

NAME: Manan Verma

SEMESTER: 4th

SECTION: 24_AIT_KRG-G1

UID: 24BAI70008

SUBJECT CODE: 24CSH-298

FACULTY'S NAME: Mr .SHALABH BHATIA

Lab Experiment 2

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.

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()**.

Software Requirements

- **Database:**
 - Oracle Database Express Edition (Oracle XE)
 - PostgreSQL Database (PgAdmin)

Experiment Question :

An organization maintains an **EMPLOYEE** table to store details of its employees.
The structure of the table is as follows:

Column Name	Data Type
emp_id	NUMBER
emp_name	VARCHAR
Department	VARCHAR
Salary	NUMBER
joining_date	DATE

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**.

Note: Use the following SQL clauses in your query:

- **WHERE**
- **GROUP BY**
- **HAVING**
- **ORDER BY**

Code:



```
CREATE TABLE employee (  
    emp_id INT,  
    emp_name VARCHAR(50),  
    department VARCHAR(50),  
    salary INT,  
    joining_date DATE  
);
```

```
INSERT INTO employee VALUES  
(1, 'Amit', 'HR', 25000, '2022-01-10'),  
(2, 'Neha', 'HR', 32000, '2021-03-15'),  
(3, 'Rahul', 'IT', 40000, '2020-06-20'),  
(4, 'Sneha', 'IT', 28000, '2022-08-05'),  
(5, 'Arjun', 'Finance', 35000, '2019-11-12'),  
(6, 'Pooja', 'Finance', 18000, '2023-02-18');
```


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

ORDER BY average_salary DESC;


Output:

	department character varying (50) 	average_salary numeric 
1	Finance	35000.000000000000
2	IT	34000.000000000000

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

Query Query History 

```
1 CREATE TABLE employee (
2     emp_id INT,
3     emp_name VARCHAR(50),
4     department VARCHAR(50),
5     salary INT,
6     joining_date DATE
7 );
8
9 INSERT INTO employee VALUES
10 (1, 'Amit', 'HR', 25000, '2022-01-10'),
11 (2, 'Neha', 'HR', 32000, '2021-03-15'),
12 (3, 'Rahul', 'IT', 40000, '2020-06-20'),
13 (4, 'Sneha', 'IT', 28000, '2022-08-05'),
14 (5, 'Arjun', 'Finance', 35000, '2019-11-12'),
15 (6, 'Pooja', 'Finance', 18000, '2023-02-18');
16
17
18 SELECT department, AVG(salary) AS average_salary
19 FROM employee
20 WHERE salary > 20000
21 GROUP BY department
22 HAVING AVG(salary) > 30000
23 ORDER BY average_salary DESC;
```

Data Output Messages Notifications 

Learning Outcome

After completing this experiment, I was able to:

- Filter records using the **WHERE** clause.
- Group records using **GROUP BY**.
- Apply conditions on grouped data using **HAVING**.
- Sort query results using **ORDER BY**.