



Apex Institute of Technology

Department of Computer Science & Engineering

Student Name: KUSHAGRA VASHISTHA

UID:24BAI70915

Branch: Computer Science & Engineering (AIML) Section/Group: 24AIT-KRG_G2

Semester:4th

Subject Name: DBMS

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



Apex Institute of Technology

Department of Computer Science & Engineering

- (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.

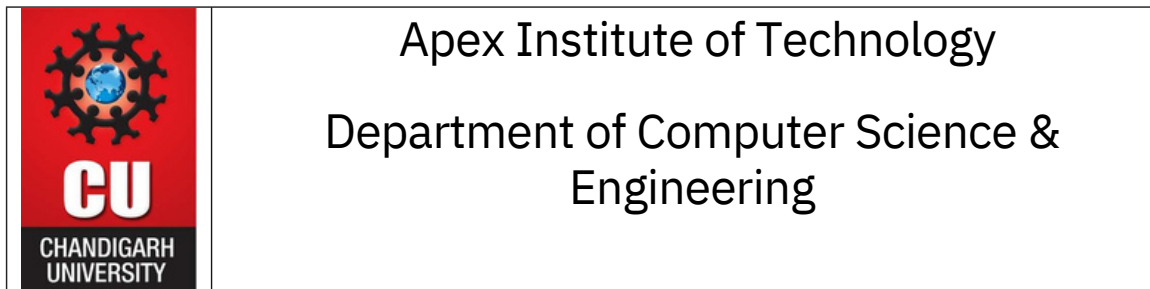
SQL Implementation (PgAdmin / PostgreSQL)

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 INT,  
    joining_date DATE  
);
```



C) Insert SampleRecords:

INSERT INTO employee VALUES

```
(1, 'Amit', 'HR', 25000, '2022-01-15'),
(2, 'Neha', 'HR', 35000, '2021-03-10'),
(3, 'Rahul', 'IT', 45000, '2020-07-22'),
(4, 'Sneha', 'IT', 55000, '2019-11-05'),
(5, 'Karan', 'Finance', 18000, '2023-02-01'),
(6, 'Pooja', 'Finance', 42000, '2021-06-18'),
(7, 'Rohit', 'Sales', 30000, '2020-09-12'),
(8, 'Anjali', 'Sales', 48000, '2018-04-25');
```

Step 1 Query:

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

Step 2 Query:

```
SELECT * FROM
    employee
WHERE
    salary > 20000;
```

Step 3 Query:

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

Step 4 Query (Final Output):

```
SELECT
    department,
    AVG(salary) AS average_salary
FROM
    employee
```

GROUP BY
department
ORDER BY
average_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 PRIMARY KEY,
    emp_name VARCHAR(50),
    department VARCHAR(50),
    salary INT,
    joining_date DATE
);
```

CREATE TABLE

Query returned successfully in 47 msec.

```
INSERT INTO employee VALUES
(1, 'Amit', 'HR', 25000, '2022-01-15'),
(2, 'Neha', 'HR', 35000, '2021-03-10'),
(3, 'Rahul', 'IT', 45000, '2020-07-22'),
(4, 'Sneha', 'IT', 55000, '2019-11-05'),
(5, 'Karan', 'Finance', 18000, '2023-02-01'),
(6, 'Pooja', 'Finance', 42000, '2021-06-18'),
(7, 'Rohit', 'Sales', 30000, '2020-09-12'),
(8, 'Anjali', 'Sales', 48000, '2018-04-25');
```

	emp_id [PK] integer	emp_name character varying (50)	department character varying (50)	salary integer	joining_date date
1	1	Amit	HR	25000	2022-01-15
2	2	Neha	HR	35000	2021-03-10
3	3	Rahul	IT	45000	2020-07-22
4	4	Sneha	IT	55000	2019-11-05
5	5	Karan	Finance	18000	2023-02-01
6	6	Pooja	Finance	42000	2021-06-18
7	7	Rohit	Sales	30000	2020-09-12
8	8	Anjali	Sales	48000	2018-04-25



Apex Institute of Technology

Department of Computer Science & Engineering

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

	department character varying (50)	average_salary numeric
1	Finance	30000.000000000000
2	Sales	39000.000000000000
3	IT	50000.000000000000
4	HR	30000.000000000000

```
SELECT
    *
FROM
    employee
WHERE
    salary > 20000;
```

	emp_id [PK] integer	emp_name character varying (50)	department character varying (50)	salary integer	joining_date date
1	1	Amit	HR	25000	2022-01-15
2	2	Neha	HR	35000	2021-03-10
3	3	Rahul	IT	45000	2020-07-22
4	4	Sneha	IT	55000	2019-11-05
5	6	Pooja	Finance	42000	2021-06-18
6	7	Rohit	Sales	30000	2020-09-12
7	8	Anjali	Sales	48000	2018-04-25

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

	department character varying (50)	average_salary numeric
1	IT	50000.000000000000
2	Finance	42000.000000000000
3	Sales	39000.000000000000

```

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

```

	department character varying (50)	average_salary numeric
1	Sales	39000.000000000000
2	IT	50000.000000000000

```

SELECT
    department,
    AVG(salary) AS average_salary
FROM
    employee
GROUP BY
    department
ORDER BY
    average_salary DESC;

```

	department character varying (50)	average_salary numeric
1	IT	50000.000000000000
2	Sales	39000.000000000000
3	Finance	30000.000000000000
4	HR	30000.000000000000