

## Worksheet -2

**Student Name:** SANCHIT KATOCH

**Branch:** MCA

**Semester:** 2<sup>nd</sup>

**Subject Name:** TECHNICAL TRAINING I

**UID:** 25MCA20059

**Section/Group:** 25MCA-1-A

**Date of Performance:** 13/01/2026

**Subject Code:** 25CAP-652

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

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

### 2. 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.

### 3. S/W Requirement:

- Oracle Database Express Edition
- 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

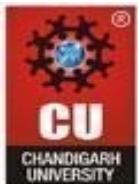
- Group records based on a common attribute such as product.
- Calculate aggregate values like total sales for each group.
- Analyze how multiple rows are combined into summarized results.

#### 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.



#### 4. 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, 'Sanchit Katoch', 'Laptop', 1, 85000, '2025-01-10'),  
(2, 'Roshan Kumar Singh', 'Mouse', 2, 1780, '2025-01-11'),  
(3, 'Anindita Dhar', 'Laptop', 1, 52250, '2025-01-12'),  
(4, 'Ashi Gupta', 'Keyboard', 1, 2575, '2025-01-12'),  
(5, 'Pratham', 'Mouse', 3, 1800, '2025-01-13'),  
(6, 'Swayam', 'Laptop', 2, 120000, '2025-01-14');
```

--step 2

```
SELECT * FROM customer_orders WHERE price > 40000;
```

--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) > 50000;
```

--step6

```
SELECT product, SUM(price) FROM customer_orders GROUP BY product HAVING  
SUM(price) > 50000;
```

## 5. Output:

step 2 output :-

	<b>order_id</b> [PK] integer	<b>customer_name</b> character varying (50)	<b>product</b> character varying (50)	<b>quantity</b> integer	<b>price</b> integer	<b>order_date</b> date
1	1	Sanchit Katoch	Laptop	1	85000	2025-01-10
2	3	Anindita Dhar	Laptop	1	52250	2025-01-12
3	6	Swayam	Laptop	2	120000	2025-01-14

step 3 outputs (ascending, descending, mul\_cols respectively) :-

	<b>customer_name</b> character varying (50)	<b>product</b> character varying (50)	<b>price</b> integer
1	Roshan Kumar Singh	Mouse	1780
2	Pratham	Mouse	1800
3	Ashi Gupta	Keyboard	2575
4	Anindita Dhar	Laptop	52250
5	Sanchit Katoch	Laptop	85000
6	Swayam	Laptop	120000

	<b>customer_name</b> character varying (50)	<b>product</b> character varying (50)	<b>price</b> integer
1	Swayam	Laptop	120000
2	Sanchit Katoch	Laptop	85000
3	Anindita Dhar	Laptop	52250
4	Ashi Gupta	Keyboard	2575
5	Pratham	Mouse	1800
6	Roshan Kumar Singh	Mouse	1780

	<b>customer_name</b> character varying (50) 	<b>product</b> character varying (50) 	<b>price</b> integer 
1	Ashi Gupta	Keyboard	2575
2	Swayam	Laptop	120000
3	Sanchit Katoch	Laptop	85000
4	Anindita Dhar	Laptop	52250
5	Pratham	Mouse	1800
6	Roshan Kumar Singh	Mouse	1780

step 4 output :-

	<b>product</b> character varying (50) 	<b>total_sales</b> 
1	Mouse	3580
2	Keyboard	2575
3	Laptop	257250

step 5 output :-

	<b>product</b> character varying (50) 	<b>total_sales</b> 
1	Laptop	257250

step 6 output :-

	<b>product</b> character varying (50) 	<b>sum</b> 
1	Laptop	257250

## 6. Learning Outcome:

- 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.