

1. 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.

2. Objective of the Session

- 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().

3. Practical / Experiment Steps

1. Display the department name and the average salary of employees for each department.
2. Consider only those employees whose salary is greater than 20,000.
3. Display only those departments where the average salary is greater than 30,000.
4. Arrange the final output in descending order of average salary.

4. Procedure of the Practical

- (1) Start the system and log in to the computer.
- (2) Open PgAdmin (PostgreSQL).
- (3) Create or select the required database (e.g., lab_db).
- (4) Create the EMPLOYEE table using the given schema.
- (5) Insert sample data into the EMPLOYEE table.
- (6) Execute the queries step-by-step according to the practical steps.
- (7) Verify the output after each query execution.
- (8) Capture screenshots of execution and results for record.
- (9) Save the work and upload worksheet (Word + PDF) on GitHub.

5. I/O Analysis (Input / Output Analysis)

Input: SQL commands and queries executed in PgAdmin (table creation, insertion, and SELECT queries).

Output: Result tables displayed in PgAdmin showing department-wise average salary after applying WHERE, HAVING, and ORDER BY clauses.

SQL Implementation (PgAdmin / PostgreSQL)

A) Create Database (Optional):

```
CREATE DATABASE lab_db;
```

B) Create Table:

```
CREATE TABLE employee (  
    emp_id          INT PRIMARY KEY,  
    emp_name        VARCHAR(50),  
    department      VARCHAR(50),  
    salary          NUMERIC(10,2),  
    joining_date    DATE  
);
```

C) Insert Sample Records:

```
INSERT INTO employee (emp_id, emp_name, department, salary,  
joining_date) VALUES  
(101, 'Amit Sharma', 'IT', 45000, '2022-01-10'),  
(102, 'Neha Verma', 'HR', 22000, '2021-03-15'),  
(103, 'Rahul Singh', 'IT', 30000, '2020-06-20'),  
(104, 'Priya Mehta', 'Finance', 55000, '2019-09-05'),  
(105, 'Karan Gupta', 'HR', 18000, '2023-02-12'),  
(106, 'Sneha Kapoor', 'Finance', 28000, '2020-11-25'),  
(107, 'Rohit Jain', 'Sales', 35000, '2021-07-30'),  
(108, 'Ananya Joshi', 'Sales', 15000, '2022-12-01'),  
(109, 'Vikram Rao', 'IT', 25000, '2022-04-18');
```

Step 1 Query:

```
SELECT department, AVG(salary) AS avg_salary  
FROM employee  
GROUP BY department;
```

Step 2 Query:

```
SELECT department, AVG(salary) AS avg_salary  
FROM employee  
WHERE salary > 20000  
GROUP BY department;
```

Step 3 Query:

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
WHERE salary > 20000
GROUP BY department
HAVING AVG(salary) > 30000;
```

Step 4 Query (Final Output):

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
WHERE salary > 20000
GROUP BY department
HAVING AVG(salary) > 30000
ORDER BY avg_salary DESC;
```

6. Learning Outcome

- Understood the syntax and usage of SQL SELECT statements.
- Gained practical knowledge of WHERE clause for filtering rows.
- Learned grouping operations using GROUP BY clause.
- Applied HAVING clause to filter grouped results.
- Sorted query outputs using ORDER BY clause.
- Got hands-on experience in PostgreSQL execution using PgAdmin.

7. Screenshots

```
CREATE TABLE EMPLOYEE(
    emp_id int,
    emp_name varchar(20),
    department varchar(20),
    salary int,
    joining_date date
)
```

```
INSERT INTO employee (emp_id, emp_name, department, salary, joining_date) VALUES
(101, 'Amit Sharma', 'IT', 45000, '2022-01-10'),
(102, 'Neha Verma', 'HR', 22000, '2021-03-15'),
(103, 'Rahul Singh', 'IT', 30000, '2020-06-20'),
(104, 'Priya Mehta', 'Finance', 55000, '2019-09-05'),
(105, 'Karan Gupta', 'HR', 18000, '2023-02-12'),
(106, 'Sneha Kapoor', 'Finance', 28000, '2020-11-25'),
(107, 'Rohit Jain', 'Sales', 35000, '2021-07-30'),
(108, 'Ananya Joshi', 'Sales', 15000, '2022-12-01'),
(109, 'Vikram Rao', 'IT', 25000, '2022-04-18');
```

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
GROUP BY department;
```

Data Output Messages Notifications		
Showing rows: 1 to 11		
	department character varying (20)	avg_salary numeric
1	Marketing	62000.000000000000
2	Operations	45000.000000000000
3	Finance	59333.333333333333
4	Design	52000.000000000000
5	Human Resources	58000.000000000000
6	IT Support	35000.000000000000
7	Engineering	81000.000000000000
8	Quality Assurance	42000.000000000000
9	Sales	26000.000000000000
10	IT	33333.333333333333
11	HR	20000.000000000000

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
WHERE salary > 20000
GROUP BY department
HAVING AVG(salary) > 30000;
```

Data Output Messages Notifications		
Showing rows: 1 to 11		
	department character varying (20)	avg_salary numeric
1	Marketing	62000.000000000000
2	Operations	45000.000000000000
3	Finance	59333.333333333333
4	Design	52000.000000000000
5	Human Resources	58000.000000000000
6	IT Support	35000.000000000000
7	Engineering	81000.000000000000
8	Quality Assurance	42000.000000000000
9	Sales	31500.000000000000
10	IT	33333.333333333333
11	HR	22000.000000000000

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
WHERE salary > 20000
GROUP BY department;
```

Data OutputMessagesNotifications

SQL

Showing rows: 1 to 10

	department character varying (20)	avg_salary numeric
1	Marketing	62000.000000000000
2	Operations	45000.000000000000
3	Finance	59333.333333333333
4	Design	52000.000000000000
5	Human Resources	58000.000000000000
6	IT Support	35000.000000000000
7	Engineering	81000.000000000000
8	Quality Assurance	42000.000000000000
9	Sales	31500.000000000000
10	IT	33333.333333333333

```
SELECT department, AVG(salary) AS avg_salary
FROM employee
WHERE salary > 20000
GROUP BY department
HAVING AVG(salary) > 30000
ORDER BY avg_salary DESC;
```

Data OutputMessagesNotifications

SQL

Showing rows: 1 to 10

	department character varying (20)	avg_salary numeric
1	Engineering	81000.000000000000
2	Marketing	62000.000000000000
3	Finance	59333.333333333333
4	Human Resources	58000.000000000000
5	Design	52000.000000000000
6	Operations	45000.000000000000
7	Quality Assurance	42000.000000000000
8	IT Support	35000.000000000000
9	IT	33333.333333333333
10	Sales	31500.000000000000