Practical 3

Aim: To perform the various Clauses in SQL.

Theory Required:

1. **ORDER BY:** The ORDER BY keyword is used to sort the result-set in ascending or descending order. The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

2. **GROUP BY:** The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country". The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

3. **HAVING:** The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Commands:

```
mysql> create database two;
Query OK, 1 row affected (0.01 sec)

mysql> use two;
Database changed
mysql> create table children(
    -> rollno int not null,
    -> age int not null,
    -> name varchar(100) not null,
    -> address varchar(200) not null,
    -> phoneno varchar(15) not null,
    -> primary key(rollno) );
Query OK, 0 rows affected (0.03 sec)

mysql> insert into children(rollno,age,name,address,phoneno)
    -> values
    -> (1,15,'Ankur','Sector 22, Gurgaon','9815678290'),
    -> (2,16,'Sameer','Sector 11, Gurgaon','9283373882'),
    -> (3,15,'Yash','Sector 07, Gurgaon','8800116789');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from children order by rollno desc;
+--------+-----+--------+--------------------+------------+
| rollno | age | name   | address            | phoneno    |
+--------+-----+--------+--------------------+------------+
|      3 |  15 | Yash   | Sector 07, Gurgaon | 8800116789 |
|      2 |  16 | Sameer | Sector 11, Gurgaon | 9283373882 |
|      1 |  15 | Ankur  | Sector 22, Gurgaon | 9815678290 |
+--------+-----+--------+--------------------+------------+
3 rows in set (0.00 sec)
```

```
mysql> select name,sum(age) from children
    -> group by name;
+--------+----------+
| name   | sum(age) |
+--------+----------+
| Ankur  |       15 |
| Sameer |       16 |
| Yash   |       15 |
+--------+----------+
3 rows in set (0.01 sec)

mysql> select name, sum(age) as age
    -> from children
    -> group by name
    -> having age >= 15;
+--------+------+
| name   | age  |
+--------+------+
| Ankur  |   15 |
| Sameer |   16 |
| Yash   |   15 |
+--------+------+
3 rows in set (0.00 sec)
```

Conclusion: Performed the various Clauses in SQL.

Open Ended Experiment

Aim: To implement cursor in PL/SQL

Theory Required:

When an SQL statement is processed, Oracle creates a memory area known as context area. A cursor is a pointer to this context area. It contains all information needed for processing the statement. In PL/SQL, the context area is controlled by Cursor. A cursor contains information on a select statement and the rows of data accessed by it.

A cursor is used to referred to a program to fetch and process the rows returned by the SQL statement, one at a time.

Commands:

```
-- Create the EMPLOYEE table
CREATE TABLE EMPLOYEE (
  empId NUMBER(4) PRIMARY KEY,
  name VARCHAR2(50) NOT NULL,
  dept VARCHAR2(50) NOT NULL,
  age NUMBER(3) NOT NULL,
  city VARCHAR2(50) NOT NULL,
  salary NUMBER(10) NOT NULL
);


INSERT INTO EMPLOYEE VALUES (1, 'Clark', 'Sales', 23, 'Noida', 10000);
INSERT INTO EMPLOYEE VALUES (2, 'Dave', 'Accounting', 33, 'Gurgaon', 20000);
INSERT INTO EMPLOYEE VALUES (3, 'Ava', 'Sales', 28, 'Delhi', 30000);
INSERT INTO EMPLOYEE VALUES (4, 'Emily', 'Marketing', 30, 'Bangalore', 40000);
INSERT INTO EMPLOYEE VALUES (5, 'William', 'IT', 27, 'Noida', 50000);

-- PL/SQL block to update salaries and count the employees updated
DECLARE
  total_rows NUMBER(2);
  CURSOR employee_cursor IS
    SELECT empId, salary
    FROM EMPLOYEE
    FOR UPDATE; -- Lock the rows for update
BEGIN
  total_rows := 0;

  FOR emp_record IN employee_cursor LOOP
    -- Update the salary for each employee
    UPDATE EMPLOYEE
    SET salary = emp_record.salary + 5000
    WHERE CURRENT OF employee_cursor;
```

```
    total_rows := total_rows + 1;
  END LOOP;

  IF total_rows = 0 THEN
    dbms_output.put_line('No employees updated');
  ELSE
    dbms_output.put_line(total_rows || ' employees updated');
  END IF;

  COMMIT; -- Commit the changes
END;
/
```

Output:

```
5 EMPLOYEE updated
PL/SQL procedure successfully completed
```

Conclusion: The cursor program is successfully created in PL/SQL.

Open Ended Experiment

Aim: To implement trigger in PL/SQL

Theory Required:

Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events –

- A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)
- A **database definition (DDL)** statement (CREATE, ALTER, or DROP).
- A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

Commands:

```
-- Create a PL/SQL block to create a table and a trigger

DECLARE

  -- Declare variables for table and trigger names

  table_name VARCHAR2(30) := 'customers';

  trigger_name VARCHAR2(30) := 'display_salary_changes';



BEGIN

  -- Create the 'customers' table

  EXECUTE IMMEDIATE 'CREATE TABLE ' || table_name || ' (

    ID NUMBER PRIMARY KEY,

    NAME VARCHAR2(50),

    SALARY NUMBER

  )';
```

```
  -- Create the trigger
  EXECUTE IMMEDIATE '
    CREATE OR REPLACE TRIGGER ' || trigger_name || '
    BEFORE DELETE OR INSERT OR UPDATE ON ' || table_name || '
    FOR EACH ROW
    WHEN (NEW.ID > 0)
    DECLARE
      sal_diff NUMBER;
    BEGIN
      -- Calculate salary difference
      sal_diff := :NEW.salary - NVL(:OLD.salary, 0);

      -- Print the information using dbms_output
      dbms_output.put_line("Old salary: " || NVL(:OLD.salary, 0));
      dbms_output.put_line("New salary: " || :NEW.salary);
      dbms_output.put_line("Salary difference: " || sal_diff);
    END;
  ';

  -- Commit the changes
  COMMIT;
END;
/
```

Output:

Trigger Created.

Conclusion: The trigger program is successfully created in PL/SQL.

DATABASE MANAGEMENT

CSE201

LAB -1: DDL AND DML COMMANDS IN SQL

Dated: 28/07/23

Aim: To create a table and insert, delete and update the elements into the table.

Commands:

```
CREATE TABLE students (
    student_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    age INT,
    major VARCHAR(100)
);
ALTER TABLE students
ADD COLUMN email VARCHAR(100)

ALTER TABLE students
RENAME COLUMN major TO department;

/*Comment: This is an example*/

INSERT INTO students (student_id, first_name, last_name, age, department, email)
VALUES
    (1, 'John', 'Doe', 20, 'Computer Science', 'john.doe@example.com'),
    (2, 'Jane', 'Smith', 22, 'Biology', 'jane.smith@example.com'),
    (3, 'Michael', 'Johnson', 21, 'Physics', 'michael.johnson@example.com');

UPDATE students
SET department = 'Chemistry'
WHERE student_id = 2;
```

```
DELETE FROM students
WHERE student_id = 3;
```

Output:

Students

student_id	first_name	last_name	age	department	email
1	John	Doe	20	Computer Science	john.doe@example.com
2	Jane	Smith	22	Chemistry	jane.smith@example.com

Conclusion: The table is created using various ddl and dml commands in sql.

DATABASE MANAGEMENT

CSE201

LAB -1 and 2: DDL AND DML COMMANDS IN SQL

Aim: To create a table and insert, delete and update the elements into the table.

Commands:

```
CREATE TABLE students (
    student_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    age INT,
    major VARCHAR(100)
);
ALTER TABLE students
ADD COLUMN email VARCHAR(100)

ALTER TABLE students
RENAME COLUMN major TO department;

/*Comment: This is an example*/

INSERT INTO students (student_id, first_name, last_name, age, department, email)
VALUES
    (1, 'John', 'Doe', 20, 'Computer Science', 'john.doe@example.com'),
    (2, 'Jane', 'Smith', 22, 'Biology', 'jane.smith@example.com'),
    (3, 'Michael', 'Johnson', 21, 'Physics', 'michael.johnson@example.com');

UPDATE students
SET department = 'Chemistry'
WHERE student_id = 2;
DELETE FROM students
WHERE student_id = 3;
```

Output:

Students

student_id	first_name	last_name	age	department	email
1	John	Doe	20	Computer Science	john.doe@example.com
2	Jane	Smith	22	Chemistry	jane.smith@example.com

Conclusion: The table is created using various ddl and dml commands in sql.

Practical-4
Lab Assignment
(Part 2)

Q. Create the following tables and answer the queries: (Take appropriate data types and relationships to define the columns and then insert relevant data).

1. SUPPLIER(SNO, SNAME, STATUS, CITY)
2. PARTS(PNO, PNAME, COLOR, WEIGHT, CITY)
3. PROJECT(JNO, JNAME, CITY)
4. SPJ(SNO, PNO, JNO, QTY)

Outputs of the Table

```
mysql> create table supplier(sno char(4), sname varchar(20), status varchar(20), city varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into supplier values('s1', 'Sanskriti', 'Active', 'Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s2', 'Karan', 'Inactive', 'Chennai');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s3', 'Atharva', 'Active', 'Bangalore');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s4', 'Samaira', 'Active', 'Mumbai');
Query OK, 1 row affected (0.00 sec)

mysql> select * from supplier;
+------+-----------+---------+-----------+
| sno  | sname     | sstatus | city      |
+------+-----------+---------+-----------+
| s1   | Sanskriti | Active  | Delhi     |
| s2   | Karan     | Inactive| Chennai   |
| s3   | Atharva   | Active  | Bangalore |
| s4   | Samaira   | Active  | Mumbai    |
+------+-----------+---------+-----------+
4 rows in set (0.00 sec)
```

```
mysql> create table parts(pno char(4), pname varchar(20), color varchar(20), weight numeric(20), city va
rchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into parts values('p1','part1','red',14.2,'Delhi');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p2','part2','blue',15.6,'Chennai');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p3','part3','green',5.3,'Bangalore');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p4','part4','yellow',44.2,'Chandigarh');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> select * from parts;
+------+-------+--------+--------+------------+
| pno  | pname | color  | weight | city       |
+------+-------+--------+--------+------------+
| p1   | part1 | red    |     14 | Delhi      |
| p2   | part2 | blue   |     16 | Chennai    |
| p3   | part3 | green  |      5 | Bangalore  |
| p4   | part4 | yellow |     44 | Chandigarh |
+------+-------+--------+--------+------------+
4 rows in set (0.00 sec)
```

```
mysql> create table project(jno char(4), jname varchar(20), city varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into project values('j1','project1','Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j2','project2','Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j3','project3','Bangalore');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j4','project4','Chandigarh');
Query OK, 1 row affected (0.00 sec)

mysql> select * from project;
+------+----------+------------+
| jno  | jname    | city       |
+------+----------+------------+
| j1   | project1 | Delhi      |
| j2   | project2 | Delhi      |
| j3   | project3 | Bangalore  |
| j4   | project4 | Chandigarh |
+------+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> create table spj(sno char(4), pno char(4), jno char(4), qty int);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into spj values('s1','p1','j2',2);
Query OK, 1 row affected (0.01 sec)

mysql> insert into spj values('s2','p1','j1',3);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s1','p3','j3',2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s3','p4','j4',4);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s2','p2','j4',1);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s4','p3','j3',3);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s3','p3','j1',4);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s4','p4','j2',5);
Query OK, 1 row affected (0.00 sec)

mysql> select * from spj;
+------+------+------+------+
| sno  | pno  | jno  | qty  |
+------+------+------+------+
| s1   | p1   | j2   |    2 |
| s2   | p1   | j1   |    3 |
| s1   | p3   | j3   |    2 |
| s3   | p4   | j4   |    4 |
| s2   | p2   | j4   |    1 |
| s4   | p3   | j3   |    3 |
| s3   | p3   | j1   |    4 |
| s4   | p4   | j2   |    5 |
+------+------+------+------+
8 rows in set (0.00 sec)
```

SQL Queries:

1. Get sno values for suppliers who supply project j1.

```
mysql> select sno from spj where jno='j1';
+------+
| sno  |
+------+
| s2   |
| s3   |
+------+
2 rows in set (0.00 sec)
```

2. Get sno values for suppliers who supply project j1 with part p1.

```
mysql> select sno from spj where jno='j1' and pno='p1';
+------+
| sno  |
+------+
| s2   |
+------+
1 row in set (0.00 sec)
```

3. Get jname values for projects supplied by supplier s1.

```
mysql> select jname from project, spj where sno='s1' and project.jno=spj.jno;
+----------+
| jname    |
+----------+
| project2 |
| project3 |
+----------+
2 rows in set (0.00 sec)
```

4. Get color values for parts supplied by supplier s1.

```
mysql> select color from parts, spj where sno='s1' and parts.pno=spj.pno;
+-------+
| color |
+-------+
| red   |
| green |
+-------+
2 rows in set (0.00 sec)
```

5. Get pno values for parts supplied to any project in London.

```
mysql> select parts.pno from parts, spj, project where project.city='Delhi' and parts.pno=spj.pno and pr
oject.jno=spj.jno;
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
+------+
4 rows in set (0.00 sec)
```

6. Get sno values for suppliers who supply project j1 with a red part.

```
mysql> select sno from spj, parts where parts.color='red' and jno='j2' and spj.pno=parts.pno;
+------+
| sno  |
+------+
| s1   |
+------+
1 row in set (0.00 sec)
```

7. Get sno values for suppliers who supply a London or Paris project with a red part.

```
mysql> select sno from spj, parts, project where parts.color='red' and (project.city='Delhi' or project.
city='Chennai') and parts.pno=spj.pno and project.jno=spj.jno;
+------+
| sno  |
+------+
| s2   |
| s1   |
+------+
2 rows in set (0.00 sec)
```

8. Get pno values for parts supplied to any project by a supplier in the same city.

```
mysql> select parts.pno from parts, spj, project where parts.city=project.city and parts.pno=spj.pno and
 project.jno=spj.jno;
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
| p3   |
+------+
5 rows in set (0.00 sec)
```

9. Get pno values for parts supplied to any project in London by a supplier in London.

```
mysql> select parts.pno from parts, spj, project where parts.city='Chandigarh' and project.city='Chandig
arh' and parts.pno=spj.pno and project.jno=spj.jno;
+------+
| pno  |
+------+
| p4   |
+------+
1 row in set (0.00 sec)
```

10. Get jno values for projects supplied by at least one supplier not in the same city.

```
mysql> select distinct(project.jno) from project, supplier, spj where supplier.city!=project.city and su
pplier.sno=spj.sno and spj.jno=project.jno;
+------+
| jno  |
+------+
| j3   |
| j4   |
| j1   |
| j2   |
+------+
4 rows in set (0.00 sec)
```

11. Get all pairs of city values such that a supplier in the first city supplies a project in the second city.

```
mysql> select distinct supplier.city, project.city from supplier, project, spj where spj.sno=supplier.sn
o and spj.jno=project.jno and project.city!=supplier.city;
+-----------+------------+
| city      | city       |
+-----------+------------+
| Bangalore | Delhi      |
| Chennai   | Delhi      |
| Mumbai    | Delhi      |
| Mumbai    | Bangalore  |
| Delhi     | Bangalore  |
| Chennai   | Chandigarh |
| Bangalore | Chandigarh |
+-----------+------------+
7 rows in set (0.00 sec)
```

12. Get sno values for suppliers who supply the same part to all projects.

```
mysql> SELECT SNO FROM SPJ
    -> GROUP BY SNO, PNO
    -> HAVING COUNT(DISTINCT JNO) = (SELECT COUNT(*) FROM PROJECT);
Empty set (0.01 sec)
```

13. Get pno values for parts supplied to all projects in Delhi.

```
mysql> select parts.pno from parts, project, spj where parts.pno=spj.pno and spj.jno=project.jno and pro
ject.city='Delhi';
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
+------+
4 rows in set (0.00 sec)
```

14. Get sname values for suppliers who supplies at least one red part to any project.

```
mysql> select distinct(supplier.sname) from supplier, parts, spj where supplier.sno=spj.sno and parts.pn
o=spj.pno and parts.color='red';
+-----------+
| sname     |
+-----------+
| Sanskriti |
| Karan     |
+-----------+
2 rows in set (0.00 sec)
```

15. Get total quantity of part p1 supplied by supplier s1.

```
mysql> select count(pno) as total_quantity from spj where sno='s1' and pno='p1';
+----------------+
| total_quantity |
+----------------+
|              1 |
+----------------+
1 row in set (0.00 sec)
```

16. Get the total number of projects supplied by supplier s3.

```
mysql> select count(jno) as total_projects from spj where sno='s3';
+----------------+
| total_projects |
+----------------+
|              2 |
+----------------+
1 row in set (0.00 sec)
```

17. Change color of all red parts to orange.

```
mysql> update parts set color='orange' where color='red'; select * from parts;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+------+-------+--------+--------+------------+
| pno  | pname | color  | weight | city       |
+------+-------+--------+--------+------------+
| p1   | part1 | orange |     14 | Delhi      |
| p2   | part2 | blue   |     16 | Chennai    |
| p3   | part3 | green  |      5 | Bangalore  |
| p4   | part4 | yellow |     44 | Chandigarh |
+------+-------+--------+--------+------------+
4 rows in set (0.00 sec)
```

18. Get sname values for suppliers who supply to both projects j1 and j2.

```
mysql> select distinct(sname) from supplier, spj where supplier.sno=spj.sno and (spj.jno='j1' or spj.jno
='j2');
+-----------+
| sname     |
+-----------+
| Sanskriti |
| Karan     |
| Atharva   |
| Samaira   |
+-----------+
4 rows in set (0.00 sec)
```

19. Get all city, pno, city triples such that a supplier in the first city supplies the specified part to a project in the second city.

```
mysql> SELECT DISTINCT S.CITY AS SupplierCity, SPJ.PNO, PJ.CITY AS ProjectCity
    -> FROM SUPPLIER S
    -> INNER JOIN SPJ ON S.SNO = SPJ.SNO
    -> INNER JOIN PARTS P ON SPJ.PNO = P.PNO
    -> INNER JOIN PROJECT PJ ON SPJ.JNO = PJ.JNO
    -> WHERE S.CITY <> PJ.CITY AND P.PNO = 'p1';
+--------------+------+-------------+
| SupplierCity | PNO  | ProjectCity |
+--------------+------+-------------+
| Chennai      | p1   | Delhi       |
+--------------+------+-------------+
```

20. Get jnames for those project which are supplied by supplier XYZ.

```
mysql> select distinct(jname) from project, spj where project.jno=spj.jno and spj.sno='s2';
+----------+
| jname    |
+----------+
| project1 |
| project4 |
+----------+
2 rows in set (0.00 sec)
```

Practical -6

Lab Assignment- Ques 1

Create the following tables

Course(course_no char(4), course_name varchar(20))
Course_fee(course_no char(4), full_part char(1) (F/P), fees number(10))
course_no and full_part should be unique
Student(prospectus_no number(10), name varchar(20), address varchar(30), phone_no number(11), D_O_B date, total_amt number(10,2), amt_paid number(10,2), installment char(1) (I/F))
Installment(prospectus_no number(10) (foreign key) on delete cascade, installment_amt number(10,2), due_dt date, paid char(1) (P,U))
prospectus_no and due_dt should be unique
Course_taken(prospectus_no number(10) (foreign key), course_no char(4), start_dt date, full_part char(1) (F/P), time_slot char(2), performance varchar(20))

Outputs of the Table:

```
mysql> create table Course(course_no char(4) primary key, course_name varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into course values('c1','DBMS');
Query OK, 1 row affected (0.01 sec)

mysql> insert into course values('c2','ADA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c3','JAVA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c4','C++');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c5','EN');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c6','C');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c7','R');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c8','BEE');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c9','DECO');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c10','MATH');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from course;
+-----------+--------------+
| course_no | course_name  |
+-----------+--------------+
| c1        | DBMS         |
| c10       | MATH         |
| c2        | ADA          |
| c3        | JAVA         |
| c4        | C++          |
| c5        | EN           |
| c6        | C            |
| c7        | R            |
| c8        | BEE          |
| c9        | DECO         |
+-----------+--------------+
10 rows in set (0.00 sec)
```

```
mysql> create table course_fee(course_no char(4) primary key, full_part char(1), fees numeric
(10), foreign key(course_no) references course(course_no), check(full_part='F' or full_part='
P'));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into course_fee values('c1','F',10000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c2','P',40000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c3','F',6000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c4','P',12000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c5','P',10000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c6','F',23000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c7','F',98600);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c8','F',87000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c9','P',695000);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into course_fee values('c10','F',3000);
Query OK, 1 row affected (0.00 sec)

mysql> select * from course_fee;
+-----------+-----------+--------+
| course_no | full_part | fees   |
+-----------+-----------+--------+
| c1        | F         |  10000 |
| c10       | F         |   3000 |
| c2        | P         |  40000 |
| c3        | F         |   6000 |
| c4        | P         |  12000 |
| c5        | P         |  10000 |
| c6        | F         |  23000 |
| c7        | F         |  98600 |
| c8        | F         |  87000 |
| c9        | P         | 695000 |
+-----------+-----------+--------+
10 rows in set (0.00 sec)
```

```
mysql> create table student(prospectus_no numeric(11) primary key, st_name varchar(20
), address varchar(30), phone_no numeric(11), D_O_B date, total_amt numeric(10,2), am
t_paid numeric(10,2), installment char(1), check(installment='I' or installment='F'))
;
Query OK, 0 rows affected (0.01 sec)

mysql> insert into student values(32456,'Sanskriti','Delhi',9876899910,date'2003-02-27',20000,10000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(42849,'Saumya','Delhi',9698270918,date'1995-11-29',20000,20000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(32345,'Parth','Mumbai',7298190145,date'2000-02-18',40000,6000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(42345,'Arnav','Bangalore',9278190187,date'1999-05-10',120000,12000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(52345,'Samiksha','Mumbai',7187298605,date'2000-01-15',120000,12000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(89172,'Hardik','Chennai',8920186528,date'1995-08-07',679893,60000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(76289,'Lakshay','Jaipur',8729017345,date'2002-04-09',34000,30000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(23487,'Rohan','Mumbai',9817829076,date'2001-11-11',78653,8000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(91829,'Harshita','Chennai',9879276422,date'2002-03-14',65000,7500,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(28397,'Manvi','Bangalore',9810255632,date'2001-08-22',20000,20000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(65784,'Shaurya','Delhi',9917289015,date'2000-09-21',10620,2300,'F');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from student;
+--------------+----------+-----------+------------+------------+-----------+----------+-------------+
| prospectus_no | st_name   | address   | phone_no   | D_O_B      | total_amt | amt_paid | installment |
+--------------+----------+-----------+------------+------------+-----------+----------+-------------+
|        23487 | Rohan     | Mumbai    | 9817829076 | 2001-11-11 |  78653.00 |  8000.00 | F           |
|        28397 | Manvi     | Bangalore | 9810255632 | 2001-08-22 |  20000.00 | 20000.00 | I           |
|        32345 | Parth     | Mumbai    | 7298190145 | 2000-02-18 |  40000.00 |  6000.00 | F           |
|        32456 | Sanskriti | Delhi     | 9876899910 | 2003-02-27 |  20000.00 | 10000.00 | I           |
|        42345 | Arnav     | Bangalore | 9278190187 | 1999-05-10 | 120000.00 | 12000.00 | F           |
|        42849 | Saumya    | Delhi     | 9698270918 | 1995-11-29 |  20000.00 | 20000.00 | I           |
|        52345 | Samiksha  | Mumbai    | 7187298605 | 2000-01-15 | 120000.00 | 12000.00 | F           |
|        65784 | Shaurya   | Delhi     | 9917289015 | 2000-09-21 |  10620.00 |  2300.00 | F           |
|        76289 | Lakshay   | Jaipur    | 8729017345 | 2002-04-09 |  34000.00 | 30000.00 | I           |
|        89172 | Hardik    | Chennai   | 8920186528 | 1995-08-07 | 679893.00 | 60000.00 | F           |
|        91829 | Harshita  | Chennai   | 9879276422 | 2002-03-14 |  65000.00 |  7500.00 | I           |
+--------------+----------+-----------+------------+------------+-----------+----------+-------------+
11 rows in set (0.00 sec)
```

```
mysql> create table installment(prospectus_no numeric(10) unique, installment_amt num
eric(10,2), due_dt date unique, paid char(1), foreign key(prospectus_no) references s
tudent(prospectus_no) on delete cascade, check(paid='P' or paid='U'));
Query OK, 0 rows affected (0.02 sec)

mysql> insert into installment values(32456, 50000 ,date'2023-01-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(42849, 20000 ,date'2023-02-01','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(32345, 150000 ,date'2023-03-01','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(42345, 120000 ,date'2023-04-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(52345, 60000 ,date'2023-05-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(89172, 60000 ,date'2023-04-27','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(76289, 98000 ,date'2022-12-24','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(23487, 56800 ,date'2022-11-14','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(91829, 78900 ,date'2023-09-07','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(28397, 897200 ,date'2023-07-25','U');
Query OK, 1 row affected (0.00 sec)

mysql> ^C
mysql> insert into installment values(65784, 45790 ,date'2022-12-29','U');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from installment;
+---------------+-----------------+------------+------+
| prospectus_no | installment_amt | due_dt     | paid |
+---------------+-----------------+------------+------+
|         32456 |        50000.00 | 2023-01-01 | U    |
|         42849 |        20000.00 | 2023-02-01 | P    |
|         32345 |       150000.00 | 2023-03-01 | P    |
|         42345 |       120000.00 | 2023-04-01 | U    |
|         52345 |        60000.00 | 2023-05-01 | U    |
|         89172 |        60000.00 | 2023-04-27 | U    |
|         76289 |        98000.00 | 2022-12-24 | U    |
|         23487 |        56800.00 | 2022-11-14 | P    |
|         91829 |        78900.00 | 2023-09-07 | P    |
|         28397 |       897200.00 | 2023-07-25 | U    |
|         65784 |        45790.00 | 2022-12-29 | U    |
+---------------+-----------------+------------+------+
11 rows in set (0.00 sec)
```

```
mysql> create table Course_taken(prospectus_no numeric(10), course_no char(4), start_dt date,
 full_part char(1), time_slot char(2), performance varchar(20), foreign key(prospectus_no) RE
FERENCES STUDENT(prospectus_no), foreign key(course_no) references course(course_no), check(f
ull_part='F' or full_part='P'));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into Course_taken values(32456, 'c1', date'2021-09-29', 'F', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> ^C
mysql> insert into Course_taken values(42849, 'c2', date'2021-05-23', 'F', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(32345, 'c3', date'2021-08-19', 'F', 'FN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(42345, 'c4', date'2021-03-23', 'P', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(89172, 'c1', date'2021-07-14', 'F', 'AN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(76289, 'c3', date'2021-05-23', 'P', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(23487, 'c2', date'2021-04-16', 'P', 'AN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(91829, 'c5', date'2021-07-13', 'F', 'FN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(28397, 'c9', date'2021-11-17', 'F', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(65784, 'c10', date'2021-12-12', 'P', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from Course_taken;
+--------------+-----------+------------+-----------+-----------+-------------+
| prospectus_no | course_no | start_dt   | full_part | time_slot | performance |
+--------------+-----------+------------+-----------+-----------+-------------+
|         32456 | c1        | 2021-09-29 | F         | AN        | GOOD        |
|         42849 | c2        | 2021-05-23 | F         | FN        | GOOD        |
|         32345 | c3        | 2021-08-19 | F         | FN        | BAD         |
|         42345 | c4        | 2021-03-23 | P         | AN        | GOOD        |
|         89172 | c1        | 2021-07-14 | F         | AN        | BAD         |
|         76289 | c3        | 2021-05-23 | P         | FN        | GOOD        |
|         23487 | c2        | 2021-04-16 | P         | AN        | BAD         |
|         91829 | c5        | 2021-07-13 | F         | FN        | BAD         |
|         28397 | c9        | 2021-11-17 | F         | FN        | GOOD        |
|         65784 | c10       | 2021-12-12 | P         | AN        | GOOD        |
+--------------+-----------+------------+-----------+-----------+-------------+
10 rows in set (0.00 sec)
```

SQL Queries:

1. Retrieve name and course no of all the students.

```
mysql> select student.st_name, Course_taken.course_no from student, Course_taken where studen
t.prospectus_no=Course_taken.prospectus_no;
+-----------+-----------+
| st_name   | course_no |
+-----------+-----------+
| Sanskriti | c1        |
| Saumya    | c2        |
| Parth     | c3        |
| Arnav     | c4        |
| Hardik    | c1        |
| Lakshay   | c3        |
| Rohan     | c2        |
| Harshita  | c5        |
| Manvi     | c9        |
| Shaurya   | c10       |
+-----------+-----------+
10 rows in set (0.01 sec)
```

2. List the names of students who have paid the full amount at the time of admission.

```
mysql> select st_name from student where total_amt=amt_paid;
+---------+
| st_name |
+---------+
| Manvi   |
| Saumya  |
+---------+
2 rows in set (0.00 sec)
```

3. Find the names of students starting with A.

```
mysql> select * from student where st_name like 'A%';
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
| prospectus_no | st_name | address   | phone_no   | D_O_B      | total_amt | amt_paid | inst
allment |
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
|         42345 | Arnav   | Bangalore | 9278190187 | 1999-05-10 | 120000.00 | 12000.00 | F
        |
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
1 row in set (0.00 sec)
```

4. Print the names of students whose total amount is not equal to amount due.

```
mysql> select student.st_name from student where amt_paid not in(select amt_paid from student
 where total_amt=amt_paid);
+-----------+
| st_name   |
+-----------+
| Rohan     |
| Parth     |
| Sanskriti |
| Arnav     |
| Samiksha  |
| Shaurya   |
| Lakshay   |
| Hardik    |
| Harshita  |
+-----------+
9 rows in set (0.00 sec)
```

5. Count the number of students who have joined in current year, current month.

```
mysql> select COUNT(*) from Course_taken where start_dt like '2023-05-__';
+----------+
| COUNT(*) |
+----------+
|        0 |
+----------+
1 row in set (0.00 sec)
```

6. Determine the maximum and minimum course fees.

```
mysql> select max(fees), min(fees) from course_fee;
+-----------+-----------+
| max(fees) | min(fees) |
+-----------+-----------+
|    695000 |      3000 |
+-----------+-----------+
1 row in set (0.00 sec)
```

7. Increase the fee of oracle by 50%.

```
mysql> update course_fee set fees=1.5*fees where course_no in(select course_no from Course wh
ere Course.course_name='DBMS'); select * from course_fee;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+-----------+-----------+--------+
| course_no | full_part | fees   |
+-----------+-----------+--------+
| c1        | F         |  15000 |
| c10       | F         |   3000 |
| c2        | P         |  40000 |
| c3        | F         |   6000 |
| c4        | P         |  12000 |
| c5        | P         |  10000 |
| c6        | F         |  23000 |
| c7        | F         |  98600 |
| c8        | F         |  87000 |
| c9        | P         | 695000 |
+-----------+-----------+--------+
10 rows in set (0.00 sec)
```

8. Print the details of courses whose fees are between 5000 and 10000.

```
mysql> select Course.course_no, Course.course_name, course_fee.fees, course_fee.full_part fro
m Course, course_fee where Course.course_no=course_fee.course_no and(course_fee.fees>5000 and
 course_fee.fees<10000);
+-----------+-------------+------+-----------+
| course_no | course_name | fees | full_part |
+-----------+-------------+------+-----------+
| c3        | JAVA        | 6000 | F         |
+-----------+-------------+------+-----------+
1 row in set (0.00 sec)
```

9. Display the admission date in Date, Month, Year format.

```
mysql> select start_dt as start_date from Course_taken;
+------------+
| start_date |
+------------+
| 2021-09-29 |
| 2021-05-23 |
| 2021-08-19 |
| 2021-03-23 |
| 2021-07-14 |
| 2021-05-23 |
| 2021-04-16 |
| 2021-07-13 |
| 2021-11-17 |
| 2021-12-12 |
+------------+
10 rows in set (0.00 sec)
```

10. Find out in which course maximum number of students have taken admission.

```
mysql> select Course.course_name from Course,Course_taken where Course.course_no=Course_taken
.course_no group by Course.course_name having count(Course_taken.prospectus_no)>=all (select
count(prospectus_no) from Course_taken group by course_no);
+-------------+
| course_name |
+-------------+
| DBMS        |
| ADA         |
| JAVA        |
+-------------+
3 rows in set (0.00 sec)
```

11. Change the course_name from C to C Programming.

```
mysql> update Course set course_name='CProgramming' WHERE course_name='C';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from course;
+-----------+--------------+
| course_no | course_name  |
+-----------+--------------+
| c1        | DBMS         |
| c10       | MATH         |
| c2        | ADA          |
| c3        | JAVA         |
| c4        | C++          |
| c5        | EN           |
| c6        | CProgramming |
| c7        | R            |
| c8        | BEE          |
| c9        | DECO         |
+-----------+--------------+
10 rows in set (0.00 sec)
```

12. Display the admission date in DD-MONTH-YYYY format.

```
mysql> select date_format(start_dt, '%d-%m-%y') from Course_taken;
+-----------------------------------+
| date_format(start_dt, '%d-%m-%y') |
+-----------------------------------+
| 29-09-21                          |
| 23-05-21                          |
| 19-08-21                          |
| 23-03-21                          |
| 14-07-21                          |
| 23-05-21                          |
| 16-04-21                          |
| 13-07-21                          |
| 17-11-21                          |
| 12-12-21                          |
+-----------------------------------+
10 rows in set (0.00 sec)
```

13. Get the sum of amount to be collected from students in this month.

```
mysql> select sum(student.total_amt-student.amt_paid) from student,installment where student.
prospectus_no=installment.prospectus_no and installment.due_dt like'__-APR-23';
```

```
727893
```

14. Find out in which course the maximum number of students have taken admission in the current month.

```
mysql> select Course.course_name from Course, Course_taken where Course.course_no=Course_take
n.course_no and start_dt like '__-MAY-21' group by Course.course_name having count(Course_tak
en.prospectus_no)>=all(select count(prospectus_no) from Course_taken group by course_no);
```

15. Select the students who have not yet paid full amount of fees.

```
mysql> select st_name from student where total_amt>amt_paid;
+-----------+
| st_name   |
+-----------+
| Rohan     |
| Parth     |
| Sanskriti |
| Arnav     |
| Samiksha  |
| Shaurya   |
| Lakshay   |
| Hardik    |
| Harshita  |
+-----------+
9 rows in set (0.00 sec)
```

Practical-4
Lab Assignment
(Part 3)

Q. Create the required Tables

Department(Dept_No, Dept_Name)

Employee(E_ID, E_Name, Salary, Hiredate, LOC, Mgr_Eid, Job, Grade, Dept_No)

Output of the Table

```
mysql> create table department(dept_no varchar(5) primary key, dept_name varchar(5));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into department values('D1', 'CSE');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D2', 'IT');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D3', 'AI');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D4', 'DS');
Query OK, 1 row affected (0.00 sec)

mysql> select * from department;
+---------+-----------+
| dept_no | dept_name |
+---------+-----------+
| D1      | CSE       |
| D2      | IT        |
| D3      | AI        |
| D4      | DS        |
+---------+-----------+
4 rows in set (0.00 sec)
```

```
mysql> create table employees (eid varchar(5) primary key, ename varchar(25), salary int, hiredate date,
 loc varchar(25), mgr_eid varchar(25), job varchar(25), grade varchar(5), dept_no varchar(5), foreign ke
y(dept_no) references department(dept_no) on delete set null);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into employees values('e1','Sanskriti', 50000, date'2022-06-01','loc1','mgr1','Software En
gineer','A','D1');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e2','Karan', 40000, date'2022-05-01','loc2','mgr2','Hardware Engine
er','A','D2');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e3','Saumya', 60000, date'2022-02-01','loc3','mgr3','Website Design
er','A','D2');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e4','Atharva', 30000, date'2022-06-10','loc4','mgr4','ML Engineer',
'A','D3');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from employees;
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job               | grade | dept_no |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| e1  | Sanskriti |  50000 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
4 rows in set (0.00 sec)
```

SQL Queries

1. Display each employee name and hiredate of systems department.

```
mysql> select ename, hiredate from employees;
+-----------+------------+
| ename     | hiredate   |
+-----------+------------+
| Sanskriti | 2022-06-01 |
| Karan     | 2022-05-01 |
| Saumya    | 2022-02-01 |
| Atharva   | 2022-06-10 |
+-----------+------------+
4 rows in set (0.00 sec)
```

2.Write query to calculate length of service of each employee.

```
mysql> select date_format(curdate(), '%Y') - date_format(hiredate, '%Y') as years, date_format(curdate()
, '%m') - date_format(hiredate, '%m') as months, date_format(curdate(), '%d') - date_format(hiredate, '%
d') as days from employees;
+-------+--------+------+
| years | months | days |
+-------+--------+------+
|     1 |      2 |   11 |
|     1 |      3 |   11 |
|     1 |      6 |   11 |
|     1 |      2 |    2 |
+-------+--------+------+
4 rows in set (0.00 sec)
```

3. Find the second maximum salary of all employees.

```
mysql> select salary from employees order by salary desc limit 1,1;
+--------+
| salary |
+--------+
|  50000 |
+--------+
1 row in set (0.00 sec)
```

4.Display all employee name and department name in department name order.

```
mysql> select ename, dept_name from employees, department where department.dept_no=employees.dept_no ord
er by dept_name;
+-----------+-----------+
| ename     | dept_name |
+-----------+-----------+
| Atharva   | AI        |
| Sanskriti | CSE       |
| Karan     | IT        |
| Saumya    | IT        |
+-----------+-----------+
4 rows in set (0.00 sec)
```

5.Find the name of lowest paid employee for each manager.

```
mysql> select mgr_eid, min(salary) from employees where mgr_eid is not null group by mgr_eid order by m
n(salary) desc;
+---------+-------------+
| mgr_eid | min(salary) |
+---------+-------------+
| mgr3    |       60000 |
| mgr1    |       50000 |
| mgr2    |       40000 |
| mgr4    |       30000 |
+---------+-------------+
4 rows in set (0.00 sec)
```

6.Display the department that has no employee.

```
mysql> select * from department where dept_no not in(select dept_no from employees);
+---------+-----------+
| dept_no | dept_name |
+---------+-----------+
| D4      | DS        |
+---------+-----------+
1 row in set (0.00 sec)
```

7.Find the employees who earn the maximum salary in each job type. Sort in descending order of salary.

```
mysql> SELECT e.job, e.ename, e.salary
    -> FROM employees e
    -> JOIN (
    ->     SELECT job, MAX(salary) as max_salary
    ->     FROM employees
    ->     GROUP BY job
    -> ) max_salaries
    -> ON e.job = max_salaries.job AND e.salary = max_salaries.max_salary
    -> ORDER BY e.salary DESC;
+-------------------+-----------+--------+
| job               | ename     | salary |
+-------------------+-----------+--------+
| Website Designer  | Saumya    |  60000 |
| Software Engineer | Sanskriti |  50000 |
| Hardware Engineer | Karan     |  40000 |
| ML Engineer       | Atharva   |  30000 |
+-------------------+-----------+--------+
4 rows in set (0.00 sec)
```

8.In which year did most people joined the company? Display the year and number of employees.

```
mysql> SELECT YEAR(Hiredate) AS JoinYear, COUNT(*) AS NumEmployees
    -> FROM Employees
    -> GROUP BY JoinYear
    -> ORDER BY NumEmployees DESC
    -> LIMIT 1;
+----------+--------------+
| JoinYear | NumEmployees |
+----------+--------------+
|     2022 |            4 |
+----------+--------------+
1 row in set (0.00 sec)
```

9.Display the details of those employees who earn greater than average of their department.

```
mysql> select dept_no, job as dept_name, ename, salary from employees e where salary > (select avg(Salar
y) from employees where employees.dept_no = dept_no) order by dept_no;
+---------+------------------+-----------+--------+
| dept_no | dept_name        | ename     | salary |
+---------+------------------+-----------+--------+
| D1      | Software Engineer | Sanskriti |  50000 |
| D2      | Website Designer  | Saumya    |  60000 |
+---------+------------------+-----------+--------+
2 rows in set (0.00 sec)
```

10.List the employees having salary between 10000 and 20000

```
mysql> select * from employees where salary between 40000 and 60000;
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job              | grade | dept_no |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| e1  | Sanskriti |  50000 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
3 rows in set (0.00 sec)
```

11.Display all employees hired during 1983. those employees who earn greater than average of their department.

```
mysql> SELECT eid, ename, salary, hiredate, LOC, Mgr_Eid, Job, Grade, Dept_No
    -> FROM Employees
    -> WHERE YEAR(Hiredate) = 1983
    -> AND Salary > (
    ->     SELECT AVG(Salary)
    ->     FROM Employees AS subquery
    ->     WHERE subquery.Dept_No = Employees.Dept_No
    -> );
Empty set (0.00 sec)
```

12.Update the salaries of all employees in marketing department & hike it by 15%.

```
mysql> update employees set salary=salary+(0.15*salary) where job='Software Engineer'; select * from emp
loyees where job='Software Engineer';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+-----+----------+--------+------------+------+---------+------------------+-------+---------+
| eid | ename    | salary | hiredate   | loc  | mgr_eid | job              | grade | dept_no |
+-----+----------+--------+------------+------+---------+------------------+-------+---------+
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
+-----+----------+--------+------------+------+---------+------------------+-------+---------+
1 row in set (0.00 sec)
```

13. Get the gross salaries of all the employees.

```
mysql> select 2.5*salary from employees as gross_salary;
+-------------+
| 2.5*salary |
+-------------+
|   143750.0 |
|   100000.0 |
|   150000.0 |
|    75000.0 |
+-------------+
4 rows in set (0.00 sec)
```

14. Get the names of employees and their manager's name.

```
mysql> SELECT E.ename AS EmployeeName, M.ename AS ManagerName
    -> FROM Employees E
    -> LEFT JOIN Employees M ON E.Mgr_Eid = M.eid;
+--------------+-------------+
| EmployeeName | ManagerName |
+--------------+-------------+
| Sanskriti    | NULL        |
| Karan        | NULL        |
| Saumya       | NULL        |
| Atharva      | NULL        |
+--------------+-------------+
4 rows in set (0.00 sec)
```

15.Display the name, location and department name of all the employees earning more than 1500.

```
mysql> select ename, loc, dept_name from department, employees where department.dept_no=employees.dept_n
o and salary>1500;
+-----------+------+-----------+
| ename     | loc  | dept_name |
+-----------+------+-----------+
| Sanskriti | loc1 | CSE       |
| Karan     | loc2 | IT        |
| Saumya    | loc3 | IT        |
| Atharva   | loc4 | AI        |
+-----------+------+-----------+
4 rows in set (0.00 sec)
```

16. Show all the employees in LOC1.

```
mysql> select ename from employees where loc='loc1';
+------------+
| ename      |
+------------+
| Sanskriti  |
+------------+
1 row in set (0.00 sec)
```

17. List the employees name, job, salary, grade, and department for employees in the company except clerks. Sort on employee names.

```
mysql> select ename, job, salary, grade, dept_name from department, employees where department.dept_no=e
mployees.dept_no and job!='Clerk' order by ename;
+-----------+-------------------+--------+-------+-----------+
| ename     | job               | salary | grade | dept_name |
+-----------+-------------------+--------+-------+-----------+
| Atharva   | ML Engineer       |  30000 | A     | AI        |
| Karan     | Hardware Engineer |  40000 | A     | IT        |
| Sanskriti | Software Engineer |  57500 | A     | CSE       |
| Saumya    | Website Designer  |  60000 | A     | IT        |
+-----------+-------------------+--------+-------+-----------+
4 rows in set (0.00 sec)
```

18. Find the employees who earns the minimum salary for their job. Sort in descending order of salary.

```
mysql> select * from employees where salary in (select min(salary) from employees group by job) order by
 salary desc;
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job               | grade | dept_no |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
4 rows in set (0.00 sec)
```

19.Find the most recently hired employees in the department order by hiredate.

```
mysql> select * from employees e where hiredate in (select max(hiredate) from employees where e.dept_no
= dept_no) order by hiredate desc;
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job               | grade | dept_no |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
3 rows in set (0.00 sec)
```

20.Find out the difference between highest and lowest salaries.

```
mysql> select max(salary) - min(salary) difference from employees;
+------------+
| difference |
+------------+
|      30000 |
+------------+
1 row in set (0.00 sec)
```

Practical-7
Lab Assignment
(Part 2)

Q. Create the following tables and answer the queries: (Take appropriate data types and relationships to define the columns and then insert relevant data).

1. SUPPLIER(SNO, SNAME, STATUS, CITY)
2. PARTS(PNO, PNAME, COLOR, WEIGHT, CITY)
3. PROJECT(JNO, JNAME, CITY)
4. SPJ(SNO, PNO, JNO, QTY)

Outputs of the Table

```
mysql> create table supplier(sno char(4), sname varchar(20), status varchar(20), city varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into supplier values('s1', 'Sanskriti', 'Active', 'Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s2', 'Karan', 'Inactive', 'Chennai');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s3', 'Atharva', 'Active', 'Bangalore');
Query OK, 1 row affected (0.00 sec)

mysql> insert into supplier values('s4', 'Samaira', 'Active', 'Mumbai');
Query OK, 1 row affected (0.00 sec)

mysql> select * from supplier;
+------+----------+---------+----------+
| sno  | sname    | sstatus | city     |
+------+----------+---------+----------+
| s1   | Sanskriti | Active  | Delhi    |
| s2   | Karan    | Inactive | Chennai  |
| s3   | Atharva  | Active  | Bangalore |
| s4   | Samaira  | Active  | Mumbai   |
+------+----------+---------+----------+
4 rows in set (0.00 sec)
```

```
mysql> create table parts(pno char(4), pname varchar(20), color varchar(20), weight numeric(20), city va
rchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into parts values('p1','part1','red',14.2,'Delhi');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p2','part2','blue',15.6,'Chennai');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p3','part3','green',5.3,'Bangalore');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> insert into parts values('p4','part4','yellow',44.2,'Chandigarh');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> select * from parts;
+------+-------+--------+--------+------------+
| pno  | pname | color  | weight | city       |
+------+-------+--------+--------+------------+
| p1   | part1 | red    |     14 | Delhi      |
| p2   | part2 | blue   |     16 | Chennai    |
| p3   | part3 | green  |      5 | Bangalore  |
| p4   | part4 | yellow |     44 | Chandigarh |
+------+-------+--------+--------+------------+
4 rows in set (0.00 sec)
```

```
mysql> create table project(jno char(4), jname varchar(20), city varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into project values('j1','project1','Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j2','project2','Delhi');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j3','project3','Bangalore');
Query OK, 1 row affected (0.00 sec)

mysql> insert into project values('j4','project4','Chandigarh');
Query OK, 1 row affected (0.00 sec)

mysql> select * from project;
+------+----------+------------+
| jno  | jname    | city       |
+------+----------+------------+
| j1   | project1 | Delhi      |
| j2   | project2 | Delhi      |
| j3   | project3 | Bangalore  |
| j4   | project4 | Chandigarh |
+------+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> create table spj(sno char(4), pno char(4), jno char(4), qty int);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into spj values('s1','p1','j2',2);
Query OK, 1 row affected (0.01 sec)

mysql> insert into spj values('s2','p1','j1',3);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s1','p3','j3',2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s3','p4','j4',4);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s2','p2','j4',1);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s4','p3','j3',3);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s3','p3','j1',4);
Query OK, 1 row affected (0.00 sec)

mysql> insert into spj values('s4','p4','j2',5);
Query OK, 1 row affected (0.00 sec)

mysql> select * from spj;
+------+------+------+------+
| sno  | pno  | jno  | qty  |
+------+------+------+------+
| s1   | p1   | j2   |    2 |
| s2   | p1   | j1   |    3 |
| s1   | p3   | j3   |    2 |
| s3   | p4   | j4   |    4 |
| s2   | p2   | j4   |    1 |
| s4   | p3   | j3   |    3 |
| s3   | p3   | j1   |    4 |
| s4   | p4   | j2   |    5 |
+------+------+------+------+
8 rows in set (0.00 sec)
```

SQL Queries:

1. Get sno values for suppliers who supply project j1.

```
mysql> select sno from spj where jno='j1';
+------+
| sno  |
+------+
| s2   |
| s3   |
+------+
2 rows in set (0.00 sec)
```

2. Get sno values for suppliers who supply project j1 with part p1.

```
mysql> select sno from spj where jno='j1' and pno='p1';
+------+
| sno  |
+------+
| s2   |
+------+
1 row in set (0.00 sec)
```

3. Get jname values for projects supplied by supplier s1.

```
mysql> select jname from project, spj where sno='s1' and project.jno=spj.jno;
+----------+
| jname    |
+----------+
| project2 |
| project3 |
+----------+
2 rows in set (0.00 sec)
```

4. Get color values for parts supplied by supplier s1.

```
mysql> select color from parts, spj where sno='s1' and parts.pno=spj.pno;
+-------+
| color |
+-------+
| red   |
| green |
+-------+
2 rows in set (0.00 sec)
```

5. Get pno values for parts supplied to any project in London.

```
mysql> select parts.pno from parts, spj, project where project.city='Delhi' and parts.pno=spj.pno and pr
oject.jno=spj.jno;
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
+------+
4 rows in set (0.00 sec)
```

6. Get sno values for suppliers who supply project j1 with a red part.

```
mysql> select sno from spj, parts where parts.color='red' and jno='j2' and spj.pno=parts.pno;
+------+
| sno  |
+------+
| s1   |
+------+
1 row in set (0.00 sec)
```

7. Get sno values for suppliers who supply a London or Paris project with a red part.

```
mysql> select sno from spj, parts, project where parts.color='red' and (project.city='Delhi' or project.
city='Chennai') and parts.pno=spj.pno and project.jno=spj.jno;
+------+
| sno  |
+------+
| s2   |
| s1   |
+------+
2 rows in set (0.00 sec)
```

8. Get pno values for parts supplied to any project by a supplier in the same city.

```
mysql> select parts.pno from parts, spj, project where parts.city=project.city and parts.pno=spj.pno and
 project.jno=spj.jno;
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
| p3   |
+------+
5 rows in set (0.00 sec)
```

9. Get pno values for parts supplied to any project in London by a supplier in London.

```
mysql> select parts.pno from parts, spj, project where parts.city='Chandigarh' and project.city='Chandig
arh' and parts.pno=spj.pno and project.jno=spj.jno;
+------+
| pno  |
+------+
| p4   |
+------+
1 row in set (0.00 sec)
```

10. Get jno values for projects supplied by at least one supplier not in the same city.

```
mysql> select distinct(project.jno) from project, supplier, spj where supplier.city!=project.city and su
pplier.sno=spj.sno and spj.jno=project.jno;
+------+
| jno  |
+------+
| j3   |
| j4   |
| j1   |
| j2   |
+------+
4 rows in set (0.00 sec)
```

11. Get all pairs of city values such that a supplier in the first city supplies a project in the second city.

```
mysql> select distinct supplier.city, project.city from supplier, project, spj where spj.sno=supplier.sn
o and spj.jno=project.jno and project.city!=supplier.city;
+-----------+-----------+
| city      | city      |
+-----------+-----------+
| Bangalore | Delhi     |
| Chennai   | Delhi     |
| Mumbai    | Delhi     |
| Mumbai    | Bangalore |
| Delhi     | Bangalore |
| Chennai   | Chandigarh |
| Bangalore | Chandigarh |
+-----------+-----------+
7 rows in set (0.00 sec)
```

12. Get sno values for suppliers who supply the same part to all projects.

```
mysql> SELECT SNO FROM SPJ
    -> GROUP BY SNO, PNO
    -> HAVING COUNT(DISTINCT JNO) = (SELECT COUNT(*) FROM PROJECT);
Empty set (0.01 sec)
```

13. Get pno values for parts supplied to all projects in Delhi.

```
mysql> select parts.pno from parts, project, spj where parts.pno=spj.pno and spj.jno=project.jno and pro
ject.city='Delhi';
+------+
| pno  |
+------+
| p1   |
| p1   |
| p3   |
| p4   |
+------+
4 rows in set (0.00 sec)
```

14. Get sname values for suppliers who supplies at least one red part to any project.

```
mysql> select distinct(supplier.sname) from supplier, parts, spj where supplier.sno=spj.sno and parts.pn
o=spj.pno and parts.color='red';
+-----------+
| sname     |
+-----------+
| Sanskriti |
| Karan     |
+-----------+
2 rows in set (0.00 sec)
```

15. Get total quantity of part p1 supplied by supplier s1.

```
mysql> select count(pno) as total_quantity from spj where sno='s1' and pno='p1';
+----------------+
| total_quantity |
+----------------+
|              1 |
+----------------+
1 row in set (0.00 sec)
```

16. Get the total number of projects supplied by supplier s3.

```
mysql> select count(jno) as total_projects from spj where sno='s3';
+----------------+
| total_projects |
+----------------+
|              2 |
+----------------+
1 row in set (0.00 sec)
```

17. Change color of all red parts to orange.

```
mysql> update parts set color='orange' where color='red'; select * from parts;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+------+-------+--------+--------+------------+
| pno  | pname | color  | weight | city       |
+------+-------+--------+--------+------------+
| p1   | part1 | orange |     14 | Delhi      |
| p2   | part2 | blue   |     16 | Chennai    |
| p3   | part3 | green  |      5 | Bangalore  |
| p4   | part4 | yellow |     44 | Chandigarh |
+------+-------+--------+--------+------------+
4 rows in set (0.00 sec)
```

18. Get sname values for suppliers who supply to both projects j1 and j2.

```
mysql> select distinct(sname) from supplier, spj where supplier.sno=spj.sno and (spj.jno='j1' or spj.jno
='j2');
+-----------+
| sname     |
+-----------+
| Sanskriti |
| Karan     |
| Atharva   |
| Samaira   |
+-----------+
4 rows in set (0.00 sec)
```

19. Get all city, pno, city triples such that a supplier in the first city supplies the specified part to a project in the second city.

```
mysql> SELECT DISTINCT S.CITY AS SupplierCity, SPJ.PNO, PJ.CITY AS ProjectCity
    -> FROM SUPPLIER S
    -> INNER JOIN SPJ ON S.SNO = SPJ.SNO
    -> INNER JOIN PARTS P ON SPJ.PNO = P.PNO
    -> INNER JOIN PROJECT PJ ON SPJ.JNO = PJ.JNO
    -> WHERE S.CITY <> PJ.CITY AND P.PNO = 'p1';
+--------------+------+-------------+
| SupplierCity | PNO  | ProjectCity |
+--------------+------+-------------+
| Chennai      | p1   | Delhi       |
+--------------+------+-------------+
```

20. Get jnames for those project which are supplied by supplier XYZ.

```
mysql> select distinct(jname) from project, spj where project.jno=spj.jno and spj.sno='s2';
+----------+
| jname    |
+----------+
| project1 |
| project4 |
+----------+
2 rows in set (0.00 sec)
```

Practical-8
Lab Assignment
(Part 3)

Q. Create the required Tables

Department(Dept_No, Dept_Name)

Employee(E_ID, E_Name, Salary, Hiredate, LOC, Mgr_Eid, Job, Grade, Dept_No)

Output of the Table

```
mysql> create table department(dept_no varchar(5) primary key, dept_name varchar(5));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into department values('D1', 'CSE');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D2', 'IT');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D3', 'AI');
Query OK, 1 row affected (0.00 sec)

mysql> insert into department values('D4', 'DS');
Query OK, 1 row affected (0.00 sec)

mysql> select * from department;
+---------+-----------+
| dept_no | dept_name |
+---------+-----------+
| D1      | CSE       |
| D2      | IT        |
| D3      | AI        |
| D4      | DS        |
+---------+-----------+
4 rows in set (0.00 sec)
```

```
mysql> create table employees (eid varchar(5) primary key, ename varchar(25), salary int, hiredate date,
 loc varchar(25), mgr_eid varchar(25), job varchar(25), grade varchar(5), dept_no varchar(5), foreign ke
y(dept_no) references department(dept_no) on delete set null);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into employees values('e1','Sanskriti', 50000, date'2022-06-01','loc1','mgr1','Software En
gineer','A','D1');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e2','Karan', 40000, date'2022-05-01','loc2','mgr2','Hardware Engine
er','A','D2');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e3','Saumya', 60000, date'2022-02-01','loc3','mgr3','Website Design
er','A','D2');
Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values('e4','Atharva', 30000, date'2022-06-10','loc4','mgr4','ML Engineer',
'A','D3');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from employees;
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job              | grade | dept_no |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| e1  | Sanskriti |  50000 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
4 rows in set (0.00 sec)
```

SQL Queries

1. Display each employee name and hiredate of systems department.

```
mysql> select ename, hiredate from employees;
+-----------+------------+
| ename     | hiredate   |
+-----------+------------+
| Sanskriti | 2022-06-01 |
| Karan     | 2022-05-01 |
| Saumya    | 2022-02-01 |
| Atharva   | 2022-06-10 |
+-----------+------------+
4 rows in set (0.00 sec)
```

2.Write query to calculate length of service of each employee.

```
mysql> select date_format(curdate(), '%Y') - date_format(hiredate, '%Y') as years, date_format(curdate()
, '%m') - date_format(hiredate, '%m') as months, date_format(curdate(), '%d') - date_format(hiredate, '%
d') as days from employees;
+-------+--------+------+
| years | months | days |
+-------+--------+------+
|     1 |      2 |   11 |
|     1 |      3 |   11 |
|     1 |      6 |   11 |
|     1 |      2 |    2 |
+-------+--------+------+
4 rows in set (0.00 sec)
```

3. Find the second maximum salary of all employees.

```
mysql> select salary from employees order by salary desc limit 1,1;
+--------+
| salary |
+--------+
|  50000 |
+--------+
1 row in set (0.00 sec)
```

4.Display all employee name and department name in department name order.

```
mysql> select ename, dept_name from employees, department where department.dept_no=employees.dept_no ord
er by dept_name;
+-----------+-----------+
| ename     | dept_name |
+-----------+-----------+
| Atharva   | AI        |
| Sanskriti | CSE       |
| Karan     | IT        |
| Saumya    | IT        |
+-----------+-----------+
4 rows in set (0.00 sec)
```

5. Find the name of lowest paid employee for each manager.

```
mysql> select mgr_eid, min(salary) from employees where mgr_eid is not null group by mgr_eid order by m
n(salary) desc;
+---------+-------------+
| mgr_eid | min(salary) |
+---------+-------------+
| mgr3    |       60000 |
| mgr1    |       50000 |
| mgr2    |       40000 |
| mgr4    |       30000 |
+---------+-------------+
4 rows in set (0.00 sec)
```

6. Display the department that has no employee.

```
mysql> select * from department where dept_no not in(select dept_no from employees);
+---------+-----------+
| dept_no | dept_name |
+---------+-----------+
| D4      | DS        |
+---------+-----------+
1 row in set (0.00 sec)
```

7. Find the employees who earn the maximum salary in each job type. Sort in descending order of salary.

```
mysql> SELECT e.job, e.ename, e.salary
    -> FROM employees e
    -> JOIN (
    ->     SELECT job, MAX(salary) as max_salary
    ->     FROM employees
    ->     GROUP BY job
    -> ) max_salaries
    -> ON e.job = max_salaries.job AND e.salary = max_salaries.max_salary
    -> ORDER BY e.salary DESC;
+-------------------+-----------+--------+
| job               | ename     | salary |
+-------------------+-----------+--------+
| Website Designer  | Saumya    |  60000 |
| Software Engineer | Sanskriti |  50000 |
| Hardware Engineer | Karan     |  40000 |
| ML Engineer       | Atharva   |  30000 |
+-------------------+-----------+--------+
4 rows in set (0.00 sec)
```

8.In which year did most people joined the company? Display the year and number of employees.

```
mysql> SELECT YEAR(Hiredate) AS JoinYear, COUNT(*) AS NumEmployees
    -> FROM Employees
    -> GROUP BY JoinYear
    -> ORDER BY NumEmployees DESC
    -> LIMIT 1;
+----------+--------------+
| JoinYear | NumEmployees |
+----------+--------------+
|     2022 |            4 |
+----------+--------------+
1 row in set (0.00 sec)
```

9. Display the details of those employees who earn greater than average of their department.

```
mysql> select dept_no, job as dept_name, ename, salary from employees e where salary > (select avg(Salar
y) from employees where employees.dept_no = dept_no) order by dept_no;
+---------+------------------+-----------+--------+
| dept_no | dept_name        | ename     | salary |
+---------+------------------+-----------+--------+
| D1      | Software Engineer | Sanskriti |  50000 |
| D2      | Website Designer  | Saumya    |  60000 |
+---------+------------------+-----------+--------+
2 rows in set (0.00 sec)
```

10. List the employees having salary between 10000 and 20000

```
mysql> select * from employees where salary between 40000 and 60000;
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job              | grade | dept_no |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| e1  | Sanskriti |  50000 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
3 rows in set (0.00 sec)
```

11. Display all employees hired during 1983. those employees who earn greater than average of their department.

```
mysql> SELECT eid, ename, salary, hiredate, LOC, Mgr_Eid, Job, Grade, Dept_No
    -> FROM Employees
    -> WHERE YEAR(Hiredate) = 1983
    -> AND Salary > (
    ->     SELECT AVG(Salary)
    ->     FROM Employees AS subquery
    ->     WHERE subquery.Dept_No = Employees.Dept_No
    -> );
Empty set (0.00 sec)
```

12.Update the salaries of all employees in marketing department & hike it by 15%.

```
mysql> update employees set salary=salary+(0.15*salary) where job='Software Engineer'; select * from emp
loyees where job='Software Engineer';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job              | grade | dept_no |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
+-----+-----------+--------+------------+------+---------+------------------+-------+---------+
1 row in set (0.00 sec)
```

13. Get the gross salaries of all the employees.

```
mysql> select 2.5*salary from employees as gross_salary;
+-------------+
| 2.5*salary |
+-------------+
|   143750.0 |
|   100000.0 |
|   150000.0 |
|    75000.0 |
+-------------+
4 rows in set (0.00 sec)
```

14. Get the names of employees and their manager's name.

```
mysql> SELECT E.ename AS EmployeeName, M.ename AS ManagerName
    -> FROM Employees E
    -> LEFT JOIN Employees M ON E.Mgr_Eid = M.eid;
+--------------+-------------+
| EmployeeName | ManagerName |
+--------------+-------------+
| Sanskriti    | NULL        |
| Karan        | NULL        |
| Saumya       | NULL        |
| Atharva      | NULL        |
+--------------+-------------+
4 rows in set (0.00 sec)
```

15.Display the name, location and department name of all the employees earning more than 1500.

```
mysql> select ename, loc, dept_name from department, employees where department.dept_no=employees.dept_n
o and salary>1500;
+-----------+------+-----------+
| ename     | loc  | dept_name |
+-----------+------+-----------+
| Sanskriti | loc1 | CSE       |
| Karan     | loc2 | IT        |
| Saumya    | loc3 | IT        |
| Atharva   | loc4 | AI        |
+-----------+------+-----------+
4 rows in set (0.00 sec)
```

16. Show all the employees in LOC1.

```
mysql> select ename from employees where loc='loc1';
+------------+
| ename      |
+------------+
| Sanskriti  |
+------------+
1 row in set (0.00 sec)
```

17. List the employees name, job, salary, grade, and department for employees in the company except clerks. Sort on employee names.

```
mysql> select ename, job, salary, grade, dept_name from department, employees where department.dept_no=e
mployees.dept_no and job!='Clerk' order by ename;
+-----------+-------------------+--------+-------+-----------+
| ename     | job               | salary | grade | dept_name |
+-----------+-------------------+--------+-------+-----------+
| Atharva   | ML Engineer       |  30000 | A     | AI        |
| Karan     | Hardware Engineer |  40000 | A     | IT        |
| Sanskriti | Software Engineer |  57500 | A     | CSE       |
| Saumya    | Website Designer  |  60000 | A     | IT        |
+-----------+-------------------+--------+-------+-----------+
4 rows in set (0.00 sec)
```

18. Find the employees who earns the minimum salary for their job. Sort in descending order of salary.

```
mysql> select * from employees where salary in (select min(salary) from employees group by job) order by
 salary desc;
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job               | grade | dept_no |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| e3  | Saumya    |  60000 | 2022-02-01 | loc3 | mgr3    | Website Designer  | A     | D2      |
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
4 rows in set (0.00 sec)
```

19.Find the most recently hired employees in the department order by hiredate.

```
mysql> select * from employees e where hiredate in (select max(hiredate) from employees where e.dept_no
= dept_no) order by hiredate desc;
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| eid | ename     | salary | hiredate   | loc  | mgr_eid | job               | grade | dept_no |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
| e4  | Atharva   |  30000 | 2022-06-10 | loc4 | mgr4    | ML Engineer       | A     | D3      |
| e1  | Sanskriti |  57500 | 2022-06-01 | loc1 | mgr1    | Software Engineer | A     | D1      |
| e2  | Karan     |  40000 | 2022-05-01 | loc2 | mgr2    | Hardware Engineer | A     | D2      |
+-----+-----------+--------+------------+------+---------+-------------------+-------+---------+
3 rows in set (0.00 sec)
```

20.Find out the difference between highest and lowest salaries.

```
mysql> select max(salary) - min(salary) difference from employees;
+------------+
| difference |
+------------+
|      30000 |
+------------+
1 row in set (0.00 sec)
```

Practical 9

Aim: To perform different operations of view on SQL.

Theory Required:

Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition

There are few operations that can be carried on views like create, delete and update. These will be shown in the commands below.

Commands:

1. **CREATE VIEW:**
   A View can be created from a single table or multiple tables.
   - **Single Table:**

```
mysql> create database views;
Query OK, 1 row affected (0.11 sec)

mysql> use views;
Database changed
mysql> create table Students( StudentId int, StudentName varchar(50), City varchar(50));
Query OK, 0 rows affected (0.06 sec)

mysql> Insert into Students(StudentId, StudentName, City)
    -> Values(1, 'Arnav Singh', 'Gurgaon'),
    -> (2,'Saumya Kapoor','Noida'),
    -> (3, 'Rohit Sharma', 'Delhi'),
    -> (4, 'Shardul Thakur','Mumbai'),
    -> (5, 'Kunal Sharma','Gurgaon');
Query OK, 5 rows affected (0.03 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> Create View Details as
    -> select StudentName, City
    -> from Students
    -> where StudentId>1;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from DetailsView;
Empty set (0.00 sec)

mysql> select * from Details;
+----------------+---------+
| StudentName    | City    |
+----------------+---------+
| Saumya Kapoor  | Noida   |
| Rohit Sharma   | Delhi   |
| Shardul Thakur | Mumbai  |
| Kunal Sharma   | Gurgaon |
+----------------+---------+
4 rows in set (0.00 sec)
```

   - **Multiple Tables:**

```
mysql> create table marks( StudentId int, StudentName varchar(50), Marks int, Age int);
Query OK, 0 rows affected (0.02 sec)

mysql> Insert into marks(StudentId, StudentName, Marks, Age)
    -> Values(1,'Arnav Singh',67,16),
    -> (2,'Saumya Kapoor',77,15),
    -> (3,'Rohit Sharma',89,17),
    -> (4,'Shardul Thakur',65,16),
    -> (5,'Kunal Sharma',90,17);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> create View MarksView as
    -> Select Students.StudentName,Students.City,marks.Marks
    -> from Students,marks
    -> where Students.StudentName=marks.StudentName;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from MarksView;
+---------------+---------+-------+
| StudentName   | City    | Marks |
+---------------+---------+-------+
| Arnav Singh   | Gurgaon |    67 |
| Saumya Kapoor | Noida   |    77 |
| Rohit Sharma  | Delhi   |    89 |
| Shardul Thakur| Mumbai  |    65 |
| Kunal Sharma  | Gurgaon |    90 |
+---------------+---------+-------+
5 rows in set (0.01 sec)
```

2. **DROP VIEW**

```
mysql> drop view MarksView;
Query OK, 0 rows affected (0.01 sec)
```

3. **UPDATE VIEW**

```
mysql> Create or replace view Details as
    -> Select Students.StudentName,Students.City,
    -> marks.Marks, marks.Age
    -> from Students, marks
    -> where Students.StudentName = marks.StudentName;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from marks;
+-----------+---------------+-------+------+
| StudentId | StudentName   | Marks | Age  |
+-----------+---------------+-------+------+
|         1 | Arnav Singh   |    67 |   16 |
|         2 | Saumya Kapoor |    77 |   15 |
|         3 | Rohit Sharma  |    89 |   17 |
|         4 | Shardul Thakur|    65 |   16 |
|         5 | Kunal Sharma  |    90 |   17 |
+-----------+---------------+-------+------+
5 rows in set (0.00 sec)
```

4. **INSERTING VIEW**

```
mysql> insert into MarksView(StudentName,Marks)
    -> Values('Naman Gupta',92);
Query OK, 1 row affected (0.01 sec)

mysql> select * from MarksView;
+---------------+-------+
| StudentName   | Marks |
+---------------+-------+
| Saumya Kapoor |    77 |
| Rohit Sharma  |    89 |
| Kunal Sharma  |    90 |
| Naman Gupta   |    92 |
+---------------+-------+
4 rows in set (0.00 sec)
```

Conclusion: The different views were created and implemented on mySQL.

Practical 10

Aim: To perform procedure in PLSQL.

Theory Required:

The PL/SQL stored procedure or simply a procedure is a PL/SQL block which performs one or more specific tasks. It is just like procedures in other programming languages.

The procedure contains a header and a body.

- **Header:** The header contains the name of the procedure and the parameters or variables passed to the procedure.
- **Body:** The body contains a declaration section, execution section and exception section similar to a general PL/SQL block.

Commands:

```
DECLARE
  message  varchar2(20):= 'Hello, World!';
  BEGIN
  dbms_output.put_line(message);
END;
/
```

```
Hello, World!

PL/SQL procedure successfully completed.
```

- **Comments**

```
DECLARE
  -- variable declaration
  message  varchar2(20):= 'Hello, World!';
BEGIN
  /*
  * PL/SQL executable statement(s)
  */
  dbms_output.put_line(message);
END;
/
```

```
Hello World

PL/SQL procedure successfully completed.
```

- **Example**

```
DECLARE
  a integer := 30;
  b integer := 40;
  c integer;
  f real;
BEGIN
  c := a + b;
  dbms_output.put_line('Value of c: ' || c);
  f := 100.0/3.0;
  dbms_output.put_line('Value of f: ' || f);
END;
```

```
Value of c: 70
Value of f: 33.333333333333333333

PL/SQL procedure successfully completed.
```

- **Variable Attributes**

% TYPE

```
DECLARE
SALARY EMP.SAL % TYPE;
ECODE EMP.empno % TYPE;
BEGIN
Ecode :=&Ecode;
Select SAL into SALARY from EMP where EMPNO = ECODE;
dbms_output.put_line('Salary of ' || ECODE || 'is = || salary');
END;
```

```
Enter value for ecode : 7499
Salary of 7499 is = 1600
PL/SQL procedure successfully completed.
```

%ROWTYPE

```
DECLARE
EMPLOYEE EMP. % ROW TYPE;
BEGIN
EMPLOYEE.EMPNO := 2092;
5  EMPLOYEE.ENAME := 'Sanju';
Insert into EMP where (EMPNO, ENAME) Values (employee.empno, employee.ename);
dbms_output.put_line('Row Inserted');
END;
```

```
Row Inserted
PL/SQL procedure successfully completed.
```

- **Conditionals**

1) IF -THEN-ELSE

```
DECLARE
  a number(3) := 500;
BEGIN
  -- check the boolean condition using if statement
  IF( a < 20 ) THEN
    -- if condition is true then print the following
    dbms_output.put_line('a is less than 20 ' );
  ELSE
    dbms_output.put_line('a is not less than 20 ' );
  END IF;
  dbms_output.put_line('value of a is : ' || a);
END;
```

```
a is not less than 20
value of a is : 500
PL/SQL procedure successfully completed.
```

2) CASE

```
DECLARE
  grade char(1) := 'A';
BEGIN
  CASE grade
    when 'A' then dbms_output.put_line('Excellent');
    when 'B' then dbms_output.put_line('Very good');
    when 'C' then dbms_output.put_line('Good');
    when 'D' then dbms_output.put_line('Average');
    when 'F' then dbms_output.put_line('Passed with Grace');
    else dbms_output.put_line('Failed');
  END CASE;
END;
```

```
Excellent
PL/SQL procedure successfully completed.
```

❖ **Loop**

1) FOR

```
DECLARE
VAR1 NUMBER;
BEGIN
VAR1:=10;
FOR VAR2 IN 1..10
LOOP
DBMS_OUTPUT.PUT_LINE (VAR1*VAR2);
END LOOP;
END;
```

```
10
20
30
40
50
60
70
80
90
100
```

2) WHILE

```
DECLARE
VAR1 NUMBER;
VAR2 NUMBER;
BEGIN
VAR1:=200;
VAR2:=1;
WHILE (VAR2<=10)
LOOP
DBMS_OUTPUT.PUT_LINE (VAR1*VAR2);
VAR2:=VAR2+1;
END LOOP;

END;
```

```
200
400
600
800
1000
1200
1400
1600
1800
2000
```

Practical 3 and 4

Aim: To perform the primary and foreign key in the table and perform various joints.

Theory Required:

1. **Primary Key:** The primary key constraint uniquely identifies each record in a table. Primary keys must contain UNIQUE values and cannot have a NULL value. A table is restricted to have only 1 primary key but that primary key can have multiple fields(columns).

2. **Foreign Key:** A foreign key is a field or a collection of fields in a table that refers to the primary key in another table. The table with foreign key is known as the child table and the table with primary key is called parent/referenced table.

3. **Joints:**
   a) **Inner Join:** The INNER JOIN keyword selects all rows from both the tables if the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

   b) **Left Join:** This join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN.

   c) **Right Join:** RIGHT JOIN is like LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.

   d) **Full Join:** FULL JOIN creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain NULL values.

Commands:

```
mysql> create Database One;
Query OK, 1 row affected (0.02 sec)

mysql> use One;
Database changed
mysql> create Table Customers(
    -> cid int not null primary key,
    -> cname varchar(100),
    -> cemail varchar(100) );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Customers;
+--------+--------------+------+-----+---------+-------+
| Field  | Type         | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| cid    | int          | NO   | PRI | NULL    |       |
| cname  | varchar(100) | YES  |     | NULL    |       |
| cemail | varchar(100) | YES  |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
```

```
4 rows in set (0.00 sec)

mysql> insert into Customers values(1,'Aman','aman23@gmail.com'),
    -> (2,'Sonia','son278@gmail.com');
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Orders value( 1, '2020/07/18', 29, 1),
    -> (3, '2020/09/11' , 39, 2);
Query OK, 2 rows affected, 2 warnings (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 2

mysql> select * from Customers;
+-----+-------+------------------+
| cid | cname | cemail           |
+-----+-------+------------------+
|   1 | Aman  | aman23@gmail.com |
|   2 | Sonia | son278@gmail.com |
+-----+-------+------------------+
2 rows in set (0.00 sec)
```

```
mysql> select * from Orders;
+-----+------------+---------+------+
| oid | orderdate  | oamount | cid  |
+-----+------------+---------+------+
|   1 | 2020-07-18 |      29 |    1 |
|   3 | 2020-09-11 |      39 |    2 |
+-----+------------+---------+------+
2 rows in set (0.00 sec)

mysql> select * from Orders,Customers;
+-----+------------+---------+------+-----+-------+------------------+
| oid | orderdate  | oamount | cid  | cid | cname | cemail           |
+-----+------------+---------+------+-----+-------+------------------+
|   3 | 2020-09-11 |      39 |    2 |   1 | Aman  | aman23@gmail.com |
|   1 | 2020-07-18 |      29 |    1 |   1 | Aman  | aman23@gmail.com |
|   3 | 2020-09-11 |      39 |    2 |   2 | Sonia | son278@gmail.com |
|   1 | 2020-07-18 |      29 |    1 |   2 | Sonia | son278@gmail.com |
+-----+------------+---------+------+-----+-------+------------------+
4 rows in set (0.01 sec)

mysql> select * from Customers,Orders;
+-----+-------+------------------+-----+------------+---------+------+
| cid | cname | cemail           | oid | orderdate  | oamount | cid  |
+-----+-------+------------------+-----+------------+---------+------+
|   2 | Sonia | son278@gmail.com |   1 | 2020-07-18 |      29 |    1 |
|   1 | Aman  | aman23@gmail.com |   1 | 2020-07-18 |      29 |    1 |
|   2 | Sonia | son278@gmail.com |   3 | 2020-09-11 |      39 |    2 |
|   1 | Aman  | aman23@gmail.com |   3 | 2020-09-11 |      39 |    2 |
+-----+-------+------------------+-----+------------+---------+------+
```

```
mysql> select * from Customers,Orders
    -> where Customers.cid=Orders.cid;
+-----+-------+------------------+-----+------------+---------+------+
| cid | cname | cemail           | oid | orderdate  | oamount | cid  |
+-----+-------+------------------+-----+------------+---------+------+
|   1 | Aman  | aman23@gmail.com |   1 | 2020-07-18 |      29 |    1 |
|   2 | Sonia | son278@gmail.com |   3 | 2020-09-11 |      39 |    2 |
+-----+-------+------------------+-----+------------+---------+------+
2 rows in set (0.00 sec)

mysql> select * from Customers
    -> join Orders
    -> on Customers.cid=Orders.cid;
+-----+-------+------------------+-----+------------+---------+------+
| cid | cname | cemail           | oid | orderdate  | oamount | cid  |
+-----+-------+------------------+-----+------------+---------+------+
|   1 | Aman  | aman23@gmail.com |   1 | 2020-07-18 |      29 |    1 |
|   2 | Sonia | son278@gmail.com |   3 | 2020-09-11 |      39 |    2 |
+-----+-------+------------------+-----+------------+---------+------+
2 rows in set (0.00 sec)

mysql> select * from Customers
    -> left join Orders
    -> on Customers.cid=Orders.cid;
+-----+-------+------------------+------+------------+---------+------+
| cid | cname | cemail           | oid  | orderdate  | oamount | cid  |
+-----+-------+------------------+------+------------+---------+------+
|   1 | Aman  | aman23@gmail.com |    1 | 2020-07-18 |      29 |    1 |
|   2 | Sonia | son278@gmail.com |    3 | 2020-09-11 |      39 |    2 |
+-----+-------+------------------+------+------------+---------+------+
2 rows in set (0.00 sec)
```

```
mysql> select Customers.cid,cname,oamount
    -> from Customers
    -> left join Orders
    -> on Customers.cid=Orders.cid;
+-----+-------+---------+
| cid | cname | oamount |
+-----+-------+---------+
|   1 | Aman  |      29 |
|   2 | Sonia |      39 |
+-----+-------+---------+
2 rows in set (0.00 sec)

mysql> select * from Customers
    -> right join Orders
    -> on Customers.cid = Orders.cid;
+------+-------+------------------+-----+------------+---------+------+
| cid  | cname | cemail           | oid | orderdate  | oamount | cid  |
+------+-------+------------------+-----+------------+---------+------+
|    1 | Aman  | aman23@gmail.com |   1 | 2020-07-18 |      29 |    1 |
|    2 | Sonia | son278@gmail.com |   3 | 2020-09-11 |      39 |    2 |
+------+-------+------------------+-----+------------+---------+------+
2 rows in set (0.00 sec)
```

```
mysql> select Customers.cid,cname,oamount
    -> from Customers
    -> right join Orders
    -> on Customers.cid = Orders.cid;
+------+-------+---------+
| cid  | cname | oamount |
+------+-------+---------+
|    1 | Aman  |      29 |
|    2 | Sonia |      39 |
+------+-------+---------+
2 rows in set (0.00 sec)

mysql> select Customers.cid,cname,oamount
    -> from Customers
    -> full join Orders
    -> on Customers.cid = Orders.cid;
ERROR 1054 (42S22): Unknown column 'Customers.cid' in 'field list'
mysql>
```

Conclusion: Performed the primary and foreign key in the table and perform various Joints.

Practical -4

Lab Assignment- Ques 1

Create the following tables

Course(course_no char(4), course_name varchar(20))
Course_fee(course_no char(4), full_part char(1) (F/P), fees number(10))
course_no and full_part should be unique
Student(prospectus_no number(10), name varchar(20), address varchar(30), phone_no number(11), D_O_B date, total_amt number(10,2), amt_paid number(10,2), installment char(1) (I/F))
Installment(prospectus_no number(10) (foreign key) on delete cascade, installment_amt number(10,2), due_dt date, paid char(1) (P,U))
prospectus_no and due_dt should be unique
Course_taken(prospectus_no number(10) (foreign key), course_no char(4), start_dt date, full_part char(1) (F/P), time_slot char(2), performance varchar(20))

Outputs of the Table:

```
mysql> create table Course(course_no char(4) primary key, course_name varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into course values('c1','DBMS');
Query OK, 1 row affected (0.01 sec)

mysql> insert into course values('c2','ADA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c3','JAVA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c4','C++');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c5','EN');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c6','C');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c7','R');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c8','BEE');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c9','DECO');
Query OK, 1 row affected (0.00 sec)

mysql> insert into course values('c10','MATH');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from course;
+-------------+---------------+
| course_no   | course_name   |
+-------------+---------------+
| c1          | DBMS          |
| c10         | MATH          |
| c2          | ADA           |
| c3          | JAVA          |
| c4          | C++           |
| c5          | EN            |
| c6          | C             |
| c7          | R             |
| c8          | BEE           |
| c9          | DECO          |
+-------------+---------------+
10 rows in set (0.00 sec)
```

```
mysql> create table course_fee(course_no char(4) primary key, full_part char(1), fees numeric
(10), foreign key(course_no) references course(course_no), check(full_part='F' or full_part='
P'));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into course_fee values('c1','F',10000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c2','P',40000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c3','F',6000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c4','P',12000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c5','P',10000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c6','F',23000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c7','F',98600);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c8','F',87000);
Query OK, 1 row affected (0.00 sec)

mysql> insert into course_fee values('c9','P',695000);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into course_fee values('c10','F',3000);
Query OK, 1 row affected (0.00 sec)

mysql> select * from course_fee;
+-----------+-----------+--------+
| course_no | full_part | fees   |
+-----------+-----------+--------+
| c1        | F         |  10000 |
| c10       | F         |   3000 |
| c2        | P         |  40000 |
| c3        | F         |   6000 |
| c4        | P         |  12000 |
| c5        | P         |  10000 |
| c6        | F         |  23000 |
| c7        | F         |  98600 |
| c8        | F         |  87000 |
| c9        | P         | 695000 |
+-----------+-----------+--------+
10 rows in set (0.00 sec)
```

```
mysql> create table student(prospectus_no numeric(11) primary key, st_name varchar(20
), address varchar(30), phone_no numeric(11), D_O_B date, total_amt numeric(10,2), am
t_paid numeric(10,2), installment char(1), check(installment='I' or installment='F'))
;
Query OK, 0 rows affected (0.01 sec)

mysql> insert into student values(32456,'Sanskriti','Delhi',9876899910,date'2003-02-27',20000,10000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(42849,'Saumya','Delhi',9698270918,date'1995-11-29',20000,20000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(32345,'Parth','Mumbai',7298190145,date'2000-02-18',40000,6000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(42345,'Arnav','Bangalore',9278190187,date'1999-05-10',120000,12000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(52345,'Samiksha','Mumbai',7187298605,date'2000-01-15',120000,12000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(89172,'Hardik','Chennai',8920186528,date'1995-08-07',679893,60000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(76289,'Lakshay','Jaipur',8729017345,date'2002-04-09',34000,30000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(23487,'Rohan','Mumbai',9817829076,date'2001-11-11',78653,8000,'F');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(91829,'Harshita','Chennai',9879276422,date'2002-03-14',65000,7500,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(28397,'Manvi','Bangalore',9810255632,date'2001-08-22',20000,20000,'I');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values(65784,'Shaurya','Delhi',9917289015,date'2000-09-21',10620,2300,'F');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from student;
+--------------+-----------+-----------+------------+------------+-----------+----------+-------------+
| prospectus_no | st_name   | address   | phone_no   | D_O_B      | total_amt | amt_paid | installment |
+--------------+-----------+-----------+------------+------------+-----------+----------+-------------+
|         23487 | Rohan     | Mumbai    | 9817829076 | 2001-11-11 |  78653.00 |  8000.00 | F           |
|         28397 | Manvi     | Bangalore | 9810255632 | 2001-08-22 |  20000.00 | 20000.00 | I           |
|         32345 | Parth     | Mumbai    | 7298190145 | 2000-02-18 |  40000.00 |  6000.00 | F           |
|         32456 | Sanskriti | Delhi     | 9876899910 | 2003-02-27 |  20000.00 | 10000.00 | I           |
|         42345 | Arnav     | Bangalore | 9278190187 | 1999-05-10 | 120000.00 | 12000.00 | F           |
|         42849 | Saumya    | Delhi     | 9698270918 | 1995-11-29 |  20000.00 | 20000.00 | I           |
|         52345 | Samiksha  | Mumbai    | 7187298605 | 2000-01-15 | 120000.00 | 12000.00 | F           |
|         65784 | Shaurya   | Delhi     | 9917289015 | 2000-09-21 |  10620.00 |  2300.00 | F           |
|         76289 | Lakshay   | Jaipur    | 8729017345 | 2002-04-09 |  34000.00 | 30000.00 | I           |
|         89172 | Hardik    | Chennai   | 8920186528 | 1995-08-07 | 679893.00 | 60000.00 | F           |
|         91829 | Harshita  | Chennai   | 9879276422 | 2002-03-14 |  65000.00 |  7500.00 | I           |
+--------------+-----------+-----------+------------+------------+-----------+----------+-------------+
11 rows in set (0.00 sec)
```

```
mysql> create table installment(prospectus_no numeric(10) unique, installment_amt num
eric(10,2), due_dt date unique, paid char(1), foreign key(prospectus_no) references s
tudent(prospectus_no) on delete cascade, check(paid='P' or paid='U'));
Query OK, 0 rows affected (0.02 sec)

mysql> insert into installment values(32456, 50000 ,date'2023-01-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(42849, 20000 ,date'2023-02-01','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(32345, 150000 ,date'2023-03-01','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(42345, 120000 ,date'2023-04-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(52345, 60000 ,date'2023-05-01','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(89172, 60000 ,date'2023-04-27','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(76289, 98000 ,date'2022-12-24','U');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(23487, 56800 ,date'2022-11-14','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(91829, 78900 ,date'2023-09-07','P');
Query OK, 1 row affected (0.00 sec)

mysql> insert into installment values(28397, 897200 ,date'2023-07-25','U');
Query OK, 1 row affected (0.00 sec)

mysql> ^C
mysql> insert into installment values(65784, 45790 ,date'2022-12-29','U');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from installment;
+---------------+-----------------+------------+------+
| prospectus_no | installment_amt | due_dt     | paid |
+---------------+-----------------+------------+------+
|         32456 |        50000.00 | 2023-01-01 | U    |
|         42849 |        20000.00 | 2023-02-01 | P    |
|         32345 |       150000.00 | 2023-03-01 | P    |
|         42345 |       120000.00 | 2023-04-01 | U    |
|         52345 |        60000.00 | 2023-05-01 | U    |
|         89172 |        60000.00 | 2023-04-27 | U    |
|         76289 |        98000.00 | 2022-12-24 | U    |
|         23487 |        56800.00 | 2022-11-14 | P    |
|         91829 |        78900.00 | 2023-09-07 | P    |
|         28397 |       897200.00 | 2023-07-25 | U    |
|         65784 |        45790.00 | 2022-12-29 | U    |
+---------------+-----------------+------------+------+
11 rows in set (0.00 sec)
```

```
mysql> create table Course_taken(prospectus_no numeric(10), course_no char(4), start_dt date,
 full_part char(1), time_slot char(2), performance varchar(20), foreign key(prospectus_no) RE
FERENCES STUDENT(prospectus_no), foreign key(course_no) references course(course_no), check(f
ull_part='F' or full_part='P'));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into Course_taken values(32456, 'c1', date'2021-09-29', 'F', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> ^C
mysql> insert into Course_taken values(42849, 'c2', date'2021-05-23', 'F', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(32345, 'c3', date'2021-08-19', 'F', 'FN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(42345, 'c4', date'2021-03-23', 'P', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(89172, 'c1', date'2021-07-14', 'F', 'AN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(76289, 'c3', date'2021-05-23', 'P', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(23487, 'c2', date'2021-04-16', 'P', 'AN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(91829, 'c5', date'2021-07-13', 'F', 'FN','BAD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(28397, 'c9', date'2021-11-17', 'F', 'FN','GOOD');
Query OK, 1 row affected (0.00 sec)

mysql> insert into Course_taken values(65784, 'c10', date'2021-12-12', 'P', 'AN','GOOD');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from Course_taken;
+--------------+-----------+------------+-----------+-----------+-------------+
| prospectus_no | course_no | start_dt   | full_part | time_slot | performance |
+--------------+-----------+------------+-----------+-----------+-------------+
|         32456 | c1        | 2021-09-29 | F         | AN        | GOOD        |
|         42849 | c2        | 2021-05-23 | F         | FN        | GOOD        |
|         32345 | c3        | 2021-08-19 | F         | FN        | BAD         |
|         42345 | c4        | 2021-03-23 | P         | AN        | GOOD        |
|         89172 | c1        | 2021-07-14 | F         | AN        | BAD         |
|         76289 | c3        | 2021-05-23 | P         | FN        | GOOD        |
|         23487 | c2        | 2021-04-16 | P         | AN        | BAD         |
|         91829 | c5        | 2021-07-13 | F         | FN        | BAD         |
|         28397 | c9        | 2021-11-17 | F         | FN        | GOOD        |
|         65784 | c10       | 2021-12-12 | P         | AN        | GOOD        |
+--------------+-----------+------------+-----------+-----------+-------------+
10 rows in set (0.00 sec)
```

SQL Queries:

1. Retrieve name and course no of all the students.

```
mysql> select student.st_name, Course_taken.course_no from student, Course_taken where studen
t.prospectus_no=Course_taken.prospectus_no;
+-----------+-----------+
| st_name   | course_no |
+-----------+-----------+
| Sanskriti | c1        |
| Saumya    | c2        |
| Parth     | c3        |
| Arnav     | c4        |
| Hardik    | c1        |
| Lakshay   | c3        |
| Rohan     | c2        |
| Harshita  | c5        |
| Manvi     | c9        |
| Shaurya   | c10       |
+-----------+-----------+
10 rows in set (0.01 sec)
```

2. List the names of students who have paid the full amount at the time of admission.

```
mysql> select st_name from student where total_amt=amt_paid;
+---------+
| st_name |
+---------+
| Manvi   |
| Saumya  |
+---------+
2 rows in set (0.00 sec)
```

3. Find the names of students starting with A.

```
mysql> select * from student where st_name like 'A%';
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
| prospectus_no | st_name | address   | phone_no   | D_O_B      | total_amt | amt_paid | inst
allment |
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
|         42345 | Arnav   | Bangalore | 9278190187 | 1999-05-10 | 120000.00 | 12000.00 | F
        |
+---------------+---------+-----------+------------+------------+-----------+----------+-----
--------+
1 row in set (0.00 sec)
```

4. Print the names of students whose total amount is not equal to amount due.

```
mysql> select student.st_name from student where amt_paid not in(select amt_paid from student
 where total_amt=amt_paid);
+-----------+
| st_name   |
+-----------+
| Rohan     |
| Parth     |
| Sanskriti |
| Arnav     |
| Samiksha  |
| Shaurya   |
| Lakshay   |
| Hardik    |
| Harshita  |
+-----------+
9 rows in set (0.00 sec)
```

5. Count the number of students who have joined in current year, current month.

```
mysql> select COUNT(*) from Course_taken where start_dt like '2023-05-__';
+----------+
| COUNT(*) |
+----------+
|        0 |
+----------+
1 row in set (0.00 sec)
```

6. Determine the maximum and minimum course fees.

```
mysql> select max(fees), min(fees) from course_fee;
+------------+------------+
| max(fees) | min(fees) |
+------------+------------+
|    695000 |      3000 |
+------------+------------+
1 row in set (0.00 sec)
```

7. Increase the fee of oracle by 50%.

```
mysql> update course_fee set fees=1.5*fees where course_no in(select course_no from Course wh
ere Course.course_name='DBMS'); select * from course_fee;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+-----------+-----------+--------+
| course_no | full_part | fees   |
+-----------+-----------+--------+
| c1        | F         |  15000 |
| c10       | F         |   3000 |
| c2        | P         |  40000 |
| c3        | F         |   6000 |
| c4        | P         |  12000 |
| c5        | P         |  10000 |
| c6        | F         |  23000 |
| c7        | F         |  98600 |
| c8        | F         |  87000 |
| c9        | P         | 695000 |
+-----------+-----------+--------+
10 rows in set (0.00 sec)
```

8. Print the details of courses whose fees are between 5000 and 10000.

```
mysql> select Course.course_no, Course.course_name, course_fee.fees, course_fee.full_part fro
m Course, course_fee where Course.course_no=course_fee.course_no and(course_fee.fees>5000 and
 course_fee.fees<10000);
+-----------+-------------+------+-----------+
| course_no | course_name | fees | full_part |
+-----------+-------------+------+-----------+
| c3        | JAVA        | 6000 | F         |
+-----------+-------------+------+-----------+
1 row in set (0.00 sec)
```

9. Display the admission date in Date, Month, Year format.

```
mysql> select start_dt as start_date from Course_taken;
+------------+
| start_date |
+------------+
| 2021-09-29 |
| 2021-05-23 |
| 2021-08-19 |
| 2021-03-23 |
| 2021-07-14 |
| 2021-05-23 |
| 2021-04-16 |
| 2021-07-13 |
| 2021-11-17 |
| 2021-12-12 |
+------------+
10 rows in set (0.00 sec)
```

10. Find out in which course maximum number of students have taken admission.

```
mysql> select Course.course_name from Course,Course_taken where Course.course_no=Course_taken
.course_no group by Course.course_name having count(Course_taken.prospectus_no)>=all (select
count(prospectus_no) from Course_taken group by course_no);
+-------------+
| course_name |
+-------------+
| DBMS        |
| ADA         |
| JAVA        |
+-------------+
3 rows in set (0.00 sec)
```

11. Change the course_name from C to C Programming.

```
mysql> update Course set course_name='CProgramming' WHERE course_name='C';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from course;
+-----------+--------------+
| course_no | course_name  |
+-----------+--------------+
| c1        | DBMS         |
| c10       | MATH         |
| c2        | ADA          |
| c3        | JAVA         |
| c4        | C++          |
| c5        | EN           |
| c6        | CProgramming |
| c7        | R            |
| c8        | BEE          |
| c9        | DECO         |
+-----------+--------------+
10 rows in set (0.00 sec)
```

12. Display the admission date in DD-MONTH-YYYY format.

```
mysql> select date_format(start_dt, '%d-%m-%y') from Course_taken;
+-----------------------------------+
| date_format(start_dt, '%d-%m-%y') |
+-----------------------------------+
| 29-09-21                          |
| 23-05-21                          |
| 19-08-21                          |
| 23-03-21                          |
| 14-07-21                          |
| 23-05-21                          |
| 16-04-21                          |
| 13-07-21                          |
| 17-11-21                          |
| 12-12-21                          |
+-----------------------------------+
10 rows in set (0.00 sec)
```

13. Get the sum of amount to be collected from students in this month.

```
mysql> select sum(student.total_amt-student.amt_paid) from student,installment where student.
prospectus_no=installment.prospectus_no and installment.due_dt like'__-APR-23';
```

```
727893
```

14. Find out in which course the maximum number of students have taken admission in the current month.

```
mysql> select Course.course_name from Course, Course_taken where Course.course_no=Course_take
n.course_no and start_dt like '__-MAY-21' group by Course.course_name having count(Course_tak
en.prospectus_no)>=all(select count(prospectus_no) from Course_taken group by course_no);
```

15. Select the students who have not yet paid full amount of fees.

```
mysql> select st_name from student where total_amt>amt_paid;
+-----------+
| st_name   |
+-----------+
| Rohan     |
| Parth     |
| Sanskriti |
| Arnav     |
| Samiksha  |
| Shaurya   |
| Lakshay   |
| Hardik    |
| Harshita  |
+-----------+
9 rows in set (0.00 sec)
```

Practical 5

Aim: To perform the various Clauses in SQL.

Theory Required:

1. **ORDER BY:** The ORDER BY keyword is used to sort the result-set in ascending or descending order. The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

2. **GROUP BY:** The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country". The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

3. **HAVING:** The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Commands:

```
mysql> create database two;
Query OK, 1 row affected (0.01 sec)

mysql> use two;
Database changed
mysql> create table children(
    -> rollno int not null,
    -> age int not null,
    -> name varchar(100) not null,
    -> address varchar(200) not null,
    -> phoneno varchar(15) not null,
    -> primary key(rollno) );
Query OK, 0 rows affected (0.03 sec)

mysql> insert into children(rollno,age,name,address,phoneno)
    -> values
    -> (1,15,'Ankur','Sector 22, Gurgaon','9815678290'),
    -> (2,16,'Sameer','Sector 11, Gurgaon','9283373882'),
    -> (3,15,'Yash','Sector 07, Gurgaon','8800116789');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from children order by rollno desc;
+--------+-----+--------+--------------------+------------+
| rollno | age | name   | address            | phoneno    |
+--------+-----+--------+--------------------+------------+
|      3 |  15 | Yash   | Sector 07, Gurgaon | 8800116789 |
|      2 |  16 | Sameer | Sector 11, Gurgaon | 9283373882 |
|      1 |  15 | Ankur  | Sector 22, Gurgaon | 9815678290 |
+--------+-----+--------+--------------------+------------+
3 rows in set (0.00 sec)
```

```
mysql> select name,sum(age) from children
    -> group by name;
+--------+----------+
| name   | sum(age) |
+--------+----------+
| Ankur  |       15 |
| Sameer |       16 |
| Yash   |       15 |
+--------+----------+
3 rows in set (0.01 sec)

mysql> select name, sum(age) as age
    -> from children
    -> group by name
    -> having age >= 15;
+--------+------+
| name   | age  |
+--------+------+
| Ankur  |   15 |
| Sameer |   16 |
| Yash   |   15 |
+--------+------+
3 rows in set (0.00 sec)
```

Conclusion: Performed the various Clauses in SQL.