A. Consider the following schema for a LibraryDatabase:

BOOK (Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS (Book_id, Author_Name)
PUBLISHER (Name, Address, Phone) BOOK_COPIES
(Book_id, Branch_id, No-of_Copies)
BOOK_LENDING (Book_id, Branch_id, Card_No, Date_Out, Due_Date)
LIBRARY_BRANCH (Branch_id, Branch_Name, Address)

TABLE CREATION

```
create table publisher (
   name varchar(20) primary key,
   phone bigint,
   address varchar(20)
);

create table book (
   book_id int primary key,
   title varchar(20),
   pub_year varchar(20),
   publisher_name varchar(20),
   foreign key (publisher_name) references publisher(name) on delete cascade
);

create table book_authors (
   author_name varchar(20),
   book_id int,
   primary key (book_id, author_name),
   foreign key (book_id) references book(book_id) on delete cascade
);

create table library_branch (
   branch_id int primary key,
```

```
    branch_name varchar(50),
    address varchar(50)
);

create table book_copies (
    no_of_copies int,
    book_id int,
    branch_id int,
    primary key (book_id, branch_id),
    foreign key (book_id) references book(book_id) on delete cascade,
    foreign key (branch_id) references library_branch(branch_id) on delete cascade
);

create table card (
    card_no int primary key
);

create table book_lending (
    date_out date,
    due_date date,
    book_id int,
    branch_id int,
    card_no int,
    primary key (book_id, branch_id, card_no),
    foreign key (book_id) references book(book_id) on delete cascade,
    foreign key (branch_id) references library_branch(branch_id) on delete cascade,
    foreign key (card_no) references card(card_no) on delete cascade
);
```

Insertion of values into tables:

```
insert into publisher values ('mcgraw-hill', 9989076587, 'bangalore');
insert into publisher values ('pearson', 9889076565, 'new delhi');
insert into publisher values ('random house', 7455679345, 'hyderabad');
insert into publisher values ('hachette livre', 8970862340, 'chennai');
```

```
insert into publisher values ('grupo planeta', 7756120238, 'bangalore');

insert into book values (1, 'dbms', 'jan-2017', 'mcgraw-hill');
insert into book values (2, 'adbms', 'jun-2016', 'mcgraw-hill');
insert into book values (3, 'cn', 'sep-2016', 'pearson');
insert into book values (4, 'cg', 'sep-2015', 'grupo planeta');
insert into book values (5, 'os', 'may-2016', 'pearson');

insert into book_authors values ('navathe', 1);
insert into book_authors values ('navathe', 2);
insert into book_authors values ('tanenbaum', 3);
insert into book_authors values ('edward angel', 4);
insert into book_authors values ('galvin', 5);

insert into library_branch values (10, 'rr nagar', 'bangalore');
insert into library_branch values (11, 'rnsit', 'bangalore');
insert into library_branch values (12, 'rajaji nagar', 'bangalore');
insert into library_branch values (13, 'nitte', 'mangalore');
insert into library_branch values (14, 'manipal', 'udupi');

insert into book_copies values (10, 1, 10);
insert into book_copies values (5, 1, 11);
insert into book_copies values (2, 2, 12);
insert into book_copies values (5, 2, 13);
insert into book_copies values (7, 3, 14);
insert into book_copies values (1, 5, 10);
insert into book_copies values (3, 4, 11);

insert into card values (100);
insert into card values (101);
insert into card values (102);
insert into card values (103);
insert into card values (104);
```

```
insert into book_lending values ('2017-01-01', '2017-06-01', 1, 10, 101);
insert into book_lending values ('2017-01-11', '2017-03-11', 3, 14, 101);
insert into book_lending values ('2017-02-21', '2017-04-21', 2, 13, 101);
insert into book_lending values ('2017-03-15', '2017-07-15', 4, 11, 101);
insert into book_lending values ('2017-04-12', '2017-05-12', 1, 11, 104);
```

Write SQL queries to

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch,etc.

```
select
   b.book_id,
   b.title,
   b.publisher_name,
   a.author_name,
   c.no_of_copies,
   l.branch_id
from
   book b,
   book_authors a,
   book_copies c,
   library_branch l
where
   b.book_id = a.book_id
   and b.book_id = c.book_id
   and l.branch_id = c.branch_id;
```

2.Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun2017

```
select card_no
from book_lending
```

```
where date_out between '2017-01-01' and '2017-07-01'
group by card_no
having count(*) > 3;
```

3.Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

```
delete from book
        where book_id=3;

select * from book;
```

4.Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

```
create view v_publication as
select pub_year
from book;

select * from v_publication;
```

5.Create a view of all books and its number of copies that are currently available in the Library.

```
create view v_books as
select b.book_id, b.title, c.no_of_copies
from book b, book_copies c, library_branch l
where b.book_id = c.book_id
and c.branch_id = l.branch_id;

select * from v_books;
```

B. Consider the following schema for OrderDatabase:

SALESMAN (Salesman_id, Name, City, Commission)
CUSTOMER (Customer_id, Cust_Name, City,
Grade,Salesman_id)
ORDERS (Ord_No, Purchase_Amt, Ord_Date, Customer_id,
Salesman_id) Write SQL queries to

TABLE CREATION:

```
create table salesman (
   salesman_id int(4) not null,
   name varchar(20),
   city varchar(20),
   commission varchar(20),
   primary key (salesman_id)
);

create table customer1 (
   customer_id int(4) not null,
   cust_name varchar(20),
   city varchar(20),
   grade int(3),
   salesman_id int(4),
   primary key (customer_id),
   foreign key (salesman_id) references salesman(salesman_id)
      on delete set null
);

create table orders (
   ord_no int(5) not null,
   purchase_amt decimal(10,2),
   ord_date date,
   customer_id int(4),
   salesman_id int(4),
```

```
    primary key (ord_no),
    foreign key (customer_id) references customer1(customer_id)
        on delete cascade,
    foreign key (salesman_id) references salesman(salesman_id)
        on delete cascade
);
```

INSERTION OF VALUES:

```
-- Insert into SALESMAN
insert into salesman values (1000, 'JOHN', 'BANGALORE', '25 %');
insert into salesman values (2000, 'RAVI', 'BANGALORE', '20 %');
insert into salesman values (3000, 'KUMAR', 'MYSORE', '15 %');
insert into salesman values (4000, 'SMITH', 'DELHI', '30 %');
insert into salesman values (5000, 'HARSHA', 'HYDERABAD', '15 %');

-- Insert into CUSTOMER1
insert into customer1 values (10, 'PREETHI', 'BANGALORE', 100, 1000);
insert into customer1 values (11, 'VIVEK', 'MANGALORE', 300, 1000);
insert into customer1 values (12, 'BHASKAR', 'CHENNAI', 400, 2000);
insert into customer1 values (13, 'CHETHAN', 'BANGALORE', 200, 2000);
insert into customer1 values (14, 'MAMATHA', 'BANGALORE', 400, 3000);

-- Insert into ORDERS
insert into orders values (50, 5000, '2017-05-04', 10, 1000);
insert into orders values (51, 450, '2017-01-20', 10, 2000);
insert into orders values (52, 1000, '2017-02-24', 13, 2000);
insert into orders values (53, 3500, '2017-04-13', 14, 3000);
insert into orders values (54, 550, '2017-03-09', 12, 2000);
```

Queries

1.Count the customers with grades above Bangalore s average.

```
select grade, count(distinct customer_id) as customer_count
```

```
from customer1
group by grade
having grade > (
   select avg(grade)
   from customer1
   where city = 'BANGALORE'
);
```

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

```
select salesman_id, name
from salesman a
where 1 < (
   select count(*)
   from customer1
   where salesman_id = a.salesman_id
);
```

3.List all salesmen and indicate those who have and don t have customers in their cities (Use UNION operation.)

```
select salesman.salesman_id, name, cust_name, commission
from salesman, customer1
where salesman.city = customer1.city
union
select salesman_id, name, 'no match', commission
from salesman
where city not in (
   select city
   from customer1
)
order by 2 desc;
```

4.Create a view that finds the salesman who has the customer with the highest order of a day.

```
create view elitsalesman as
select b.ord_date, a.salesman_id, a.name
from salesman a, orders b
where a.salesman_id = b.salesman_id
and b.purchase_amt = (
   select max(c.purchase_amt)
   from orders c
   where c.ord_date = b.ord_date
);
```

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

```
delete from salesman
where salesman_id = 1000;

--to view the output:
select * from salesman;
```

C. 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)

Write SQL queries to

1. List the titles of all movies directed by 'Hitchcock'.
2. Find the movie names where one or more actors acted in two or more movies.
3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
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.
5. Update rating of all movies directed by 'Steven Spielberg' to5.

Table Creation:

```
CREATE TABLE ACTOR (
   ACT_ID INT(3) PRIMARY KEY,
   ACT_NAME VARCHAR(20),
   ACT_GENDER CHAR(1)
);

CREATE TABLE DIRECTOR (
   DIR_ID INT(3) PRIMARY KEY,
   DIR_NAME VARCHAR(20),
   DIR_PHONE BIGINT(10)
);

CREATE TABLE MOVIES (
   MOV_ID INT(4) PRIMARY KEY,
```

```
    MOV_TITLE VARCHAR(25),
    MOV_YEAR INT(4),
    MOV_LANG VARCHAR(12),
    DIR_ID INT(3),
    FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR(DIR_ID)
);

CREATE TABLE MOVIE_CAST (
    ACT_ID INT(3),
    MOV_ID INT(4),
    ROLE VARCHAR(10),
    PRIMARY KEY (ACT_ID, MOV_ID),
    FOREIGN KEY (ACT_ID) REFERENCES ACTOR(ACT_ID),
    FOREIGN KEY (MOV_ID) REFERENCES MOVIES(MOV_ID)
);

CREATE TABLE RATING (
    MOV_ID INT(4) PRIMARY KEY,
    REV_STARS VARCHAR(25),
    FOREIGN KEY (MOV_ID) REFERENCES MOVIES(MOV_ID)
);
```

Insertion of Values to Tables:

```
INSERT INTO ACTOR VALUES (301, 'ANUSHKA', 'F');
INSERT INTO ACTOR VALUES (302, 'PRABHAS', 'M');
INSERT INTO ACTOR VALUES (303, 'PUNITH', 'M');
INSERT INTO ACTOR VALUES (304, 'JERMY', 'M');

INSERT INTO DIRECTOR VALUES (60, 'RAJAMOULI', 8751611001);
INSERT INTO DIRECTOR VALUES (61, 'HITCHCOCK', 7766138911);
INSERT INTO DIRECTOR VALUES (62, 'FARAN', 9986776531);
INSERT INTO DIRECTOR VALUES (63, 'STEVEN SPIELBERG', 8989776530);
```

```
INSERT INTO MOVIES VALUES (1001, 'BAHUBALI-2', 2017, 'TELUGU', 60);
INSERT INTO MOVIES VALUES (1002, 'BAHUBALI-1', 2015, 'TELUGU', 60);
INSERT INTO MOVIES VALUES (1003, 'AKASH', 2008, 'KANNADA', 61);
INSERT INTO MOVIES VALUES (1004, 'WAR HORSE', 2011, 'ENGLISH', 63);

INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (301, 1001, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (303, 1003, 'HERO');
INSERT INTO MOVIE_CAST VALUES (303, 1002, 'GUEST');
INSERT INTO MOVIE_CAST VALUES (304, 1004, 'HERO');

INSERT INTO RATING VALUES (1001, '4');
INSERT INTO RATING VALUES (1002, '2');
INSERT INTO RATING VALUES (1003, '5');
INSERT INTO RATING VALUES (1004, '4');
```

Queries:

1. List the titles of all movies directed by 'Hitchcock'.

```
SELECT MOV_TITLE
FROM MOVIES
WHERE DIR_ID IN (
   SELECT DIR_ID
   FROM DIRECTOR
   WHERE DIR_NAME = 'HITCHCOCK'
);
```

select mov_title from movies, director
where director.dir_id=movies.dir_id and dir_name = "HITCHCOCK";

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

```
SELECT M.MOV_TITLE
FROM MOVIES M, MOVIE_CAST MV
WHERE M.MOV_ID = MV.MOV_ID
  AND MV.ACT_ID IN (
     SELECT ACT_ID
```

```
    FROM MOVIE_CAST
    GROUP BY ACT_ID
    HAVING COUNT(ACT_ID) > 1
 )
GROUP BY M.MOV_TITLE
HAVING COUNT(*) > 1;
```

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

```
SELECT A.ACT_NAME, C.MOV_TITLE, C.MOV_YEAR
FROM ACTOR A
JOIN MOVIE_CAST B ON A.ACT_ID = B.ACT_ID
JOIN MOVIES C ON B.MOV_ID = C.MOV_ID
WHERE C.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

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.

```
SELECT M.MOV_TITLE, MAX(R.REV_STARS) AS HIGHEST_STARS
FROM MOVIES M
INNER JOIN RATING R ON M.MOV_ID = R.MOV_ID
GROUP BY M.MOV_TITLE
HAVING MAX(R.REV_STARS) > 0
ORDER BY M.MOV_TITLE;
```

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

```
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'
   )
);
```

To view result: SELECT * FROM RATING;

D. Consider the schema for College Database:

STUDENT (USN, SName, Address, Phone,
Gender) SEMSEC (SSID, Sem, Sec)
CLASS (USN, SSID)
SUBJECT (Subcode, Title, Sem, Credits)
IAMARKS (USN, Subcode, SSID, Test1, Test2, Test3,FinalIA)

Write SQL queries to

1. List all the student details studying in fourth semester 'C' section.
2. Compute the total number of male and female students in each semester and in each section.
3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.
4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.
5. Categorize students based on the following criterion:

   If FinalIA = 17 to 20 then
   CAT ='Outstanding'
   If FinalIA = 12 to 16 then CAT =
   'Average' If FinalIA< 12 then CAT =
   'Weak'

Give these details only for 8th semester A, B, and C section students.

TABLE CREATION:

```
CREATE TABLE STUDENT (
   USN VARCHAR(10) PRIMARY KEY,
   SNAME VARCHAR(25),
   ADDRESS VARCHAR(25),
   PHONE BIGINT(10),
   GENDER CHAR(1)
);

CREATE TABLE SEMSEC (
   SSID VARCHAR(5) PRIMARY KEY,
   SEM INT(2),
   SEC CHAR(1)
);

CREATE TABLE CLASS (
   USN VARCHAR(10),
   SSID VARCHAR(5),
   PRIMARY KEY (USN, SSID),
   FOREIGN KEY (USN) REFERENCES STUDENT(USN),
   FOREIGN KEY (SSID) REFERENCES SEMSEC(SSID)
);

CREATE TABLE SUBJECT (
   SUBCODE VARCHAR(8) PRIMARY KEY,
   TITLE VARCHAR(20),
   SEM INT(2),
   CREDITS INT(2)
);

CREATE TABLE IAMARKS (
   USN VARCHAR(10),
   SUBCODE VARCHAR(8),
   SSID VARCHAR(5),
```

```
    TEST1 INT(2),
    TEST2 INT(2),
    TEST3 INT(2),
    FINALIA INT(2),
    PRIMARY KEY (USN, SUBCODE, SSID),
    FOREIGN KEY (USN) REFERENCES STUDENT(USN),
    FOREIGN KEY (SUBCODE) REFERENCES SUBJECT(SUBCODE),
    FOREIGN KEY (SSID) REFERENCES SEMSEC(SSID)
);
```

INSERTION OF VALUES:

```
INSERT INTO STUDENT VALUES ('1RN13CS020', 'AKSHAY', 'BELAGAVI', 8877881122,
'M');
INSERT INTO STUDENT VALUES ('1RN13CS062', 'SANDHYA', 'BENGALURU',
7722829912, 'F');
INSERT INTO STUDENT VALUES ('1RN13CS091', 'TEESHA', 'BENGALURU', 7712312312,
'F');
INSERT INTO STUDENT VALUES ('1RN13CS066', 'SUPRIYA', 'MANGALURU',
8877881122, 'F');
INSERT INTO STUDENT VALUES ('1RN14CS010', 'ABHAY', 'BENGALURU', 9900211201,
'M');
INSERT INTO STUDENT VALUES ('1RN14CS032', 'BHASKAR', 'BENGALURU',
9923211099, 'M');
INSERT INTO STUDENT VALUES ('1RN14CS025', 'ASMI', 'BENGALURU', 7894737377,
'F');
INSERT INTO STUDENT VALUES ('1RN15CS011', 'AJAY', 'TUMKUR', 9845091341, 'M');
INSERT INTO STUDENT VALUES ('1RN15CS029', 'CHITRA', 'DAVANGERE', 7696772121,
'F');
INSERT INTO STUDENT VALUES ('1RN15CS045', 'JEEVA', 'BELLARY', 9944850121,
'M');
INSERT INTO STUDENT VALUES ('1RN15CS091', 'SANTOSH', 'MANGALURU',
8812332201, 'M');
INSERT INTO STUDENT VALUES ('1RN16CS045', 'ISMAIL', 'KALBURGI', 9900232201,
'M');
```

```
INSERT INTO STUDENT VALUES ('1RN16CS088', 'SAMEERA', 'SHIMOGA', 9905542212,
'F');
INSERT INTO STUDENT VALUES ('1RN16CS122', 'VINAYAKA', 'CHIKAMAGALUR',
8800880011, 'M');

INSERT INTO SEMSEC VALUES ('CSE8A', 8, 'A');
INSERT INTO SEMSEC VALUES ('CSE8B', 8, 'B');
INSERT INTO SEMSEC VALUES ('CSE8C', 8, 'C');
INSERT INTO SEMSEC VALUES ('CSE7A', 7, 'A');
INSERT INTO SEMSEC VALUES ('CSE7B', 7, 'B');
INSERT INTO SEMSEC VALUES ('CSE7C', 7, 'C');
INSERT INTO SEMSEC VALUES ('CSE6A', 6, 'A');
INSERT INTO SEMSEC VALUES ('CSE6B', 6, 'B');
INSERT INTO SEMSEC VALUES ('CSE6C', 6, 'C');
INSERT INTO SEMSEC VALUES ('CSE5A', 5, 'A');
INSERT INTO SEMSEC VALUES ('CSE5B', 5, 'B');
INSERT INTO SEMSEC VALUES ('CSE5C', 5, 'C');
INSERT INTO SEMSEC VALUES ('CSE4A', 4, 'A');
INSERT INTO SEMSEC VALUES ('CSE4B', 4, 'B');
INSERT INTO SEMSEC VALUES ('CSE4C', 4, 'C');
INSERT INTO SEMSEC VALUES ('CSE3A', 3, 'A');
INSERT INTO SEMSEC VALUES ('CSE3B', 3, 'B');
INSERT INTO SEMSEC VALUES ('CSE3C', 3, 'C');
INSERT INTO SEMSEC VALUES ('CSE2A', 2, 'A');
INSERT INTO SEMSEC VALUES ('CSE2B', 2, 'B');
INSERT INTO SEMSEC VALUES ('CSE2C', 2, 'C');
INSERT INTO SEMSEC VALUES ('CSE1A', 1, 'A');
INSERT INTO SEMSEC VALUES ('CSE1B', 1, 'B');
INSERT INTO SEMSEC VALUES ('CSE1C', 1, 'C');

INSERT INTO CLASS VALUES ('1RN13CS020', 'CSE8A');
INSERT INTO CLASS VALUES ('1RN13CS062', 'CSE8A');
INSERT INTO CLASS VALUES ('1RN13CS066', 'CSE8B');
INSERT INTO CLASS VALUES ('1RN13CS091', 'CSE8C');
INSERT INTO CLASS VALUES ('1RN14CS010', 'CSE7A');
```

```
INSERT INTO CLASS VALUES ('1RN14CS025', 'CSE7A');
INSERT INTO CLASS VALUES ('1RN14CS032', 'CSE7A');
INSERT INTO CLASS VALUES ('1RN15CS011', 'CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS029', 'CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS045', 'CSE4B');
INSERT INTO CLASS VALUES ('1RN15CS091', 'CSE4C');
INSERT INTO CLASS VALUES ('1RN16CS045', 'CSE3A');
INSERT INTO CLASS VALUES ('1RN16CS088', 'CSE3B');
INSERT INTO CLASS VALUES ('1RN16CS122', 'CSE3C');

INSERT INTO SUBJECT VALUES ('10CS81', 'ACA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82', 'SSM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83', 'NM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84', 'CC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS85', 'PW', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS71', 'OOAD', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS72', 'ECS', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS73', 'PTW', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS74', 'DWDM', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS75', 'JAVA', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS76', 'SAN', 7, 4);
INSERT INTO SUBJECT VALUES ('15CS51', 'ME', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS52', 'CN', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS53', 'DBMS', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS54', 'ATC', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS55', 'JAVA', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS56', 'AI', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS41', 'M4', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS42', 'SE', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS43', 'DAA', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS44', 'MPMC', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS45', 'OOC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS46', 'DC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS31', 'M3', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS32', 'ADE', 3, 4);
```

```
INSERT INTO SUBJECT VALUES ('15CS33', 'DSA', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS34', 'CO', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS35', 'USP', 3, 3);
INSERT INTO SUBJECT VALUES ('15CS36', 'DMS', 3, 3);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091', '10CS81', 'CSE8C', 15, 16, 18);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091', '10CS82', 'CSE8C', 12, 19, 14);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091', '10CS83', 'CSE8C', 19, 15, 20);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091', '10CS84', 'CSE8C', 20, 16, 19);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091', '10CS85', 'CSE8C', 15, 15, 12);
```

Queries:

1. List all the student details studying in fourth semester 'C 'section.

```
SELECT S.*, SS.SEM, SS.SEC
FROM STUDENT S, SEMSEC SS, CLASS C
WHERE S.USN = C.USN
  AND SS.SSID = C.SSID
  AND SS.SEM = 4
  AND SS.SEC = 'C';
```

2. Compute the total number of male and female students in each semester and in each section.

```
SELECT
   SS.SEM,
   SS.SEC,
```

```
    S.GENDER,
    COUNT(S.GENDER) AS COUNT
FROM
    STUDENT S, SEMSEC SS, CLASS C
WHERE
    S.USN = C.USN
    AND SS.SSID = C.SSID
GROUP BY
    SS.SEM, SS.SEC, S.GENDER
ORDER BY
    SS.SEM;
```

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE VIEW STU_TEST1_MARKS_VIEW AS
SELECT
    SUBCODE,
    TEST1
FROM
    IAMARKS
WHERE
    USN = '1BI15CS101';
```

4. Calculate the Final IA (average of best two test marks) and update the corresponding table for all students.

```
UPDATE IAMARKS
SET FINALIA = (
    (GREATEST(TEST1, TEST2, TEST3) +
     (TEST1 + TEST2 + TEST3 - LEAST(TEST1, TEST2, TEST3) - GREATEST(TEST1,
TEST2, TEST3))
    ) / 2
);
```

to view result:

SELECT USN, SUBCODE, TEST1, TEST2, TEST3, FINALIA FROM IAMARKS;

5. Categorize students based on the followingcriterion:

If FinalIA = 17 to 20 then

CAT =‘Outstanding’

If FinalIA = 12 to 16 then CAT =

‘Average’ If FinalIA< 12 then CAT =

‘Weak’

Give these details only for 8th semester A, B, and C section students.

```
SELECT
   S.USN,
   S.SNAME,
   S.ADDRESS,
   S.PHONE,
   S.GENDER,
   CASE
      WHEN IA.FINALIA BETWEEN 17 AND 20 THEN 'OUTSTANDING'
      WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
      ELSE 'WEAK'
   END AS CAT
FROM
   STUDENT S,
   SEMSEC SS,
   IAMARKS IA,
   SUBJECT SUB
WHERE
   S.USN = IA.USN
   AND SS.SSID = IA.SSID
   AND SUB.SUBCODE = IA.SUBCODE
   AND SUB.SEM = 8
   AND SS.SEC IN ('A', 'B', 'C')
```

E. Consider the schema for CompanyDatabase:

EMPLOYEE (SSN, Name, Address, Sex, Salary, SuperSSN,
DNo) DEPARTMENT (DNo, DName, MgrSSN, MgrStartDate)
DLOCATION (DNo,DLoc)
PROJECT (PNo, PName, PLocation,
DNo) WORKS_ON (SSN, PNo, Hours)

Table creation:

```
CREATE TABLE DEPARTMENT (
   DNO VARCHAR(20) PRIMARY KEY,
   DNAME VARCHAR(20),
   MGRSTARTDATE DATE
);

CREATE TABLE EMPLOYEE (
   SSN VARCHAR(20) PRIMARY KEY,
   FNAME VARCHAR(20),
   LNAME VARCHAR(20),
   ADDRESS VARCHAR(50),
   SEX CHAR(1),
   SALARY INT,
   SUPERSSN VARCHAR(20),
   DNO VARCHAR(20),
   FOREIGN KEY (SUPERSSN) REFERENCES EMPLOYEE(SSN),
   FOREIGN KEY (DNO) REFERENCES DEPARTMENT(DNO)
);

ALTER TABLE DEPARTMENT
ADD MGRSSN VARCHAR(20),
ADD FOREIGN KEY (MGRSSN) REFERENCES EMPLOYEE(SSN);

CREATE TABLE DLOCATION (
```

```
    DLOC VARCHAR(20),
    DNO VARCHAR(20),
    PRIMARY KEY (DNO, DLOC),
    FOREIGN KEY (DNO) REFERENCES DEPARTMENT(DNO)
);

CREATE TABLE PROJECT (
    PNO INT PRIMARY KEY,
    PNAME VARCHAR(20),
    PLOCATION VARCHAR(20),
    DNO VARCHAR(20),
    FOREIGN KEY (DNO) REFERENCES DEPARTMENT(DNO)
);

CREATE TABLE WORKS_ON (
    HOURS INT,
    SSN VARCHAR(20),
    PNO INT,
    PRIMARY KEY (SSN, PNO),
    FOREIGN KEY (SSN) REFERENCES EMPLOYEE(SSN),
    FOREIGN KEY (PNO) REFERENCES PROJECT(PNO)
);
```

Insertion of values:

```
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSECE01','JOHN','SCOTT','BANGALORE','M',450000),
('RNSCSE01','JAMES','SMITH','BANGALORE','M',500000),
('RNSCSE02','HEARN','BAKER','BANGALORE','M',700000),
('RNSCSE03','EDWARD','SCOTT','MYSORE','M',500000),
('RNSCSE04','PAVAN','HEGDE','MANGALORE','M',650000),
('RNSCSE05','GIRISH','MALYA','MYSORE','M',450000),
('RNSCSE06','NEHA','SN','BANGALORE','F',800000),
('RNSACC01','AHANA','K','MANGALORE','F',350000),
('RNSACC02','SANTHOSH','KUMAR','MANGALORE','M',300000),
```

('RNSISE01','VEENA','M','MYSORE','F',600000),
('RNSIT01','NAGESH','HR','BANGALORE','M',500000);

INSERT INTO DEPARTMENT (DNO, DNAME, MGRSTARTDATE, MGRSSN) VALUES
('1','ACCOUNTS','2001-01-01','RNSACC02'),
('2','IT','2016-08-01','RNSIT01'),
('3','ECE','2008-06-01','RNSECE01'),
('4','ISE','2015-08-01','RNSISE01'),
('5','CSE','2002-06-01','RNSCSE05');

UPDATE EMPLOYEE SET SUPERSSN=NULL, DNO='3' WHERE SSN='RNSECE01';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE02', DNO='5' WHERE SSN='RNSCSE01';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE03', DNO='5' WHERE SSN='RNSCSE02';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE04', DNO='5' WHERE SSN='RNSCSE03';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN='RNSCSE05' WHERE SSN='RNSCSE04';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN='RNSCSE06' WHERE SSN='RNSCSE05';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN=NULL WHERE SSN='RNSCSE06';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN='RNSACC02' WHERE SSN='RNSACC01';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN=NULL WHERE SSN='RNSACC02';
UPDATE EMPLOYEE SET DNO='4', SUPERSSN=NULL WHERE SSN='RNSISE01';
UPDATE EMPLOYEE SET DNO='2', SUPERSSN=NULL WHERE SSN='RNSIT01';

INSERT INTO DLOCATION VALUES ('BANGALORE','1');
INSERT INTO DLOCATION VALUES ('BANGALORE','2');
INSERT INTO DLOCATION VALUES ('BANGALORE','3');
INSERT INTO DLOCATION VALUES ('MANGALORE','4');
INSERT INTO DLOCATION VALUES ('MANGALORE','5');

INSERT INTO PROJECT VALUES (100,'IOT','BANGALORE','5');
INSERT INTO PROJECT VALUES (101,'CLOUD','BANGALORE','5');
INSERT INTO PROJECT VALUES (102,'BIGDATA','BANGALORE','5');
INSERT INTO PROJECT VALUES (103,'SENSORS','BANGALORE','3');
INSERT INTO PROJECT VALUES (104,'BANK MANAGEMENT','BANGALORE','1');

```
INSERT INTO PROJECT VALUES (105,'SALARY MANAGEMENT','BANGALORE','1');
INSERT INTO PROJECT VALUES (106,'OPENSTACK','BANGALORE','4');
INSERT INTO PROJECT VALUES (107,'SMARTCITY','BANGALORE','2');

INSERT INTO WORKS_ON VALUES (4,'RNSCSE01',100);
INSERT INTO WORKS_ON VALUES (6,'RNSCSE01',101);
INSERT INTO WORKS_ON VALUES (8,'RNSCSE01',102);
INSERT INTO WORKS_ON VALUES (10,'RNSCSE02',100);
INSERT INTO WORKS_ON VALUES (3,'RNSCSE04',100);
INSERT INTO WORKS_ON VALUES (4,'RNSCSE05',101);
INSERT INTO WORKS_ON VALUES (5,'RNSCSE06',102);
INSERT INTO WORKS_ON VALUES (6,'RNSCSE03',102);
INSERT INTO WORKS_ON VALUES (7,'RNSECE01',103);
INSERT INTO WORKS_ON VALUES (5,'RNSACC01',104);
INSERT INTO WORKS_ON VALUES (6,'RNSACC02',105);
INSERT INTO WORKS_ON VALUES (4,'RNSISE01',106);
INSERT INTO WORKS_ON VALUES (10,'RNSIT01',107);
```

Queries:

1. Make a list of all project numbers for projects that involve an employee whose last name is ‘Scott’, either as a worker or as a manager of the department that controls the project.

```
(
   SELECT DISTINCT P.PNO
   FROM PROJECT AS P
   JOIN DEPARTMENT AS D ON P.DNO = D.DNO
   JOIN EMPLOYEE AS E ON D.MGRSSN = E.SSN
   WHERE E.LNAME = 'SCOTT'
)
UNION
(
   SELECT DISTINCT P1.PNO
   FROM PROJECT AS P1
   JOIN WORKS_ON AS W ON P1.PNO = W.PNO
```

```
    JOIN EMPLOYEE AS E1 ON W.SSN = E1.SSN
    WHERE E1.LNAME = 'SCOTT'
);
```

2. Show the resulting salaries if every employee working on the ‘IoT’ project is given a 10 percentraise.

```
SELECT DISTINCT
    E.FNAME,
    E.LNAME,
    1.1 * E.SALARY AS INCR_SAL
FROM
    EMPLOYEE E,
    WORKS_ON W,
    PROJECT P
WHERE
    E.SSN = W.SSN
    AND W.PNO = P.PNO
    AND P.PNAME = 'IOT';
```

3. Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as the maximum salary, the minimum salary, and the average salary in this department

```
SELECT
    SUM(E.SALARY),
    MAX(E.SALARY),
    MIN(E.SALARY),
    AVG(E.SALARY)
FROM EMPLOYEE E, DEPARTMENT D
WHERE E.DNO = D.DNO
AND D.DNAME = 'ACCOUNTS';
```

4.Retrieve the name of each employee who works on all the projects Controlled by department number 5 (use NOT EXISTSoperator).

```
SELECT E.FNAME, E.LNAME
FROM EMPLOYEE E
WHERE NOT EXISTS (
   SELECT P.PNO
   FROM PROJECT P
   WHERE P.DNO = '5'
   AND P.PNO NOT IN (
      SELECT W.PNO
      FROM WORKS_ON W
      WHERE W.SSN = E.SSN
   )
);
```

5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.

```
SELECT D.DNO, COUNT(*)
FROM DEPARTMENT D, EMPLOYEE E
WHERE D.DNO = E.DNO
AND E.SALARY > 600000
AND D.DNO IN (
   SELECT E1.DNO
   FROM EMPLOYEE E1
   GROUP BY E1.DNO
   HAVING COUNT(*) > 5
)
GROUP BY D.DNO;
```