

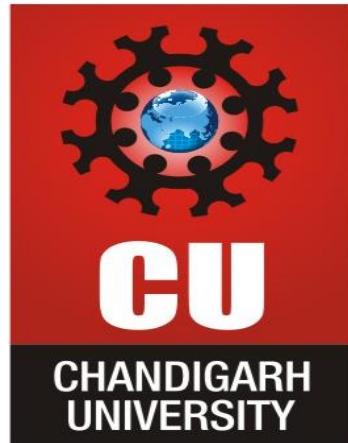
```

23 CREATE TABLE LIBRARY_VISITORS(
24   USER_ID INT PRIMARY KEY,
25   USER_NAME VARCHAR(40) NOT NULL,
26   AGE INT CHECK(AGE>=17) NOT NULL,
27   EMAIL VARCHAR(40) NOT NULL UNIQUE
28 )
29
30 INSERT INTO LIBRARY_VISITORS(USER_ID,USER_NAME,AGE,EMAIL)
31 VALUES(501,'SIMRAN',19,'SIMRAN@GMAIL.COM')
32
33 select * from library_visitors
34
35 CREATE TABLE BOOK_ISSUE(
36   BOOK_ISSUE_ID INT PRIMARY KEY

```

Data Output Messages Notifications

	user_id [PK] integer	user_name character varying (40)	age integer	email character varying (20)
1	501	SIMRAN	19	SIMRAN@GMAIL.COM



NAME :Simran Chouhan

SEMESTER:4th

SECTION:24_AIT_KRG-G1

UID:24BAI70124

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:

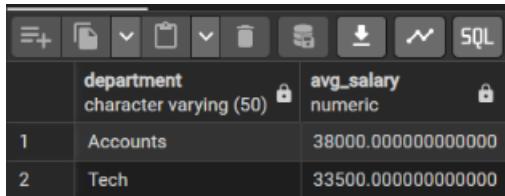
1. Display the **department name** and the **average salary** of employees for each department.
2. Consider **only those employees whose salary is greater than 20,000**.
3. Display **only those departments** where the **average salary is greater than 30,000**.
4. Arrange the final output in **descending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

Note: Use the following SQL clauses in your query:

- WHERE
 - GROUP BY
 - HAVING
 - ORDER BY
-

CODE:

OUTPUT:



	department character varying (50)	avg_salary numeric
1	Accounts	38000.00000000000000
2	Tech	33500.00000000000000

	emp_id integer	emp_name character varying (50)	department character varying (50)	salary integer	joining_date date
1	101	Karan	Sales	27000	2021-07-12
2	102	Riya	Sales	31000	2020-09-25
3	103	Ankit	Tech	45000	2019-04-18
4	104	Meera	Tech	22000	2022-12-03
5	105	Vikas	Accounts	38000	2020-01-30
6	106	Nisha	Accounts	19000	2023-06-10

The screenshot shows a software interface for managing database tables. At the top, there are three tabs: "Data Output" (which is selected), "Messages", and "Notifications". Below the tabs is a toolbar with several icons: a plus sign inside a box, a file icon, a dropdown arrow, a clipboard icon, another dropdown arrow, a trash bin icon, a database icon, a download arrow icon, a chart icon, and an "SQL" button. Underneath the toolbar, there are two rows of table metadata. The first row contains a column header and a column detail. The second row contains a column header and a column detail. Both rows have a lock icon at the end.

	department	avg_sal
	character varying (20)	numeric (10,2)

Learning Outcome

After completing this experiment, students will be 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**.