Branch: MCA (Data Science) Kargil	**Semester:** 2
Student Name: Sameer Shahi	**UID:** 25MCD10025
Subject Name: Technical Training - I Lab	**Subject Code**: 25CAP-652
Section/Group: A	**Date of Performance:** 20-01-2026

Experiment No. 2

1. **Aim/Overview of the practical:**

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

2. **Tools Used:**

   PostgreSQL

3. **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

4. **Practical / Experiment Steps:**

   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.

Purpose: To create a realistic dataset for performing analytical queries.

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.

Observation: Filtering reduces unnecessary data processing and improves query efficiency.

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.

Observation: Sorting is essential for reports, rankings, and ordered displays.

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.

Observation: Grouping transforms transactional data into analytical insights.

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.

Observation: Conditions applied after grouping allow refined analytical reporting.

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.

Observation: Understanding execution order prevents common SQL mistakes frequently tested in interviews.

5. Queries for experiment/Practical:

```
-- Query1. create table
create table orders (
    order_id serial primary key,
    customer_name varchar(50),
    product varchar(50),
    quantity int,
    price numeric(10,2),
    order_date date
);

-- Query2. insert sample data
insert into orders (customer_name, product, quantity, price, order_date) values
('Amit','Laptop',1,55000,'2024-01-10'),
('Riya','Mobile',2,30000,'2024-01-12'),
('Sameer','Laptop',1,60000,'2024-01-15'),
('Neha','Tablet',3,45000,'2024-01-18'),
('Rahul','Mobile',1,15000,'2024-01-20');

-- Query3. filtering
select * from orders
```

where price > 40000;

-- Query4. sorting ascending

select * from orders

order by price asc;

-- Query5. sorting descending

select * from orders

order by price desc;

-- Query6. multiple column sorting

select * from orders

order by product asc, price desc;

-- Query7. grouping with aggregation

select product, sum(price) as total_sales

from orders

group by product;

-- Query8. average price per product

select product, avg(price) as avg_price

from orders

group by product;

-- Query9. count orders per product

select product, count(*) as total_orders

from orders

group by product;

-- Query10. condition on aggregated data (having)

select product, sum(price) as total_sales

from orders

```
group by product
having sum(price) > 50000;
```

-- Filtering vs Aggregation Conditions

-- Query11. Incorrect: condition applied before grouping (using where)

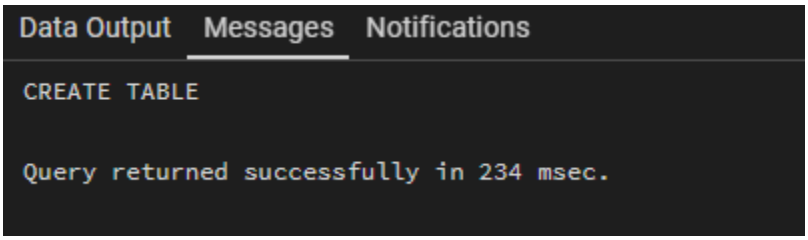
```
select product, sum(price) as total_sales
from orders
where price > 20000
group by product;
```

-- Query12. Correct: condition applied after grouping (using having)

```
select product, sum(price) as total_sales
from orders
where price > 20000
group by product
having sum(price) > 50000;
```

6. Output:

Query1.



Query2.

Data Output | Messages | Notifications

```
INSERT 0 5

Query returned successfully in 81 msec.
```

Query3.

Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1 of 1

	order_id [PK] integer	customer_name character varying (50)	product character varying (50)	quantity integer	price numeric (10,2)	order_date date
1	1	Amit	Laptop	1	55000.00	2024-01-10
2	3	Sameer	Laptop	1	60000.00	2024-01-15
3	4	Neha	Tablet	3	45000.00	2024-01-18

Query4.

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Page No: 1 of 1

	order_id [PK] integer	customer_name character varying (50)	product character varying (50)	quantity integer	price numeric (10,2)	order_date date
1	5	Rahul	Mobile	1	15000.00	2024-01-20
2	2	Riya	Mobile	2	30000.00	2024-01-12
3	4	Neha	Tablet	3	45000.00	2024-01-18
4	1	Amit	Laptop	1	55000.00	2024-01-10
5	3	Sameer	Laptop	1	60000.00	2024-01-15

Query5.

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Page No: 1 of 1

	order_id [PK] integer	customer_name character varying (50)	product character varying (50)	quantity integer	price numeric (10,2)	order_date date
1	3	Sameer	Laptop	1	60000.00	2024-01-15
2	1	Amit	Laptop	1	55000.00	2024-01-10
3	4	Neha	Tablet	3	45000.00	2024-01-18
4	2	Riya	Mobile	2	30000.00	2024-01-12
5	5	Rahul	Mobile	1	15000.00	2024-01-20

Query6.

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Page No: 1 of 1

	order_id [PK] integer	customer_name character varying (50)	product character varying (50)	quantity integer	price numeric (10,2)	order_date date
1	3	Sameer	Laptop	1	60000.00	2024-01-15
2	1	Amit	Laptop	1	55000.00	2024-01-10
3	2	Riya	Mobile	2	30000.00	2024-01-12
4	5	Rahul	Mobile	1	15000.00	2024-01-20
5	4	Neha	Tablet	3	45000.00	2024-01-18

Query7.

Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1

	product character varying (50)	total_sales numeric
1	Mobile	45000.00
2	Tablet	45000.00
3	Laptop	115000.00

Query8.

Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1 of 1

	product character varying (50)	avg_price numeric
1	Mobile	22500.000000000000
2	Tablet	45000.000000000000
3	Laptop	57500.000000000000

Query9.

Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1 of 1

	product character varying (50)	total_orders bigint
1	Mobile	2
2	Tablet	1
3	Laptop	2

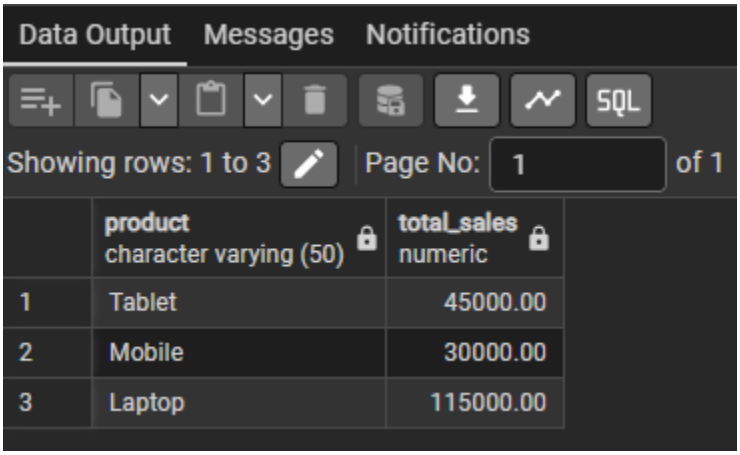
Query10.

Data Output | Messages | Notifications

Showing rows: 1 to 1 | Page No: 1 of 1

	product character varying (50)	total_sales numeric
1	Laptop	115000.00

Query11.

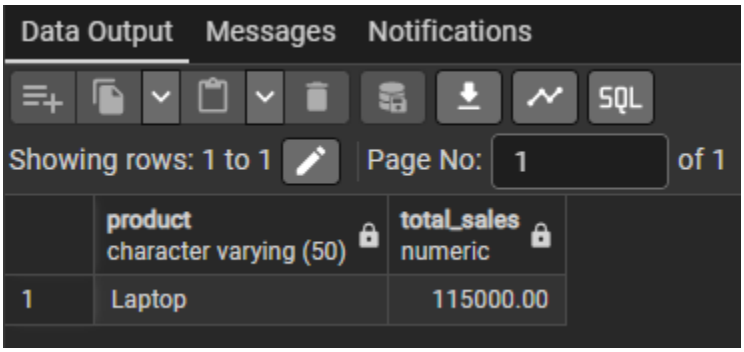


Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1 of 1

	product character varying (50)	total_sales numeric
1	Tablet	45000.00
2	Mobile	30000.00
3	Laptop	115000.00

Query12.



Data Output | Messages | Notifications

Showing rows: 1 to 1 | Page No: 1 of 1

	product character varying (50)	total_sales numeric
1	Laptop	115000.00

7. Learning outcomes (What I have learnt):

- We understand how data can be filtered to retrieve only relevant records from a database.
- We learn how sorting improves readability and usefulness of query results in reports.
- We gain the ability to group data for analytical purposes.
- We clearly differentiate between row-level conditions and group-level conditions.
- We develop confidence in writing analytical SQL queries used in real-world scenarios.
- We are better prepared to answer SQL-based placement and interview questions related to filtering, grouping, and aggregation.