



## Worksheet 2

Student Name: Arnab Lala  
Branch: MCA (AI&ML)  
Semester: 2nd  
Subject Name:- DBMS LAB

UID:25MCI10232  
Section/Group: 1/A  
Date of Performance: 22/01/2026  
Subject Code: 25CAP-652

### 1. Aim of the Session

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

### 2. Software Requirements

- PostgreSQL (Database Server)
- pgAdmin
- Windows Operating System

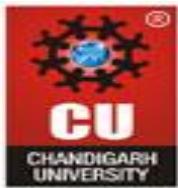
### 3. Objective of the Session

After completing this practical, the student will be able to:

- Retrieve specific data using filtering conditions
- Sort query results using single and multiple attributes
- Perform aggregation using grouping techniques
- Apply conditions on aggregated data using HAVING clause
- Understand real-world analytical queries commonly asked in placement interviews

### 4. Practical / Experiment Steps

- Create a sample table representing customer orders
- Insert realistic records into the table
- Retrieve filtered data using WHERE clause



- Sort query results using ORDER BY
- Group records and apply aggregate functions
- Apply conditions on grouped data using HAVING
- Analyze execution order of WHERE and HAVING clauses

## 5. Procedure of the Practical

**(i)** Start the system and log in to the computer.

**(ii)** Open PostgreSQL software.

**iii) Create and select the database.**

```
create database CompanyDB;
```

**(iv) Create table using DDL command.**

```
create table customer_orders(
```

```
order_id serial primary key,
```

```
customer_name varchar(20),
```

```
product varchar(20),
```

```
quantity int,
```

```
price numeric(10,2),
```

```
order_date date
```

```
);
```

---

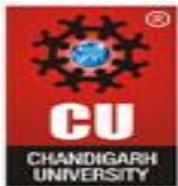
**(v) Insert records into the table.**

```
insert into customer_orders(customer_name,product,quantity,price,order_date) values
```

```
('Amit', 'Laptop', 1, 55000, '2025-01-05'),
```

```
('Amit', 'Mouse', 2, 800, '2025-01-06'),
```

```
('Riya', 'Mobile', 1, 22000, '2025-01-10'),
```



('Riya', 'Headphones', 1, 2000, '2025-01-10'),

('Karan', 'Laptop', 1, 60000, '2025-02-02'),

('Karan', 'Keyboard', 1, 1500, '2025-02-05'),

('Neha', 'Mobile', 2, 21000, '2025-02-15'),

('Neha', 'Charger', 3, 900, '2025-02-18');

### (vi) Display all records.

```
select * from customer_orders;
```

	order_id [PK] integer	customer_name character varying (20)	product character varying (20)	quantity integer	price numeric (10,2)	order_date date
1	1	Amit	Laptop	1	55000.00	2025-01-05
2	2	Amit	Mouse	2	800.00	2025-01-06
3	3	Riya	Mobile	1	22000.00	2025-01-10
4	4	Riya	Headphones	1	2000.00	2025-01-10
5	5	Karan	Laptop	1	60000.00	2025-02-02
6	6	Karan	Keyboard	1	1500.00	2025-02-05
7	7	Neha	Mobile	2	21000.00	2025-02-15
8	8	Neha	Charger	3	900.00	2025-02-18
9	9	Arnab	Laptop	1	55000.00	2025-01-05
10	10	Deepraj	Mouse	2	800.00	2025-01-06
11	11	Riya	Mobile	1	22000.00	2025-01-10
12	12	Raima	Headphones	1	2000.00	2025-01-10
13	13	Karan	Laptop	1	60000.00	2025-02-02
14	14	Subham	Keyboard	1	1500.00	2025-02-05
15	15	Nehashis	Mobile	2	21000.00	2025-02-15
16	16	Neha	Charger	3	900.00	2025-02-18

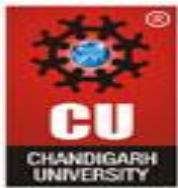
### (vii) Filtering Data Using WHERE clause.

```
select order_id, customer_name, product, quantity, price
```

```
from customer_orders
```

```
where price > 20000;
```

	order_id [PK] integer	customer_name character varying (20)	product character varying (20)	quantity integer	price numeric (10,2)
1	1	Amit	Laptop	1	55000.00
2	3	Riya	Mobile	1	22000.00
3	5	Karan	Laptop	1	60000.00
4	7	Neha	Mobile	2	21000.00
5	9	Arnab	Laptop	1	55000.00
6	11	Riya	Mobile	1	22000.00
7	13	Karan	Laptop	1	60000.00
8	15	Nehashis	Mobile	2	21000.00



### (viii) Sorting Query Results.

#### Ascending Order

```
select order_id, customer_name, product, quantity, price  
from customer_orders  
where price > 20000  
order by price;
```

	order_id [PK] integer ↴	customer_name character varying (20) ↴	product character varying (20) ↴	quantity integer ↴	price numeric (10,2) ↴
1	15	Nehashis	Mobile	2	21000.00
2	7	Neha	Mobile	2	21000.00
3	3	Riya	Mobile	1	22000.00
4	11	Riya	Mobile	1	22000.00
5	9	Arnab	Laptop	1	55000.00
6	1	Amit	Laptop	1	55000.00
7	13	Karan	Laptop	1	60000.00
8	5	Karan	Laptop	1	60000.00

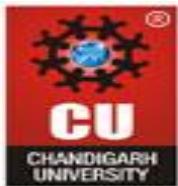
#### Descending Order

```
select order_id, customer_name, product, quantity, price  
from customer_orders  
where price > 20000  
order by price desc;
```

	order_id [PK] integer ↴	customer_name character varying (20) ↴	product character varying (20) ↴	quantity integer ↴	price numeric (10,2) ↴
1	5	Karan	Laptop	1	60000.00
2	13	Karan	Laptop	1	60000.00
3	1	Amit	Laptop	1	55000.00
4	9	Arnab	Laptop	1	55000.00
5	11	Riya	Mobile	1	22000.00
6	3	Riya	Mobile	1	22000.00
7	15	Nehashis	Mobile	2	21000.00
8	7	Neha	Mobile	2	21000.00

### (ix) Grouping Data for Aggregation.

```
select product, count(*) as total_product_sale  
from customer_orders
```



group by product;

	product character varying (20)	total_product_sale bigint
1	Charger	2
2	Mobile	4
3	Mouse	2
4	Keyboard	2
5	Laptop	4
6	Headphones	2

**(x) Applying conditions on aggregated data (HAVING).**

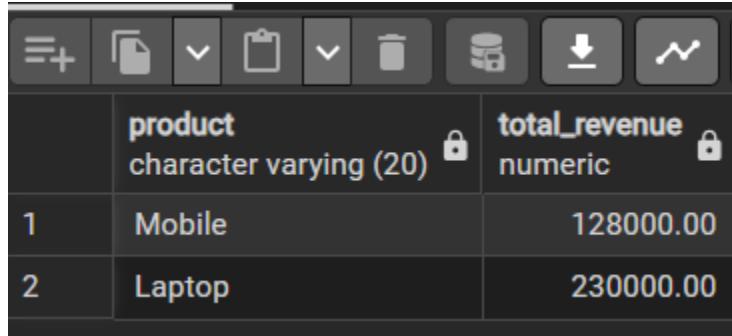
select product,

sum(quantity\*price) as total\_revenue

from customer\_orders

group by product

having sum(quantity\*price) > 50000;



	product character varying (20)	total_revenue numeric
1	Mobile	128000.00
2	Laptop	230000.00

**(xi) Using WHERE and HAVING together.**

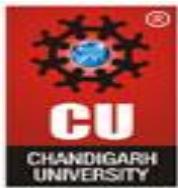
select product, sum(quantity\*price) as total\_revenue

from customer\_orders

where order\_date >= '2025-01-01'

group by product

having sum(quantity\*price) > 50000;



	product character varying (20) 	total_revenue numeric 
1	Mobile	128000.00
2	Laptop	230000.00

## 6. I/O Analysis (Input / Output)

### Input:

- Customer order details
- Filtering, sorting, grouping, and aggregation queries

### Output:

- Filtered customer records
- Sorted result sets
- Group-wise sales summary
- Aggregated revenue reports

(Screenshots of execution and output attached)

## 7. Learning Outcomes

- Students understand how data can be filtered to retrieve only relevant records.
- Students learn how sorting improves readability and usefulness of reports.
- Students gain the ability to group data for analytical purposes.
- Students clearly differentiate between WHERE and HAVING clauses.
- Students develop confidence in writing analytical SQL queries.
- Students are better prepared for SQL-based placement and interview questions.