Practical - 2 Name:Nakshtra Chavan
roll:-L011

1. Count the customers with grades above NewYork average

```
mysql> SELECT grade, COUNT(*) FROM customer GROUP BY grade HAVING grade > (SELECT
AVG(grade) FROM customer WHERE city = 'New York');
+-------+----------+
| grade | COUNT(*) |
+-------+----------+
|   200 |        3 |
|   300 |        2 |
+-------+----------+
2 rows in set (0.02 sec)
```

2.Find the name and numbers of all salesmen who had more than one customer

```
mysql> select salesman_id,name from salesman a where 1<(select count(*) from
customer where salesman_id=a.salesman_id);
+-------------+------------+
| salesman_id | name       |
+-------------+------------+
|        5001 | James Hoog |
|        5002 | Nail Knite |
+-------------+------------+
2 rows in set (0.01 sec)
```

3)Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted

```
mysql> delete from salesman where salesman_id=1000;
Query OK, 0 rows affected (0.00 sec)
```

Q2. Design ERD for the following schema and execute the following Queries on it:

Consider the schema for Movie Database:
ACTOR (Act_id, Act_Name, Act_Gender)
DIRECTOR (Dir_id, Dir_Name, Dir_Phone)
MOVIES (Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)
MOVIE_CAST (Act_id, Mov_id, Role)
RATING (Mov_id, Rev_Stars)

```
mysql> create table Actor(act_id integer primary key,act_name
varchar(100),act_gender varchar(10));
Query OK, 0 rows affected (0.01 sec)

mysql> create table Director(dir_id integer primary key,dir_name
varchar(200),dir_phone varchar(100));
Query OK, 0 rows affected (0.01 sec)


mysql> create table Movies(mov_id integer primary key,mov_title
varchar(255),mov_year year,mov_lang varchar(100),dir_id int, foreign key (dir_id)
references Director(dir_id));
Query OK, 0 rows affected (0.02 sec)

mysql> create table Movie_cast (act_id int,foreign key (act_id) references
Actor(act_id), mov_id int, foreign key(mov_id) references Movies(mov_id),role
varchar(100), primary key(act_id,mov_id) );
Query OK, 0 rows affected (0.02 sec)

mysql> create table Rating(mov_id integer primary key , foreign key(mov_id)
references Movies(mov_id),rev_stars integer);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into Actor values(301, 'anuska','f'),
    -> (302,'PRABHAS','M'),
    -> (303,'PUNITH','M'),
    -> (304,'jermy','M');
Query OK, 4 rows affected (0.03 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> insert into director values(60, 'rajamouli',8751611001),
    -> (61,'HITCHCOCK', 7766138911),
    -> (62,'FARAN', 9986776531),
    -> (63,'STEVEN SPIELBERG', 8989776530);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> insert into movies values(1001,'BAHUBALI-2', 2017, 'TELAGU', 60),
    -> (1002,'BAHUBALI-2', 2015, 'TELAGU', 60),
    -> (1003,'AKASH', 2008, 'KANNADA', 61),
    -> (1004,'WAR HORSE', 2011, 'ENGLISH', 63);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE'),
    -> (301, 1001, 'HEROINE'),
    -> (303, 1003, 'HERO'),
    -> (303, 1002, 'guest'),
```

```
    -> (304, 1004, 'hero');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0


mysql> INSERT INTO RATING VALUES (1001, 4),
    ->  (1002, 2),
    -> (1003, 5),
    -> (1004, 4);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

#Write SQL queries to
1. List the titles of all movies directed by ‘Hitchcock

mysql> select mov_title from movies where dir_id in(select dir_id from director
where dir_name='hitchcock');
+-----------+
| mov_title |
+-----------+
| AKASH     |
+-----------+
1 row in set (0.00 sec)

2. Find the movie names where one or more actors acted in two or more movies.

mysql> select mov_title from movies m,movie_cast mv where m.mov_id=mv.mov_id and
act_id in(select act_id from movie_cast group by act_id having count(act_id)>1)
group by mov_title having count(*)>1;
+------------+
| mov_title  |
+------------+
| BAHUBALI-2 |
+------------+
1 row in set (0.00 sec)

3. List all actors who acted in a movie before 2000 and also in a movie after
2015 (use JOIN operation).

mysql> select a.act_name,c.mov_title,c.mov_year from actor a,movie_cast b,movies c
where a.act_id=b.act_id and b.mov_id=c.mov_id and c.mov_year not between 2000 and
2015;
+----------+------------+----------+
| act_name | mov_title  | mov_year |
+----------+------------+----------+
| anuska   | BAHUBALI-2 |     2017 |
+----------+------------+----------+
1 row in set (0.00 sec)

4. Find the title of movies and number of stars for each movie that has at least one
```

rating and find the highest number of stars that movie received. Sort the result by movie title

```
mysql> select mov_title,max(rev_stars) from movies inner join rating using(mov_id)
group by mov_title having max(rev_stars)>0 order by mov_tit
le;
+------------+----------------+
| mov_title  | max(rev_stars) |
+------------+----------------+
| AKASH      |              5 |
| BAHUBALI-2 |              4 |
| WAR HORSE  |              4 |
+------------+----------------+
3 rows in set (0.00 sec)
```

5. Update rating of all movies directed by 'Steven Spielberg' to 5.

```
mysql> update rating set rev_stars=5 where mov_id in(select mov_id from movies where
dir_id in (select dir_id from director where dir_name='STEVEN SPIELBERG'));
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from rating;
+--------+-----------+
| mov_id | rev_stars |
+--------+-----------+
|   1001 |         4 |
|   1002 |         2 |
|   1003 |         5 |
|   1004 |         5 |
+--------+-----------+
4 rows in set (0.00 sec)
```

3. Design ERD for the following schema and execute the following Queries on it:

```
mysql> CREATE TABLE students (
    ->     stno INT PRIMARY KEY,
    ->     name VARCHAR(50),
    ->     addr VARCHAR(255),
    ->     city VARCHAR(50),
    ->     state VARCHAR(2),
    ->     zip VARCHAR(10)
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE INSTRUCTORS (
    ->     empno INT PRIMARY KEY,
    ->     name VARCHAR(50),
    ->     ranks  VARCHAR(20),
    ->     roomno VARCHAR(10),
```

```
    ->     telno VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE COURSES (
    ->     cno text PRIMARY KEY,
    ->     cname VARCHAR(50),
    ->     cr INT,
    ->     cap INT
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE GRADES (
    ->   stno INT,
    ->   empno INT,
    ->   cno VARCHAR(50),
    ->   sem VARCHAR(10),
    ->   year INT,
    ->   grade INT,
    ->   FOREIGN KEY (stno) REFERENCES students(stno),
    ->   FOREIGN KEY (empno) REFERENCES INSTRUCTORS(empno),
    ->   FOREIGN KEY (cno) REFERENCES COURSES(cno)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE ADVISING (
    ->     stno INT,
    ->     empno INT,
    ->     PRIMARY KEY (stno, empno),
    ->     FOREIGN KEY (stno) REFERENCES students(stno),
    ->     FOREIGN KEY (empno) REFERENCES INSTRUCTORS(empno)
    -> );
Query OK, 0 rows affected (0.02 sec)


mysql> insert into students values
     ->(1011,'edwards p. david','10 red rd','newton','MA','02159')
     ->(2415, 'Grogan A. Mary', '8 Walnut St', 'Malden', 'MA', '02148'),
    -> (2661, 'Mixon Leatha', '100 School St', 'Brookline', 'MA', '02146'),
    -> (2890, 'McLane Sandy', '30 Case Rd', 'Boston', 'MA', '02122'),
    -> (3442, 'Novak Roland', '42 Beacon St', 'Nashua', 'NH', '03060'),
    -> (3566, 'Pierce Richard', '70 Park St', 'Brookline', 'MA', '02146'),
    -> (4022, 'Prior Lorraine', '8 Beacon St', 'Boston', 'MA', '02125'),
    -> (5544, 'Rawlings Jerry', '15 Pleasant Dr', 'Boston', 'MA', '02115'),
    -> (5571, 'Lewis Jerry', '1 Main Rd', 'Providence', 'RI', '02904');

mysql> select * from students;
+------+------------------+----------------+------------+-------+-------+
| stno | name             | addr           | city       | state | zip   |
+------+------------------+----------------+------------+-------+-------+
```

```
| 1011 | edwards p. david | 10 red rd      | newton     | MA    | 02159 |
| 2415 | Grogan A. Mary   | 8 Walnut St    | Malden     | MA    | 02148 |
| 2661 | Mixon Leatha     | 100 School St  | Brookline  | MA    | 02146 |
| 2890 | McLane Sandy     | 30 Case Rd     | Boston     | MA    | 02122 |
| 3442 | Novak Roland     | 42 Beacon St   | Nashua     | NH    | 03060 |
| 3566 | Pierce Richard   | 70 Park St     | Brookline  | MA    | 02146 |
| 4022 | Prior Lorraine   | 8 Beacon St    | Boston     | MA    | 02125 |
| 5544 | Rawlings Jerry   | 15 Pleasant Dr | Boston     | MA    | 02115 |
| 5571 | Lewis Jerry      | 1 Main Rd      | Providence | RI    | 02904 |
+------+------------------+----------------+------------+-------+-------+
9 rows in set (0.00 sec)

mysql> INSERT INTO instructors VALUES
    -> (19, 'Evans Robert', 'Professor', '82', '7122'),
    -> (23, 'Exxon George', 'Professor', '90', '9101'),
    -> (56, 'Sawyer Kathy', 'Assoc Prof', '91', '5110'),
    -> (126, 'Davis William', 'Assoc Prof', '72', '5411'),
    -> (234, 'Will Samuel', 'Assist Prof', '90', '7024');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from instructors;
+-------+---------------+-------------+--------+-------+
| empno | name          | ranks       | roomno | telno |
+-------+---------------+-------------+--------+-------+
|    19 | Evans Robert  | Professor   | 82     | 7122  |
|    23 | Exxon George  | Professor   | 90     | 9101  |
|    56 | Sawyer Kathy  | Assoc Prof  | 91     | 5110  |
|   126 | Davis William | Assoc Prof  | 72     | 5411  |
|   234 | Will Samuel   | Assist Prof | 90     | 7024  |
+-------+---------------+-------------+--------+-------+
5 rows in set (0.00 sec)


mysql> insert into courses values
    -> ('cs110', 'Introduction to Computing', 4, 120),
    -> ('cs210', 'Computer Programming', 4, 100),
    -> ('cs240', 'Computer Architecture', 3, 100),
    -> ('cs310', 'Data Structures', 3, 60),
    -> ('cs350', 'Higher Level Languages', 3, 50),
    -> ('cs410', 'Software Engineering', 3, 40),
    -> ('cs460', 'Graphics', 3, 30);
Query OK, 7 rows affected (0.00 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> select * from courses;
+-------+--------------------------+------+------+
| cno   | cname                    | cr   | cap  |
+-------+--------------------------+------+------+
| cs110 | Introduction to Computing |    4 |  120 |
```

```
| cs210 | Computer Programming      |    4 |  100 |
| cs240 | Computer Architecture     |    3 |  100 |
| cs310 | Data Structures           |    3 |   60 |
| cs350 | Higher Level Languages    |    3 |   50 |
| cs410 | Software Engineering      |    3 |   40 |
| cs460 | Graphics                  |    3 |   30 |
+-------+---------------------------+------+------+
7 rows in set (0.00 sec)

mysql> insert into grades values
    -> (1011, 019, 'cs110', 'Fall', 2001, 40),
    -> (2661, 019, 'cs110', 'Fall', 2001, 80),
    -> (3566, 019, 'cs110', 'Fall', 2001, 95),
    -> (5544, 019, 'cs110', 'Fall', 2001, 100),
    -> (1011, 023, 'cs110', 'Spring', 2002, 75),
    -> (4022, 023, 'cs110', 'Spring', 2002, 60),
    -> (3566, 019, 'cs240', 'Spring', 2002, 100),
    -> (5571, 019, 'cs240', 'Spring', 2002, 50),
    -> (2415, 019, 'cs240', 'Spring', 2002, 100),
    -> (3442, 234,'cs410', 'Spring', 2002, 60),
    -> (5571, 234, 'cs410', 'Spring', 2002, 80),
    -> (1011, 019, 'cs210', 'Fall', 2002, 90),
    -> (2661, 019, 'cs210', 'Fall', 2002, 70),
    -> (3566, 019, 'cs210', 'Fall', 2002, 90),
    -> (5571, 019, 'cs210', 'Spring', 2003, 85),
    -> (4022, 019, 'cs210', 'Spring', 2003, 70),
    -> (5544, 56, 'cs240', 'Spring', 2003, 70),
    -> (1011, 56, 'cs240', 'Spring', 2003, 90),
    -> (4022, 56, 'cs240', 'Spring', 2003, 80),
    -> (2661, 234, 'cs310', 'Spring', 2003, 100),
    -> (4022, 234, 'cs310', 'Spring',2003, 75);
Query OK, 21 rows affected (0.00 sec)
Records: 21  Duplicates: 0  Warnings: 0

mysql> select * from grades;
+------+-------+-------+--------+------+-------+
| stno | empno | cno   | sem    | year | grade |
+------+-------+-------+--------+------+-------+
| 1011 |    19 | cs110 | Fall   | 2001 |    40 |
| 2661 |    19 | cs110 | Fall   | 2001 |    80 |
| 3566 |    19 | cs110 | Fall   | 2001 |    95 |
| 5544 |    19 | cs110 | Fall   | 2001 |   100 |
| 1011 |    23 | cs110 | Spring | 2002 |    75 |
| 4022 |    23 | cs110 | Spring | 2002 |    60 |
| 3566 |    19 | cs240 | Spring | 2002 |   100 |
| 5571 |    19 | cs240 | Spring | 2002 |    50 |
| 2415 |    19 | cs240 | Spring | 2002 |   100 |
| 3442 |   234 | cs410 | Spring | 2002 |    60 |
| 5571 |   234 | cs410 | Spring | 2002 |    80 |
| 1011 |    19 | cs210 | Fall   | 2002 |    90 |
```

```
| 2661 |    19 | cs210 | Fall   | 2002 |    70 |
| 3566 |    19 | cs210 | Fall   | 2002 |    90 |
| 5571 |    19 | cs210 | Spring | 2003 |    85 |
| 4022 |    19 | cs210 | Spring | 2003 |    70 |
| 5544 |    56 | cs240 | Spring | 2003 |    70 |
| 1011 |    56 | cs240 | Spring | 2003 |    90 |
| 4022 |    56 | cs240 | Spring | 2003 |    80 |
| 2661 |   234 | cs310 | Spring | 2003 |   100 |
| 4022 |   234 | cs310 | Spring | 2003 |    75 |
+------+-------+-------+--------+------+-------+
21 rows in set (0.00 sec)

mysql> insert into advising values
    -> (1011,019);
    -> (2415,019),
    -> (2661,0023),
    -> (2890,023),
    -> (3442,0056),
    -> (3566,126),
    -> (4022,234),
    ->  (5544,023),
    -> (5571,234);
Query OK, 8 rows affected (0.00 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> select * from advising;
+------+-------+
| stno | empno |
+------+-------+
| 1011 |    19 |
| 2415 |    19 |
| 2661 |    23 |
| 2890 |    23 |
| 5544 |    23 |
| 3442 |    56 |
| 3566 |   126 |
| 4022 |   234 |
| 5571 |   234 |
+------+-------+
9 rows in set (0.00 sec)


#Queries
1. Find the names of students who took only four-credit courses.

mysql> SELECT DISTINCT s.name
    -> FROM students s
    -> JOIN grades g ON s.stno = g.stno
    -> JOIN courses c ON g.cno = c.cno
    -> WHERE c.cr = 4
```

```
    -> AND g.cno NOT IN (
    ->     SELECT cno
    ->     FROM courses
    ->     WHERE cr != 4
    -> );
+------------------+
| name             |
+------------------+
| edwards p. david |
| Mixon Leatha     |
| Pierce Richard   |
| Rawlings Jerry   |
| Prior Lorraine   |
| Lewis Jerry      |
+------------------+
6 rows in set (0.00 sec)
```

2.Find the names of students who took no four-credit courses.

```
mysql> SELECT DISTINCT s.name
    -> FROM students s
    -> WHERE s.stno NOT IN (
    ->     SELECT DISTINCT g.stno
    ->     FROM grades g
    ->     JOIN courses c ON g.cno = c.cno
    ->     WHERE c.cr = 4
    -> );
+----------------+
| name           |
+----------------+
| Grogan A. Mary |
| McLane Sandy   |
| Novak Roland   |
+----------------+
3 rows in set (0.00 sec)
```

3. Find the names of students who took cs210 or cs310

```
mysql> select name from students where stno in (select stno from grades where
cno='cs210' or cno='cs310');
+------------------+
| name             |
+------------------+
| edwards p. david |
| Mixon Leatha     |
| Pierce Richard   |
| Prior Lorraine   |
| Lewis Jerry      |
+------------------+
5 rows in set (0.00 sec)
```

4. Find names of all students who have a cs210 grade higher than the highest grade given in cs310 and did not take any course with Prof. Evans.

```
mysql> SELECT s.name
    -> FROM students s
    -> WHERE s.stno IN (
    ->     SELECT g1.stno
    ->     FROM grades g1
    ->     WHERE g1.cno = 'cs210'
    ->       AND g1.grade > (
    ->         SELECT MAX(g2.grade)
    ->         FROM grades g2
    ->         WHERE g2.cno = 'cs310'
    ->     )
    -> )
    -> AND s.stno NOT IN (
    ->     SELECT g3.stno
    ->     FROM grades g3
    ->     JOIN instructors i ON g3.empno = i.empno
    ->     WHERE i.name = 'Evans Robert'
    -> );
Empty set (0.00 sec)
```

5.. Find course numbers for courses that enrol at least two students,solve the same query for courses that enroll at least three students

```
mysql> SELECT cno
    -> FROM grades
    -> GROUP BY cno
    -> HAVING COUNT(DISTINCT stno) >= 2;
+-------+
| cno   |
+-------+
| cs110 |
| cs210 |
| cs240 |
| cs310 |
| cs410 |
+-------+
5 rows in set (0.00 sec)

mysql> SELECT cno
    -> FROM grades
    -> GROUP BY cno
    -> HAVING COUNT(DISTINCT stno) >= 3;
+-------+
| cno   |
+-------+
| cs110 |
```

```
| cs210 |
| cs240 |
+-------+
3 rows in set (0.00 sec)
```

6. Find the names of students who obtained the highest grade in cs210.

```
mysql> SELECT s.name
    -> FROM students s
    -> JOIN grades g ON s.stno = g.stno
    -> WHERE g.cno = 'cs210' AND g.grade = (SELECT MAX(grade) FROM grades WHERE cno
= 'cs210');
+------------------+
| name             |
+------------------+
| edwards p. david |
| Pierce Richard   |
+------------------+
2 rows in set (0.00 sec)
```

7. Find course numbers for courses that enroll exactly two students;

```
mysql> SELECT cno
    -> FROM grades
    -> GROUP BY cno
    -> HAVING COUNT(DISTINCT stno) = 2;
+-------+
| cno   |
+-------+
| cs310 |
| cs410 |
+-------+
2 rows in set (0.00 sec)
```

8. Find the names of all students for whom no other student lives in the same city.

```
mysql> SELECT DISTINCT s1.name
    -> FROM students s1
    -> WHERE NOT EXISTS (
    ->     SELECT 1
    ->     FROM students s2
    ->     WHERE s2.city = s1.city AND s2.stno <> s1.stno
    -> );
+------------------+
| name             |
+------------------+
| edwards p. david |
| Grogan A. Mary   |
| Novak Roland     |
| Lewis Jerry      |
```

```
+------------------+
4 rows in set (0.00 sec)

9.Find the names of students whose advisor did not teach them any course

mysql> SELECT s.name
    -> FROM students s
    -> WHERE NOT EXISTS (
    ->     SELECT 1
    ->     FROM advising a
    ->     WHERE a.stno = s.stno
    ->       AND NOT EXISTS (
    ->           SELECT 1
    ->           FROM grades g
    ->           WHERE g.stno = a.stno
    ->             AND g.empno = a.empno
    ->       )
    -> );
+------------------+
| name             |
+------------------+
| edwards p. david |
| Grogan A. Mary   |
| Prior Lorraine   |
| Lewis Jerry      |
+------------------+
4 rows in set (0.00 sec)

10.Find the highest grade of a student who never took cs110
mysql> SELECT MAX(grade) AS highest_grade
    -> FROM grades
    -> WHERE stno NOT IN (
    ->     SELECT stno
    ->     FROM grades
    ->     WHERE cno = 'cs110'
    -> );
+---------------+
| highest_grade |
+---------------+
|           100 |
+---------------+
1 row in set (0.00 sec)
```