

**Laboratory Manual 24CS202P: Database Management Systems Laboratory**

**DEPARTMENT of COMPUTER Sc. & ENGINEERING SCHOOL OF TECHNOLOGY**

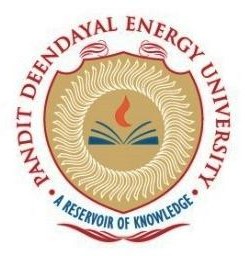
**Name of Student: Panav Brijesh Patel**

**Roll No: 24BCP128**

**Branch: CSE**

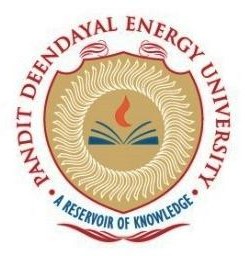
**Sem: 3rd**

**Academic-Year: 2025-26**



Pandit Deendayal Energy University

Raisan, Gandhinagar – 380 007, Gujarat, India



Department of Computer Science & Engineering

Certificate

This is to certify that

Mr./Ms. Panav Brijesh Patel Roll no. 24bcp128

Exam No. of 3rd Semester Degree course in Computer Science and Engineering has satisfactorily completed his/her term work in Database Management System Lab (24CS202P) subject during the semester from to

at School of Technology, PDEU.

Date of Submission:

##### Signature:

Faculty In-charge Head of Department

*Index*

Name: Roll No: Exam No:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Experiment Title** | **Pages** | | **Date of Completion** | **Marks (out of 10)** | **Sign.** |
| **From** | **To** |
| 1 | Dataset Creation and Updating using File Handling Program |  |  |  |  |  |
| 2 | DDL (Data Definition Language) commands |  |  |  |  |  |
| 3 | DML commands with constraints |  |  |  |  |  |
| 4 | DDL (Data Definition Language) commands with Data Constraints |  |  |  |  |  |
| 5 | DCL Commands in SQL: Transaction control commands, Commit,  Rollback, Save point |  |  |  |  |  |
| 6 | Use of Inbuilt functions and relational algebra  operation |  |  |  |  |  |
| 7 | Nested SQL Queries or Subqueries |  |  |  |  |  |
| 8 | Group by & having clause, and Joins in SQL |  |  |  |  |  |
| 9 | Implementation of  Embedded SQL, PL SQL Concepts. |  |  |  |  |  |
| 10 | Implementation of Cursors, Stored Procedures, Stored  Function, Triggers. |  |  |  |  |  |

**Experiment 1:**

##### TITLE: Dataset Creation and Updating using File Handling Program.

**Objective:** How to store and retrieve dataset in table format using file handling programming.

### Aim:

To create and update a dataset using a file handling program and import it into MySQL database.

### Theory:

File handling allows us to read and write data into files. We can use Python/Java/C/C++ to create and update text or CSV files, which can then be imported into a MySQL table using the `LOAD DATA INFILE` command.

### SQL Code:

***LOAD DATA INFILE 'data.csv'***

***INTO TABLE student***

***FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' IGNORE 1 ROWS;***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

# Exercise

Design a program (using C, C++, Java or Python) that performs Create, Read, Update, Delete record using file handling. The file store information about students in the following format:

##### Roll Number (unique identifier), Name, Age, and Department

All data should be stored in a text file (students.txt) where each record is stored on a new line. The initial dataset can be downloaded from Kaggle or other sources for any other records like Employee Dataset, HR Analytics Employee Dataset etc.

##### Tasks to be carried out in LAB

1. **Create a Record:**

Add a new student record to the file. Ensure that the roll number is unique (no duplicates).

##### Read Records:

Display all student records stored in the file in a tabular format. Allow searching for a record by roll number.

##### Update a Record:

Update the details of a student by their roll number. Ensure the updated data is saved in the file.

##### Delete a Record:

Delete a student record using their roll number. Ensure the file reflects the changes after deletion.

##### Exit Program:

Provide an option to exit the program.

**Menu:**

1. **Add a Student Record**
2. **View All Records**
3. **Search a Record by Roll Number**
4. **Update a Record**
5. **Delete a Record**
6. **Exit**

**Enter your choice:**

**Experiment 2**

##### TITLE: DDL (Data Definition Language) commands

**Objective:** To understand the concept of designing issues related to the database with creating, populating the tables.

### Aim:

To use various DDL commands like CREATE, ALTER, and DROP.

### Theory:

DDL commands are used to define the database structure or schema. They do not manipulate data but affect table definitions and structures.

### SQL Code:

***CREATE TABLE employee ( id INT PRIMARY KEY, name VARCHAR(50),***

***salary DECIMAL(10, 2)***

***);***

***ALTER TABLE employee ADD department VARCHAR(30); DROP TABLE employee;***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

# Exercise

1. **Create the tables described below: Table name: CLIENT\_MASTER Description:** used to store client information.

|  |  |  |
| --- | --- | --- |
| **Column name** | **data type** | **Size** |
| CLIENTNO | Varchar | 6 |
| NAME | Varchar | 20 |
| ADDRESS 1 | Varchar | 30 |

|  |  |  |
| --- | --- | --- |
| ADDRESS 2 | Varchar | 30 |
| CITY | Varchar | 15 |
| PINCODE | Integer |  |
| STATE | Varchar | 15 |
| BALDUE | decimal | 10,2 |

**Table Name: PRODUCT\_MASTER Description:** used to store product information

|  |  |  |
| --- | --- | --- |
| **Column name** | **data type** | **Size** |
| PRODUCTNO | Varchar | 6 |
| DESCRIPTION | Varchar | 15 |
| PROFITPERCENT | Decimal | 4,2 |
| UNIT MEASURE | Varchar | 10 |
| QTYONHAND | Integer |  |
| REORDERL VL | Integer |  |
| SELLPRICE | Decimal | 8,2 |
| COSTPRICE | Decimal | 8,2 |

##### Table Name: SALESMAN\_MASTER

**Description:** Used to store salesman information working for the company.

|  |  |  |
| --- | --- | --- |
| **Column name** | **data type** | **Size** |
| SALESMANNO | Varchar | 6 |
| SALESMANNAME | Varchar | 20 |
| ADDRESS 1 | Varchar | 30 |

|  |  |  |
| --- | --- | --- |
| ADDRESS 2 | Varchar | 30 |
| CITY | Varchar | 20 |
| PINCODE | Integer |  |
| STATE | Varchar | 20 |
| SALAMT | Real |  |
| TGTTOGET | Decimal |  |
| YTDSALES | Double | 6,2 |
| REMARKS | Varchar | 60 |

##### Insert the following data into their respective tables:

* 1. Data for **CLIENT\_MASTER table**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Client no | Name | city | Pincode | state | BalDue |
| C00001 | Ivan bayross | Mumbai | 400054 | Maharashtra | 15000 |
| C00002 | Mamta muzumdar | Madras | 780001 | Tamil nadu | 0 |
| C00003 | Chhaya bankar | Mumbai | 400057 | Maharashtra | 5000 |
| C00004 | Ashwini joshi | Bangalore | 560001 | Karnataka | 0 |
| C00005 | Hansel colaco | Mumbai | 400060 | Maharashtra | 2000 |
| C00006 | Deepak sharma | Mangalore | 560050 | Karnataka | 0 |

* 1. Data for PRODUCT\_**MASTER** table:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ProductNo | Description | Profit percent | Unit measure | Qtyonhand | RecorderLvl | SellPrice | CostPrice |
| P00001 | T-Shirt | 5 | Piece | 200 | 50 | 350 | 250 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P0345 | Shirts | 6 | Piece | 150 | 50 | 500 | 350 |
| P06734 | Cotton jeans | 5 | Piece | 100 | 20 | 600 | 450 |
| P07865 | Jeans | 5 | Piece | 100 | 20 | 750 | 500 |
| P07868 | Trousers | 2 | Piece | 150 | 50 | 850 | 550 |
| P07885 | Pull Overs | 2.5 | Piece | 80 | 30 | 700 | 450 |
| P07965 | Denim jeans | 4 | Piece | 100 | 40 | 350 | 250 |
| P07975 | Lycra tops | 5 | Piece | 70 | 30 | 300 | 175 |
| P08865 | Skirts | 5 | Piece | 75 | 30 | 450 | 300 |

* 1. Data for **SALESMAN\_MASTER** table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SalesmanNo** | **Name** | **Address1** | **Address2** | **City** | **PinCode** | **State** |
| S00001 | Aman | A/14 | Worli | Mumbai | 400002 | Maharashtra |
| S00002 | Omkar | 65 | Nariman | Mumbai | 400001 | Maharashtra |
| S00003 | Raj | P-7 | Bandra | Mumbai | 400032 | Maharashtra |
| S00004 | Ashish | A/5 | Juhu | Mumbai | 400044 | Maharashtr(a |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SalesmanNo** | **SalAmt** | **TgtToGet** | **YtdSales** | **Remarks** |
| S00001 | 3000 | 100 | 50 | Good |
| S00002 | 3000 | 200 | 100 | Good |
| S00003 | 3000 | 200 | 100 | Good |
| S00004 | 3500 | 200 | 150 | Good |

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Experiment 3**

##### Title: DML commands with constraints

**Objective: -** To understand the concept of different DML commands.

### Aim:

To perform DML operations with constraints like NOT NULL, UNIQUE, CHECK.

### Theory:

DML allows manipulation of data within tables. Constraints enforce data integrity.

### SQL Code:

***CREATE TABLE student ( id INT PRIMARY KEY,***

***name VARCHAR(50) NOT NULL, age INT CHECK (age >= 18), email VARCHAR(50) UNIQUE***

***);***

***INSERT INTO student VALUES (1, 'John', 20,*** [***'john@example.com');***](mailto:%27john@example.com)

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

##### Exercise

Exercise on retrieving records from a table.

1. Find out the names of all the clients.
2. Retrieve the entire contents of the Client\_Master table.
3. Retrieve the list of names,city and the state of all the clients.
4. List the various products available from the Product\_Master table.
5. List all the clients who are located in Mumbai.
6. Find the names of salesman who have a salary equal to Rs.3000.
   1. Exercise on updating records in a table
      1. Change the city of ClientNo ‘C00005’ to ‘Bangalore’.
      2. Change the BalDue of ClientNo ‘C00001’ to Rs.1000.
      3. Change the cost price of ‘Trousers’ to rs.950.00.
      4. Change the city of the salesman to Pune.
   2. Exercise on deleting records in a table
      1. Delete all salesman from the Salesman\_Master whose salaries are equal to Rs.3500.
   3. b. Delete all products from Product\_Master where the quantity on hand is equal to 100.

c. Delete from Client\_Master where the column state holds the value ‘Tamil Nadu’.

* 1. Exercise on altering the table structure
     1. Add a column called ‘Telephone’ of data type integer to the Client\_Master table.
     2. Change the size of SellPrice column in Product \_Master to 10, 2.
  2. Exercise on deleting the table structure along with the data
     1. Destroy the table Client\_Master along with its data.
  3. Exercise on renaming the table
     1. Change the name of the Salesman\_Master to sman\_mast.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#### EXPERIMENT-4

##### TITLE: DDL (Data Definition Language) commands with Data Constraints

**Objective:** To understand the concept of data constraints that is enforced on data being stored in the table. Focus on Primary Key and the Foreign Key

### Aim:

To define tables using constraints such as PRIMARY KEY, FOREIGN KEY, etc.

### Theory:

Constraints are used to maintain accuracy and integrity of data.

### SQL Code:

***CREATE TABLE department ( dept\_id INT PRIMARY KEY, dept\_name VARCHAR(30)***

***);***

***CREATE TABLE employee ( emp\_id INT PRIMARY KEY, emp\_name VARCHAR(50), dept\_id INT,***

***FOREIGN KEY (dept\_id) REFERENCES department(dept\_id)***

***);***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

## Exercise

##### Create the tables described below: Table name: CLIENT\_MASTER\_1

**Description:** used to store client information.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column name** | **data type** | **Size** | **Constraints** |
| CLIENTNO | Varchar | 6 | Primary key / first letter must start with ‘C’ |
| NAME | Varchar | 20 | Not Null |
| ADDRESS 1 | Varchar | 30 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ADDRESS 2 | Varchar | 30 |  |
| CITY | Varchar | 15 |  |
| PINCODE | Integer | 8 |  |
| STATE | Varchar | 15 |  |
| BALDUE | Decimal | 10,2 |  |

**Table Name: PRODUCT\_MASTER\_1 Description:** used to store product information

|  |  |  |  |
| --- | --- | --- | --- |
| **Column name** | **data type** | **Size** | **Attributes** |
| PRODUCTNO | Varchar | 6 | Primary Key/ first letter must start with ‘P’ |
| DESCRIPTION | Varchar | 15 | Not Null |
| PROFITPERCENT | Decimal | 4,2 | Not Null |
| UNIT MEASURE | Varchar | 10 | Not Null |
| QTYONHAND | Integer | 8 | Not Null |
| REORDERL VL | Integer | 8 | Not Null |
| SELLPRICE | Decimal | 8,2 | Not Null |
| COSTPRICE | Decimal | 8,2 | Not Null |

##### Table Name: SALESMAN\_MASTER \_1

**Description:** used to store salesman information working for the company.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column name** | **data type** | **Size** | **Attributes** |
| SALESMANNO | Varchar | 6 | Primary Key/ first letter must start with ‘S’ |
| SALESMANNAME | Varchar | 20 | Not Null |
| ADDRESS 1 | Varchar | 30 | Not Null |

|  |  |  |  |
| --- | --- | --- | --- |
| ADDRESS 2 | Varchar | 30 |  |
| CITY | Varchar | 20 |  |
| PINCODE | Integer | 8 |  |
| STATE | Varchar | 20 |  |
| SALAMT | Real | 8,2 | Not Null , Cannot be 0 |
| TGTTOGET | Decimal | 6,2 | Not Null , Cannot be 0 |
| YTDSALES | Double | 6,2 | Not Null |
| REMARKS | Varchar | 60 |  |

1. **Reinsert the data in these two tables based upon Lab 2.**
2. **Display the contents of each table.**

#### EXPERIMENT-5

##### TITLE: DDL (Data Definition Language) commands with Data Constraints

**Objective:** To understand the concept of data constraints that are enforced on data being stored in the table. Focus on Primary Key, The Foreign Key and constraints.

Review this diagram

### Aim:

To implement Data Control Language commands like GRANT and REVOKE.

### Theory:

DCL deals with rights, permissions, and other controls of the database system.

### SQL Code:

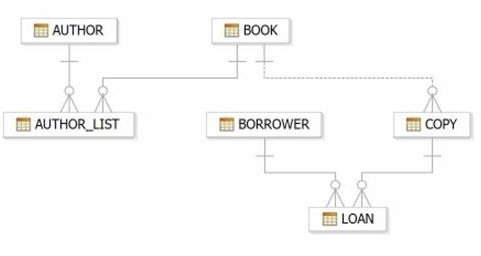
***GRANT SELECT, INSERT ON employee TO 'user1'@'localhost'; REVOKE INSERT ON employee FROM 'user1'@'localhost';***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.



## Exercise

1. Create table AUTHOR = {Author\_ID , Lastname, Firstname, Email, City, Country} Where:

Author\_ID – text data type, 5 characters, primary key Lastname – text data type, 15 characters, not null Firstname – text data type, 15 characters, not null Email – text data type, 40 characters,

City – text data type, 15 characters,

Country – text data type, 15 characters,

1. Create Table BOOK={ Book\_ID, Book\_Title, Copies) Where:

Book\_ID – text data type, 5 characters Primary Key Start with Character **B**

Book\_Title - Text data Type Not Null

Copies- No.of copies Data Type int always greater the 2

1. Create table AUTHOR\_LIST = {Author\_ID , Book\_ID , Role} Where:

Author\_ID – text data type, 5 characters, referenced by Author\_ID from AUTHOR table

Book\_ID – text data type, 5 characters Role – text data type, 15 characters

and primary key is Author\_ID, Book\_ID

1. Add four records in each tables AUTHOR, BOOK, BOOK\_LIST.
2. Alter structure of table AUTHOR\_LIST add the field Publisher data type of 30 Character.

#### EXPERIMENT- 6

**Title:** Use of Inbuilt functions and relational algebra operation and Transaction control commands, Commit, Rollback, save point

**Objective:** To understand the use of inbuilt function and relational algebra with SQL query.

### Aim:

To demonstrate use of Relational algebra, COMMIT, ROLLBACK, and SAVEPOINT.

### Theory:

Transaction control commands manage changes made by DML statements.

### SQL Code:

***START TRANSACTION;***

***INSERT INTO employee VALUES (101, 'Alice', 'HR'); SAVEPOINT sp1;***

***INSERT INTO employee VALUES (102, 'Bob', 'IT'); ROLLBACK TO sp1;***

***COMMIT;***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

##### Exercise

1. Consider the following table structure and attempt. Supplier-(scode,sname,scity,turnover)

Part-(pcode,weigh,color,cost,sellingprice) Supplier\_Part-(scode,pcode,qty)

* 1. Create tables
  2. Populate the table.

1. Write appropriate SQL Statement for the following:
2. Get the supplier number and part number in ascending order of supplier number.
3. Get the details of supplier who operate from Bombay with turnover 50.
4. Get the total number of supplier.
5. Get the part number weighing between 25 and 35.
6. Get the supplier number whose turnover is null.
7. Get the part number that cost 20, 30 or 40 rupees.
8. Get the total quantity of part 2 that is supplied.
9. Get the name of supplier who supply part 2.
10. Get the part number whose cost is greater than the average cost.
11. Get the supplier number and turnover in descending order of turnover.

#### EXPERIMENT-7:

**Use of Inbuilt functions and relational algebra operation Contd…**

**Objective:** To understand the use of inbuilt function and relational algebra with SQL query.

### Aim:

To use MySQL inbuilt functions and understand relational algebra equivalents.

### Theory:

MySQL provides aggregate functions like SUM(), AVG(), COUNT() etc., which map to relational algebra operations like projection and selection.

### SQL Code:

***SELECT AVG(salary) FROM employee;***

***SELECT \* FROM employee WHERE department = 'IT';***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

#### EXPERIMENT- 8

**TITLE:** Nested SQL Queries or Subqueries

**Objective:** To understand the use **SQL Subquery**

### Aim:

To implement nested queries (subqueries) in SQL.

### Theory:

A subquery is a query within another query, used for complex filtering or calculations.

### SQL Code:

***SELECT name FROM employee***

***WHERE salary > (SELECT AVG(salary) FROM employee);***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

**Exercise**

1. **Create the following two tables (EMP and DEPT) EMP TABLE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EMPNO DEPTNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM |
|  | | | | | | |
| 7369  20 | SMITH | CLERK | 7902 | 17-DEC-80 | 500 | 800 |
| 7499  30 | ALLEN | SALESMAN | 7698 | 20-FEB-81 | 1600 | 300 |
| 7521  30 | WARD | SALESMAN | 7698 | 22-FEB-81 | 1250 | 500 |
| 7566 JONES MANAGER 7839 02-APR-81 2975 20 | | | | | | |
| 7654  30 | MARTIN | SALESMAN | 7698 | 28-SEP-81 | 1250 | 1400 |
| 7698 BLAKE MANAGER 7839 01-MAY-81 2850 30 | | | | | | |
| 7782 CLARK MANAGER 7839 09-JUN-81 2450 10 | | | | | | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7788 | SCOTT | ANALYST | 7566 |  | 09-DEC-82 | 3000 |  | 20 |
|  | 7839 | KING | PRESIDENT | |  | 17-NOV-81 5000 | |  | 10 |
| 30 | 7844 | TURNER | SALESMAN | | 7698 | 08-SEP-81 | | 1500 | 0 |
|  | 7876 | ADAMS | CLERK | 7788 |  | 12-JAN-83 | 1100 |  | 20 |
|  | 7900 | JAMES | CLERK | 7698 |  | 03-DEC-81 | 950 |  | 30 |
|  | 7902 | FORD | ANALYST | 7566 |  | 03-DEC-81 | 3000 |  | 20 |
|  | 7934 | MILLER | CLERK | 7782 |  | 23-JAN-82 | 1300 |  | 10 |

#### DEPT TABLE

|  |  |  |
| --- | --- | --- |
| DEPTNO | DNAME | LOC |
|  | | |
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |

Write the Nested Queries for the following queries.

* 1. List the details of the emps whose Salaries more than the employee BLAKE.
  2. List the emps whose Jobs are same as ALLEN.
  3. List the Emps whose Sal is same as FORD or SMITH in desc order of Names.
  4. List the emps Whose Jobs are same as MILLER or Sal is more than ALLEN.
  5. Find the highest paid employee of sales department.
  6. List the employees who are senior to most recently hired employee working under king.

List the employees who are senior to most recently hired employee working under king.

select \* from emp where hiredate < (select max(hiredate) from emp where mgr in (select empno from emp where ename = 'KING')) ;

* 1. List the names of the emps who are getting the highest sal dept wise.

select e.ename,e.deptno from emp e where e.sal in (select max(sal) from emp group by deptno) ;

* 1. List the emps whose sal is equal to the average of max and minimum select \* from emp where sal =(select (max(sal)+min(sal))/2 from emp);
  2. List the emps who joined in the company on the same date.

select \* from emp e where hiredate in (select hiredate from emp where e.empno <> empno);

* 1. Find out the emps who joined in the company before their Managers.

select \* from emp e where hiredate < (select hiredate from emp where empno = e.mgr)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#### EXPERIMENT-9

**TITLE: Group by & having clause and Join in SQL Objective:** To understand the use of group by and having clause. To use GROUP BY, HAVING clauses and SQL JOIN operations.

### Theory:

GROUP BY is used to group rows with same values. HAVING is used to filter aggregated data. JOIN is used to combine rows from two or more tables.

### SQL Code:

***SELECT dept\_id, COUNT(\*) FROM employee***

***GROUP BY dept\_id HAVING COUNT(\*) > 1;***

***SELECT e.emp\_name, d.dept\_name FROM employee e***

***JOIN department d ON e.dept\_id = d.dept\_id;***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.

### Result:

Successfully executed as per the aim of the experiment.

##### Exercise

Write the SQL Queries for the following queries (use EMP and DEPT table of Exp 8).

1. List the Deptno where there are no emps.
2. List the No.of emp’s and Avg salary within each department for each job.
3. Find the maximum average salary drawn for each job except for ‘President’.
4. List the department details where at least two emps are working.
5. List the no. of emps in each department where the no. is more than 3.
6. List the names of the emps who are getting the highest sal dept wise.
7. List the Deptno and their average salaries for dept with the average salary less than the averages for all departments.
8. In addition refer Experiment 7 & 8 and execute the same questions by using join.

#### EXPERIMENT-10

##### TITLE: Joins in SQL

OBJECTIVE: SQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.

### Aim:

To demonstrate the use of Embedded SQL, PL/SQL blocks including Cursors, Procedures, Functions, and Triggers.

### Theory:

PL/SQL is Oracle's procedural extension to SQL. Cursors allow row-by-row processing, triggers respond to database events, and procedures/functions encapsulate reusable logic.

### SQL Code:

***DELIMITER //***

***CREATE PROCEDURE GetEmployee() BEGIN***

***SELECT \* FROM employee; END//***

***CREATE TRIGGER before\_insert\_employee BEFORE INSERT ON employee***

***FOR EACH ROW BEGIN***

***SET NEW.salary = IFNULL(NEW.salary, 10000); END//***

***DELIMITER ;***

### Output:

The expected output confirms successful execution of SQL commands and matches the theoretical concept.Result:

Successfully executed as per the aim of the experiment.