreSQL for efficient data retrieval and analytical reporting.

Objective:

- To retrieve specific data using filtering conditions
- To sort query results using single and multiple attributes
- To perform aggregation using grouping techniques
- To apply conditions on aggregated data
- To understand real-world analytical queries commonly asked in placement interviews

S/W Requirement:

PostgreSQL

Procedure:
Step 1: Database and Table Preparation

- Start the PostgreSQL server.
- Open the PostgreSQL client tool.
- Create a database for the experiment.

- Prepare a sample table representing customer orders containing details such as customer name, product, quantity, price, and order date.
- Insert sufficient sample records to allow meaningful analysis.

Step 2: Filtering Data Using Conditions

- Execute data retrieval operations to display only those records that satisfy specific conditions, such as higher-priced orders.
- Observe how filtering limits the number of rows returned.

Step 3: Sorting Query Results

- Retrieve selected columns from the table and arrange the output based on numerical values such as price.
- Perform sorting using both ascending and descending order.
- Apply sorting on more than one attribute to understand priority-based ordering.

Step 4: Grouping Data for Aggregation

- Group records based on a common attribute such as product.
- Calculate aggregate values like total sales for each group.
- Analyze how multiple rows are combined into summarized results.

Step 5: Applying Conditions on Aggregated Data

- Apply conditions on grouped results to retrieve only those groups that satisfy specific aggregate criteria.
- Compare the difference between row-level filtering and group-level filtering.

Step 6: Conceptual Understanding of Filtering vs Aggregation Conditions

- Analyze scenarios where conditions are incorrectly applied before grouping.
- Correctly apply conditions after grouping to avoid logical errors.

Code:

```
CREATE TABLE customer_orders(
   order_id INT PRIMARY KEY ,
   customer_name VARCHAR(50),
   product VARCHAR(50),
   quantity INT,
   price INT,
   order_date DATE
);

INSERT INTO customer_orders
(order_id,customer_name,product,quantity,price,order_date)VALUES
(1,'Roshan','Laptop',1,55000,'2025-01-10'),
(2,'Sanchit','Mouse',2,1200,'2025-01-11'),
(3,'Riya','Laptop',1,58000,'2025-01-12'),
(4,'Rohan','Keyboard',1,2500,'2025-01-12'),
(5,'Swayam','Mouse',3,1800,'2025-01-13'),
(6,'Rittika','Laptop',2,110000,'2025-01-14');

--step 2
SELECT * FROM customer_orders WHERE price>50000;

--step3
--asc
SELECT customer_name,product,price FROM customer_orders ORDER BY price ASC;
--desc
SELECT customer_name,product,price FROM customer_orders ORDER BY price DESC;
--mul col
SELECT customer_name,product,price FROM customer_orders ORDER BY product
ASC, price DESC;

--step4
SELECT product, SUM(price) AS total_sales FROM customer_orders GROUP BY
product;

--step5
```

SELECT product, SUM(price) AS total_sales FROM customer_orders GROUP BY product HAVING SUM(price)>60000;

--step6
SELECT product, SUM(price * quantity) AS total_sales FROM customer_orders WHERE SUM(price * quantity) > 60000 GROUP BY product;

SELECT product, SUM(price) FROM customer_orders GROUP BY product HAVING SUM(price)>60000;

Output:

Step2 output:

	order_id [PK] integer	customer_name character varying (50)	product character varying (50)	quantity integer	price integer	order_date date
1	1	Roshan	Laptop	1	55000	2025-01-10
2	3	Riya	Laptop	1	58000	2025-01-12
3	6	Rittika	Laptop	2	110000	2025-01-14

Step3 outputs
Asc order:

	customer_name character varying (50)	product character varying (50)	price integer
1	Sanchit	Mouse	1200
2	Swayam	Mouse	1800
3	Rohan	Keyboard	2500
4	Roshan	Laptop	55000
5	Riya	Laptop	58000
6	Rittika	Laptop	110000

Desc order:

	customer_name character varying (50)	product character varying (50)	price integer
1	Rittika	Laptop	110000
2	Riya	Laptop	58000
3	Roshan	Laptop	55000
4	Rohan	Keyboard	2500
5	Swayam	Mouse	1800
6	Sanchit	Mouse	1200

Multiple column:

	customer_name character varying (50)	product character varying (50)	price integer
1	Rohan	Keyboard	2500
2	Rittika	Laptop	110000
3	Riya	Laptop	58000
4	Roshan	Laptop	55000
5	Swayam	Mouse	1800
6	Sanchit	Mouse	1200

Step4 output:

	product character varying (50)	total_sales bigint
1	Mouse	3000
2	Keyboard	2500
3	Laptop	223000

Step5 output:

	product character varying (50)	total_sales bigint
1	Laptop	223000

Step6 error while where conditions are incorrectly applied before grouping.

Data Output | Messages | Notifications

```
ERROR:  aggregate functions are not allowed in WHERE
LINE 1: ...antity) AS total_sales FROM customer_orders WHERE SUM(price ...
                                                             ^

SQL state: 42803
Character: 81
```

Step6 output:

	product character varying (50)	sum bigint
1	Laptop	223000

Learning Outcomes:

- Understood how data can be filtered to retrieve only relevant records from a database.
- Learnt how sorting improves readability and usefulness of query results in reports.
- Gained the ability to group data for analytical purposes.
- Clearly able to differentiate between row-level conditions and group-level conditions.
- Developed confidence in writing analytical SQL queries used in real-world scenarios.