1) Consider the following schema for a Library Database:
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)

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.
2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017
3. Delete a book in the BOOK table. Update the contents of other tables to reflect this data manipulation operation.
4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.
5. Create a view of all books and its number of copies that are currently available in the Library.

Query to create tables :->

TABLE PUBLISHER

```
CREATE TABLE PUBLISHER(
PUB_ID INT PRIMARY KEY,
PUB_NAME VARCHAR(20) UNIQUE,
ADDRESS VARCHAR(20),
PHONE INT);
```

TABLE BOOK

```
CREATE TABLE BOOK(
BOOK_ID INT PRIMARY KEY,
TITLE VARCHAR(20),
PUBLISHER_NAME VARCHAR(20) REFERENCES PUBLISHER(NAME)ON
DELETE CASCADE,
PUB_YEAR INT );
```

LIBRARY_BRANCH

```
CREATE TABLE LIBRARY_BRANCH(
BRANCH_ID INT PRIMARY KEY,
BRANCH_NAME VARCHAR(20),
ADDRESS VARCHAR(30));
```

BOOK_AUTHORS
CREATE TABLE BOOK_AUTHORS(
BOOK_ID INT,
AUTHOR_NAME VARCHAR(20),
PRIMARY KEY(BOOK_ID, AUTHOR_NAME),
FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE);

BOOK_COPIES
CREATE TABLE BOOK_COPIES(
BOOK_ID INT, BRANCH_ID INT, NO_OF_COPIES 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);

BOOK_LENDING
CREATE TABLE BOOK_LENDING(
BOOK_ID INT,BRANCH_ID INT,
CARD_NO INT,DATE_OUT DATE,
DUE_DATE DATE, 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);

TABLE DATA ENTRY
TABLE BOOK
INSERT INTO BOOK VALUES (001, ‘MCGRAW-HILL’, ‘GANGA’, 2001);
INSERT INTO BOOK VALUES (002, ‘MY ARTEMIS’, ‘KVS’, 2004);
INSERT INTO BOOK VALUES (003, ‘CHEMISTRY VOL 1’, ‘WESTLAND’, 2006);
INSERT INTO BOOK VALUES (004, ‘UPRISING’, ‘RUPA’, 2018);
INSERT INTO BOOK VALUES (005, ‘CHEMISTRY VOL 2’, ‘WESTLAND’, 2021);

book_id	title	publisher_name	pub_year
1	MCGRAW-HILL	GANGA	2001
2	MY ARTEMIS	KVS	2004
3	CHEMISTRY VOL 1	WESTLAND	2006
4	UPRISING	RUPA	2018
5	CHEMISTRY VOL 2	WESTLAND	2021

(5 rows)

TABLE PUBLISHER

INSERT INTO PUBLISHER VALUES (501, ‘KVS’, ‘BANGALORE’, 9535616745);
INSERT INTO PUBLISHER VALUES (502, ‘WESTLAND’, PUNE’, 8768916745);
INSERT INTO PUBLISHER VALUES (503, ‘RUPA’, ‘BANGALORE’, 6478989715);
INSERT INTO PUBLISHER VALUES (504, ‘GANGA’, ‘MUMBAI’, 9876985645);
INSERT INTO PUBLISHER VALUES (505, ‘HACHETTE’, ‘MATTUR’, 7013458745);

```
 pub_id | pub_name |  address  |   phone
--------+----------+-----------+------------
    501 | KVS      | BANGALORE | 9535616745
    502 | WESTLAND | PUNE      | 8768916745
    503 | RUPA     | BANGALORE | 6478989715
    504 | GANGA    | MUMBAI    | 9876985645
    505 | HACHETTE | MATTUR    | 7013458745
    (5 rows)
```

TABLE LIBRARY_BRANCH

INSERT INTO LIBRARY_BRANCH VALUES (101, ‘BOOK AXIS', ‘BANGALORE’);
INSERT INTO LIBRARY_BRANCH VALUES (102, ‘BOOK SQUARE', ‘PUNE’);
INSERT INTO LIBRARY_BRANCH VALUES (103, ‘CLAUS BOOKS', ‘MUMBAI’);
INSERT INTO LIBRARY_BRANCH VALUES (104, ‘COMIC CON', ‘PUNE’);
INSERT INTO LIBRARY_BRANCH VALUES (105, ‘FANDOM', ‘BANGALORE’);

```
 branch_id | branch_name |  address
-----------+-------------+-----------
       101 | BOOK AXIS   | BANGALORE
       102 | BOOK SQUARE | PUNE
       103 | CLAUS BOOKS | MUMBAI
       104 | COMIC CON   | PUNE
       105 | FANDOM      | BANGALORE
 (5 rows)
```

TABLE BOOK_AUTHORS
INSERT INTO BOOK_AUTHORS VALUES (001, ‘ASHISH C’);
INSERT INTO BOOK_AUTHORS VALUES (002, ‘ANEESHA’);
INSERT INTO BOOK_AUTHORS VALUES (003, ‘ADITYA KUL C’);
INSERT INTO BOOK_AUTHORS VALUES (004, ‘SAQUIB M’);
INSERT INTO BOOK_AUTHORS VALUES (005, ‘ARJUN S’);

```
 book_id | author_name
---------+--------------
       1 | ASHISH C
       2 | ANEESHA
       3 | ADITYA KUL C
       4 | SAQUIB M
       5 | ARJUN S
(5 rows)
```

TABLE BOOK_COPIES

```
INSERT INTO BOOK_COPIES VALUES (001, 102, 40);
INSERT INTO BOOK_COPIES VALUES (002, 101, 18);
INSERT INTO BOOK_COPIES VALUES (003, 104, 53);
INSERT INTO BOOK_COPIES VALUES (004, 103, 4);
INSERT INTO BOOK_COPIES VALUES (005, 105, 20);
```

```
 book_id | branch_id | no_of_copies
---------+-----------+--------------
       1 |       102 |           40
       2 |       101 |           18
       3 |       104 |           53
       4 |       103 |            4
       5 |       105 |           20
(5 rows)
```

TABLE BOOK_LENDING

```
INSERT INTO BOOK_LENDING VALUES (001, 101, 5001, '21-SEP-2021', '19-OCT-2021');
INSERT INTO BOOK_LENDING VALUES (002, 102, 5002, '07-JAN-2017', '18-MAY-2017');
INSERT INTO BOOK_LENDING VALUES (003, 102, 5002, '02-FEB-2017', '22-MAR-2020');
INSERT INTO BOOK_LENDING VALUES (004, 102, 5002, '14-MAR-2017', '08-MAY-2019');
INSERT INTO BOOK_LENDING VALUES (005, 104, 5005, '18-JUN-2020', '14-AUG-2021');
```

```
 book_id | branch_id | card_no |  date_out  |  due_date
---------+-----------+---------+------------+------------
       1 |       101 |    5001 | 2021-09-21 | 2021-10-19
       1 |       102 |    5002 | 2017-01-07 | 2017-05-18
       2 |       102 |    5003 | 2017-02-02 | 2020-03-22
       3 |       103 |    5004 | 2016-09-14 | 2021-10-08
```

```
      5 |      104 |    5005 | 2020-06-18 | 2021-08-14
      2 |      102 |    5002 | 2017-01-07 | 2017-05-18
      3 |      102 |    5002 | 2017-02-02 | 2020-03-22
      4 |      102 |    5002 | 2017-03-14 | 2019-05-08
(8 rows)
```

QUERIES:->

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

```
SELECT LB.BRANCH_NAME, B.BOOK_ID,TITLE,
PUBLISHER_NAME,AUTHOR_NAME, NO_OF_COPIES FROM BOOK B, BOOK_AUTHORS
BA, BOOK_COPIES BC, LIBRARY_BRANCH LB
WHERE B.BOOK_ID = BA.BOOK_ID AND BA.BOOK_ID = BC.BOOK_ID AND
BC.BRANCH_ID = LB.BRANCH_ID GROUP BY LB.BRANCH_NAME, B.BOOK_ID, TITLE,
PUBLISHER_NAME, AUTHOR_NAME, NO_OF_COPIES;
```

```
 branch_name | book_id |  title  | publisher_name | author_name  | no_of_copies
-------------+---------+-----------------+----------------+--------------+--------------
 COMIC CON   |       3 | CHEMISTRY VOL 1 | WESTLAND       | ADITYA KUL C |53
 BOOK SQUARE |       1 | MCGRAW-HILL     | GANGA          | ASHISH C     | 40
 FANDOM      |       5 | CHEMISTRY VOL 2 | WESTLAND       | ARJUN S      | 20
 CLAUS BOOKS |       4 | UPRISING        | RUPA           | SAQUIB M     | 4
 BOOK AXIS   |       2 | MY ARTEMIS      | KVS            | ANEESHA      | 18
 (5 rows)
```

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

```
SELECT CARD_NO FROM BOOK_LENDING
WHERE DATE_OUT > '01-JAN-2017' AND
DATE_OUT < '01-JUN-2017'
GROUP BY CARD_NO
HAVING COUNT (*) > 3;
 card_no
---------
    5002
(1 row)
```

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

```
DELETE FROM BOOK  WHERE BOOK_ID = 001;
                book_id |      title      | publisher_name | pub_year
               -------+-----------------+----------------+----------
                            2 | MY ARTEMIS      | KVS          |     2004
                            3 | CHEMISTRY VOL 1 | WESTLAND     | 2006
                            4 | UPRISING        | RUPA         |     2018
                            5 | CHEMISTRY VOL 2 | WESTLAND     | 2021
(4 rows)
```

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;
```

To view it: SELECT * FROM V_PUBLICATION;

```
 pub_year
----------
    2001
    2004
    2006
    2018
    2021
(5 rows)
```

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

```
CREATE VIEW BOOKS_AVAILABLE AS
SELECT B.BOOK_ID, B.TITLE, C.NO_OF_COPIES
FROM LIBRARY_BRANCH L, BOOK B, BOOK_COPIES C
WHERE B.BOOK_ID = C.BOOK_ID AND
L.BRANCH_ID=C.BRANCH_ID;
                book_id |      title      | no_of_copies
---------+-----------------+--------------
      1 | MCGRAW-HILL     |          40
      2 | MY ARTEMIS      |          18
      3 | CHEMISTRY VOL 1 |          53
                        4 | UPRISING        |           4
                        5 | CHEMISTRY VOL 2 |          20
(5 rows)
```

2) Consider the following schema for Order Database:

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:

1. Count the customers with grades above Bangalore's average.
2. Find the name and numbers of all salesmen who had more than one customer.
3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)
4. Create a view that finds the salesman who has the customer with the highest order of a day.
5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

Table creation

Salesman

```
CREATE TABLE SALESMAN(
SALESMAN_ID INT PRIMARY KEY,
NAME CHAR(20) NOT NULL,
CITY CHAR(20) NOT NULL,
COMMISSION NUMERIC(10,2));
```

Customer

```
CREATE TABLE CUSTOMER (
CUSTOMER_ID INT PRIMARY KEY,
CUSTOMER_NAME CHAR(20),
CUST_CITY CHAR(20), GRADE INT, SALESMAN_ID INT,
FOREIGN KEY(SALESMAN_ID) REFERENCES SALESMAN (SALESMAN_ID)ON DELETE
CASCADE);
```

Orders

```
CREATE TABLE ORDERS (
ORDER_NO INT PRIMARY KEY,
PURCHASE_AMT DECIMAL(10,2),
ORDER_DATE DATE,CUSTOMER_ID INT,
SALESMAN_ID INT,
```

```
FOREIGN KEY(SALESMAN_ID) REFERENCES SALESMAN (SALESMAN_ID) ON DELETE
CASCADE,
FOREIGN KEY(CUSTOMER_ID) REFERENCES CUSTOMER (CUSTOMER_ID) ON DELETE
CASCADE);
```

TABLE DATA ENTRY

SALESMAN TABLE

```
INSERT INTO SALESMAN VALUES (1000, 'VARSHA', 'BANGALORE', 8000 );
INSERT INTO SALESMAN VALUES (1001, 'MOHIT', 'PUNE', 6500 );
INSERT INTO SALESMAN VALUES (1002, 'RUBIN', 'CHENNAI', 10500 );
INSERT INTO SALESMAN VALUES (1003, 'SAQUIB', 'BANGALORE', 12000 );
INSERT INTO SALESMAN VALUES (1004, 'KARTHIK', 'BANGALORE', 18000 );
```

```
 salesman_id |      name       |       city       | commission
-------------+-----------------+------------------+------------
        1000 | VARSHA          | BANGALORE        |    8000.00
        1001 | MOHIT           | PUNE             |    6500.00
        1002 | RUBIN           | CHENNAI          |   10500.00
        1003 | SAQUIB          | BANGALORE        |   12000.00
        1004 | KARTHIK         | BANGALORE        |   18000.00
(5 rows)
```

CUSTOMER TABLE

```
INSERT INTO CUSTOMER VALUES (1, 'INFOSYS', 'BANGALORE', 2, 1000);
INSERT INTO CUSTOMER VALUES (2, 'WIPRO', 'PUNE', 4, 1000);
INSERT INTO CUSTOMER VALUES (3, 'ACCENTURE', 'CHENNAI', 3, 1002);
INSERT INTO CUSTOMER VALUES (4, 'GOLDMAN SACHS', 'BANGALORE', 2, 1003);
INSERT INTO CUSTOMER VALUES (5, 'JP MORGAN', 'BANGALORE', 4, 1002);
```

```
 customer_id |  customer_name     |cust_city | grade | salesman_id
-------------+--------------------+----------+-------+-------------
           2 | WIPRO              | PUNE     |     4 |        1000
           3 | ACCENTURE          | CHENNAI  |     4 |        1002
           4 | GOLDMAN SACHS      | BANGALORE|     4 |        1003
           5 | JP MORGAN          | BANGALORE|     4 |        1002
           1 | INFOSYS            | BANGALORE|     2 |        1000
(5 rows)
```

ORDERS TABLE

```
INSERT INTO ORDERS VALUES (501, 50000, '21-OCT-2021', 1, 1000);
INSERT INTO ORDERS VALUES (502, 1000, '05-JUN-2020', 2, 1002);
INSERT INTO ORDERS VALUES (503, 300000, '05-JUN-2020', 2, 1003);
INSERT INTO ORDERS VALUES (504, 10000, '16-SEP-2019', 3, 1000);
INSERT INTO ORDERS VALUES (505, 90000, '12-MAR-2022', 5, 1004);
```

```
 order_no | purchase_amt | order_date | customer_id | salesman_id
----------+--------------+------------+-------------+-------------
      501 |     50000.00 | 2021-10-21 |           1 |        1000
      502 |      1000.00 | 2020-06-05 |           2 |        1002
      503 |    300000.00 | 2020-06-05 |           2 |        1003
      504 |     10000.00 | 2019-09-16 |           3 |        1000
      505 |     90000.00 | 2022-03-12 |           5 |        1004
     (5 rows)
```

QUERIES:->

Count the customers with grades above Bangalore's average.

```
SELECT COUNT (CUSTOMER_ID)
FROM CUSTOMER WHERE GRADE >
(SELECT AVG(GRADE) FROM CUSTOMER
WHERE CUST_CITY = 'BANGALORE');
```

```
 count
-------
     4
(1 row)
```

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

```
SELECT NAME, COUNT (CUSTOMER_ID)
FROM SALESMAN S, CUSTOMER C
WHERE S.SALESMAN_ID = C.SALESMAN_ID
GROUP BY NAME
HAVING COUNT (CUSTOMER_ID) > 1;
```

```
  name         |           count
---------------------+-------
 VARSHA              |     2
 RUBIN               |     2
 (2 rows)
```

List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

```
SELECT NAME
FROM SALESMAN S, CUSTOMER C
WHERE S.SALESMAN_ID = C.SALESMAN_ID
AND S.CITY = C.CITY
UNION
SELECT NAME
FROM SALESMAN
WHERE SALESMAN_ID NOT IN
( SELECT S1.SALESMAN_ID FROM SALESMAN S1, CUSTOMER C1
WHERE S1.SALESMAN_ID = C.SALESMAN_ID
AND S1.CITY = C1.CITY);
```

```
      name
----------------------
 KARTHIK
 SAQUIB
 VARSHA
 MOHIT
 RUBIN
(5 rows)

      name
----------------------
 VARSHA
 SAQUIB
 RUBIN
(3 rows)
```

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

```
CREATE VIEW SALES_ORDER AS
SELECT SALESMAN_ID, PURCHASE_AMT
FROM ORDERS WHERE PURCHASE_AMT = (
SELECT PURCHASE_AMT FROM ORDERS
WHERE ORDER_DATE = '05-JUN-2020' );
```

```
 salesman_id | purchase_amt
-------------+--------------
      1003 |       300000.00
(1 row)
```

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

```
DELETE FROM SALESMAN
WHERE SALESMAN_ID = 1000;
```

```
 salesman_id |        name        |        city        | commission
-------------+--------------------+--------------------+------------
       1001 | MOHIT              | PUNE               |    6500.00
       1002 | RUBIN              | CHENNAI            |   10500.00
       1003 | SAQUIB             | BANGALORE          |   12000.00
       1004 | KARTHIK            | BANGALORE          |   18000.00
(4 rows)
```

3) 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 'DWARKESH'.
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 2005 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 'LANKESH' to 3.

TABLES CREATION

TABLE ACTOR
```
CREATE TABLE ACTOR (
ACT_ID INT PRIMARY KEY,
ACT_NAME CHAR(20) NOT NULL,
ACT_GENDER CHAR(10));
```

TABLE DIRECTOR
```
CREATE TABLE DIRECTOR (
DIR_ID INT PRIMARY KEY,
DIR_NAME CHAR (20),
DIR_PHONE DECIMAL(10,0));
```

TABLE MOVIES
```
CREATE TABLE MOVIES (
MOV_ID INT PRIMARY KEY,
MOV_TITLE VARCHAR(20),
MOV_YEAR INT,MOV_LANG CHAR(20),
DIR_ID INT,
FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR (DIR_ID)
ON DELETE CASCADE );
```

TABLE MOVIE_CAST

```
CREATE TABLE MOVIE_CAST (
ACT_ID INT, MOV_ID INT,
ROLE CHAR(20),
PRIMARY KEY(ACT_ID, MOV_ID),
FOREIGN KEY(ACT_ID) REFERENCES ACTOR (ACT_ID)
ON DELETE CASCADE,
FOREIGN KEY(MOV_ID) REFERENCES MOVIES (MOV_ID)
ON DELETE CASCADE );
```

TABLE RATING

```
CREATE TABLE RATING (
MOV_ID INT, REV_STARS DECIMAL(5,2),
FOREIGN KEY(MOV_ID) REFERENCES MOVIES (MOV_ID)
ON DELETE CASCADE );
```

TABLE DATA ENTRY

TABLE ACTOR

```
INSERT INTO ACTOR VALUES (01, 'SAQUIB', 'MALE');
INSERT INTO ACTOR VALUES (02, 'KARTHIK', 'MALE');
INSERT INTO ACTOR VALUES (03, 'ADITYA', 'MALE');
INSERT INTO ACTOR VALUES (04, 'VARSHA', 'FEMALE');
INSERT INTO ACTOR VALUES (05, 'RUPA', 'FEMALE');
```

```
 act_id |      act_name       | act_gender
--------+---------------------+------------
      1 | SAQUIB              | MALE
      2 | KARTHIK             | MALE
      3 | ADITYA              | MALE
      4 | VARSHA              | FEMALE
      5 | RUPA                | FEMALE
(5 rows)
```

TABLE DIRECTOR

```
INSERT INTO DIRECTOR VALUES (101, 'SANTOSH', 9684769487);
INSERT INTO DIRECTOR VALUES (102, 'DWARKESH', 9675869487);
INSERT INTO DIRECTOR VALUES (103, 'LANKESH', 7898768909);
INSERT INTO DIRECTOR VALUES (104, 'ASHOK', 9789784206);
INSERT INTO DIRECTOR VALUES (105, 'ASHISH', 9789679780);

 dir_id |      dir_name       | dir_phone
--------+---------------------+------------
    101 | SANTOSH             | 9684769487
    102 | DWARKESH            | 9675869487
    103 | LANKESH             | 7898768909
    105 | ASHISH              | 9789679780
    104 | ASHOK               | 9789784206
   (5 rows)
```

TABLE MOVIES

```
INSERT INTO MOVIES VALUES (1001, 'JAMES BOND', 2002, 'ENGLISH', 101);
INSERT INTO MOVIES VALUES (1002, 'SPONGEBOB', 2017, 'ENGLISH', 102);
INSERT INTO MOVIES VALUES (1003, 'THANGI GAGI', 2003, 'KANNADA', 102);
INSERT INTO MOVIES VALUES (1004, 'BIGIL', 2016, 'TAMIL', 103);
INSERT INTO MOVIES VALUES (1005, 'CARS 2', 2019, 'ENGLISH', 103);

 mov_id |  mov_title  | mov_year |      mov_lang       | dir_id
--------+-------------+----------+---------------------+--------
   1001 | JAMES BOND  |     2002 | ENGLISH             |    101
   1002 | SPONGEBOB   |     2017 | ENGLISH             |    102
   1003 | THANGI GAGI |     2003 | KANNADA             |    102
   1004 | BIGIL       |     2016 | TAMIL               |    103
   1005 | CARS 2      |     2019 | ENGLISH             |    103
(5 rows)
```

TABLE MOVIE_CAST

```
INSERT INTO MOVIE_CAST VALUES (1, 1001, 'HERO');
INSERT INTO MOVIE_CAST VALUES (1, 1002, 'PATRICK');
INSERT INTO MOVIE_CAST VALUES (2, 1004, 'SIDE KICK');
INSERT INTO MOVIE_CAST VALUES (2, 1001, 'AGENT');
INSERT INTO MOVIE_CAST VALUES (3, 1005, 'MATER');
```

```
 act_id | mov_id |        role
--------+--------+---------------------
      1 |   1001 | HERO
      1 |   1002 | PATRICK
      2 |   1004 | SIDE KICK
      2 |   1001 | AGENT
      3 |   1005 | MATER
(5 rows)
```

TABLE RATING

```
INSERT INTO RATING VALUES (1001, 5);
INSERT INTO RATING VALUES (1002, 9);
INSERT INTO RATING VALUES (1003, 8);
INSERT INTO RATING VALUES (1004, 6);
INSERT INTO RATING VALUES (1005, 9);
```

```
 mov_id | rev_stars
--------+-----------
   1001 |      5.00
   1002 |      9.00
   1003 |      8.00
   1004 |      6.00
   1005 |      9.00
(5 rows)
```

QUERIES:->

List the titles of all movies directed by 'DWARKESH'.

```
SELECT MOV_TITLE
FROM MOVIES M, DIRECTOR D
WHERE D.DIR_ID=M.DIR_ID AND
DIR_NAME=' DWARKESH ';
```

```
  mov_title
-------------
 SPONGEBOB
 THANGI GAGI
 (2 rows)
```

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

```
SELECT MOV_TITLE
FROM MOVIES M, MOVIE_CAST MC
WHERE M.MOV_ID=MC.MOV_ID AND
MC.ACT_ID IN (SELECT ACT_ID FROM MOVIE_CAST
GROUP BY ACT_ID HAVING COUNT(MOV_ID)>=2);
```

```
 mov_title
------------
 JAMES BOND
 SPONGEBOB
 BIGIL
 JAMES BOND
(4 rows)
```

List all actors who acted in a movie before 2005 and also in a movie after 2015 (use JOIN operation).

```
SELECT ACT_NAME FROM ACTOR A
INNER JOIN MOVIE_CAST MC ON MC.ACT_ID = A.ACT_ID
INNER JOIN MOVIES M ON MC.MOV_ID = M.MOV_ID
WHERE M.MOV_YEAR < 2005 AND ACT_NAME IN
(SELECT ACT_NAME FROM ACTOR A
INNER JOIN MOVIE_CAST MC ON MC.ACT_ID = A.ACT_ID
INNER JOIN MOVIES M ON MC.MOV_ID = M.MOV_ID
WHERE M.MOV_YEAR > 2015 );
```

```
   act_name
----------------------
 SAQUIB
 KARTHIK
(2 rows)
```

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 MOV_TITLE, REV_STARS
FROM MOVIES M, RATING R
WHERE M.MOV_ID=R.MOV_ID AND
REV_STARS >= 1
ORDER BY MOV_TITLE;
```

```
 mov_title  | rev_stars
------------+-----------
        BIGIL       |      6.00
 CARS 2      |      9.00
 JAMES BOND  |      5.00
        SPONGEBOB   |      9.00
 THANGI GAGI |      8.00
 (5 rows)
```

Update rating of all movies directed by 'LANKESH' to 3.

```
UPDATE RATING
SET REV_STARS = 3
WHERE MOV_ID IN (
SELECT M.MOV_ID
FROM MOVIES M, DIRECTOR D
WHERE M.DIR_ID = D.DIR_ID
AND D.DIR_NAME = 'LANKESH' );
```

```
 mov_id | rev_stars
--------+-----------
   1001 |      5.00
   1002 |      9.00
   1003 |      8.00
   1004 |      3.00
   1005 |      3.00
   (5 rows)
```

4. Consider the schema for college database

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

Write SQL QUERIES to :
i. List all the student details studying in fourth semester 'C' section.
ii. Compute the total number of male and female students in each semester and in each section.
iii. Create a view of Test1 marks of student with USN '1DS18IS101' in all Courses.
iv. Calculate the FinalIA (average of best two test marks) and update the Corresponding table for all students.
v. 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

STUDENT TABLE

```
CREATE TABLE STUDENT (
USN VARCHAR(15) PRIMARY KEY,
SNAME CHAR(20),
ADDRESS VARCHAR(50),
PHONE DECIMAL(10,0),
GENDER CHAR(10) );
```

SEMSEC TABLE

```
CREATE TABLE SEMSEC (
SSID VARCHAR(20) PRIMARY KEY,
SEM INT, SEC CHAR(2));
```

CLASS TABLE

```
CREATE TABLE CLASS (
USN VARCHAR(20), SSID VARCHAR(20),
FOREIGN KEY (USN) REFERENCES STUDENT (USN)
ON DELETE CASCADE,
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID)
ON DELETE CASCADE );
```

COURSE TABLE

```
CREATE TABLE COURSE (
SUBCODE INT PRIMARY KEY,
TITLE CHAR (20),SEM INT,
CREDITS INT );
```

IAMARKS TABLE

```
CREATE TABLE IAMARKS (
USN VARCHAR(20),
SUBCODE INT, SSID VARCHAR(20),
TEST1 INT, TEST2 INT, TEST3 INT,
FINALIA DECIMAL(5,2),
FOREIGN KEY (USN) REFERENCES STUDENT(USN) ON DELETE CASCADE );
```

TABLE DATA ENTRY

TABLE STUDENT

```
INSERT INTO STUDENT VALUES ('1DS19IS049', 'ARJUN', 'BANGALORE', 9535616756,
'MALE');
INSERT INTO STUDENT VALUES ('1DS19IS022', 'ADITYA', 'BANGALORE', 7896716657,
'MALE');
INSERT INTO STUDENT VALUES ('1DS19IS017', 'ANEESH', 'MATTUR', 7898676647,
'MALE');
INSERT INTO STUDENT VALUES ('1DS19IS058', 'VARSHA', 'PUNE', 8978916756, 'FEMALE');
INSERT INTO STUDENT VALUES ('1DS19IS007', 'RUPA', 'MANGALORE', 9535616756,
'FEMALE');
```

```
   usn     |        sname         | address |   phone    |   gender
-----------+----------------------+-----------+------------+------------
 1DS19IS049 | ARJUN               | BANGALORE | 9535616756 | MALE
 1DS19IS022 | ADITYA              | BANGALORE | 7896716657 | MALE
 1DS19IS017 | ANEESH              | MATTUR    | 7898676647 | MALE
 1DS19IS058 | VARSHA              | PUNE      | 8978916756 | FEMALE
 1DS19IS007 | RUPA                | MANGALORE | 9535616756 | FEMALE
(5 rows)
```

TABLE SEMSEC

```
INSERT INTO SEMSEC VALUES ('4A01', 4, 'A');
INSERT INTO SEMSEC VALUES ('4C02', 4, 'C');
INSERT INTO SEMSEC VALUES ('8A01', 8, 'A');
INSERT INTO SEMSEC VALUES ('8C03', 8, 'C');
INSERT INTO SEMSEC VALUES ('5A01', 5, 'A');
```

```
 ssid | sem | sec
------+-----+-----
 4A01 |   4 | A
 4C02 |   4 | C
 8A01 |   8 | A
 8C03 |   8 | C
 5A01 |   5 | A
(5 rows)
```

TABLE CLASS

```
INSERT INTO CLASS VALUES ('1DS19IS049', '8A01');
INSERT INTO CLASS VALUES ('1DS19IS022', '8A01');
INSERT INTO CLASS VALUES ('1DS19IS017', '4C02');
INSERT INTO CLASS VALUES ('1DS19IS058', '4C02');
INSERT INTO CLASS VALUES ('1DS19IS007', '5A01');
```

```
   usn     | ssid
-----------+------
 1DS19IS049 | 8A01
 1DS19IS022 | 8A01
 1DS19IS017 | 4C02
 1DS19IS058 | 4C02
 1DS19IS007 | 5A01
(5 rows)
```

TABLE COURSE

```
INSERT INTO COURSE VALUES (1, 'DBMS' ,5, 4);
INSERT INTO COURSE VALUES (2, 'PHYSICS' ,8, 3);
INSERT INTO COURSE VALUES (3, 'CHEMISTRY' ,8, 4);
INSERT INTO COURSE VALUES (4, 'CNCS' ,4, 2);
INSERT INTO COURSE VALUES (5, 'PP' ,5, 3);
```

```
 subcode |        title         | sem | credits
--------+----------------------+-----+---------
       1 | DBMS                 |   5 |       4
       3 | CHEMISTRY            |   8 |       4
       4 | CNCS                 |   4 |       2
       5 | PP                   |   5 |       3
(5 rows)
```

TABLE IAMARKS

```
INSERT INTO IAMARKS VALUES ('1DS19IS049', 2, '8A01', 19, 18, 20, 0);
INSERT INTO IAMARKS VALUES ('1DS19IS058', 4, '4C02', 18, 12, 13, 0);
INSERT INTO IAMARKS VALUES ('1DS19IS022', 2, '8A01', 14, 16, 18, 0);
INSERT INTO IAMARKS VALUES ('1DS19IS007', 1, '5A01', 18, 16, 7, 0);
INSERT INTO IAMARKS VALUES ('1DS19IS017', 4, '4C02', 2, 4, 3, 0);
```

```
    usn     | subcode | ssid | test1 | test2 | test3 | finalia
------------+---------+------+-------+-------+-------+---------
 1DS19IS049 |       2 | 8A01 |    19 |    18 |    20 |    0.00
 1DS19IS058 |       4 | 4C02 |    18 |    12 |    13 |    0.00
 1DS19IS022 |       2 | 8A01 |    14 |    16 |    18 |    0.00
 1DS19IS007 |       1 | 5A01 |    18 |    16 |     7 |    0.00
 1DS19IS017 |       4 | 4C02 |     2 |     4 |     3 |    0.00
(5 rows)
```

QUERIES:->

List all the student details studying in 4th sem c sec

```
SELECT S1.* FROM STUDENT S1, SEMSEC S2, CLASS C
WHERE S1.USN = C.USN AND C.SSID = S2.SSID
AND S2.SEM = 4 AND S2.SEC = 'C';
```

```
   usn     |        sname         | address |   phone    |   gender
-----------+----------------------+---------+------------+------------
 1DS19IS017 | ANEESH               | MATTUR  | 7898676647 | MALE
 1DS19IS058 | VARSHA               | PUNE    | 8978916756 | FEMALE
(2 rows)
```

Compute the total no' of male and female students in each sem and section'

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

GENDER	SEM	SEC	COUNT(GENDER)
FEMALE	4	C	1
MALE	8	A	2
MALE	4	C	1
FEMALE	5	A	1

(5 rows)

Create a view of test1 marks of student usn 1ds19is049 in all subjects

```
CREATE VIEW IAMARKS_1 AS
SELECT SUBCODE , TEST1
FROM IAMARKS WHERE USN = '1DS19IS049';

SELECT * FROM IAMARKS_1;
```

```
 subcode | test1
---------+-------
    2    |    19
(1 row)
```

Calculate the final ia avg of best 2 test marks & update the corresponding finalia marks

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

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
1DS19IS049	2	8A01	19	18	20	19.5
1DS19IS058	4	4C02	18	12	13	15.5
1DS19IS022	2	8A01	14	16	18	17
1DS19IS007	1	5A01	18	16	7	17
1DS19IS017	4	4C02	2	4	3	3.5

(5 rows)

(the logic here is that you find the second highest marks by subtracting the total with the highest marks and the lowest marks)

Categorize the students based on following criteria
If finalia in between 17-20 then 'outstanding'
If in between 12-16 then 'average'
If finalia < 12 then 'weak'
Give these details for 8th sem, a, b and c sec students

```
SELECT SS.SEC ,S.*,
( 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, COURSE SUB
WHERE S.USN = IA.USN AND SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE  AND SUB.SEM = 8;
```

```
   usn     |       sname                  | address |       phone     |       gender    |     cat
  ------------+----------------------+-----------+------------+------------+-------------
 1DS19IS049 | ARJUN      | BANGALORE | 9535616756 | MALE       | OUTSTANDING
 1DS19IS022 | ADITYA   | BANGALORE | 7896716657 | MALE       | OUTSTANDING
(2 rows)
```

5. Consider the following database for a banking enterprise
BRANCH(branch-name:string, branch-city:string, assets:real)
ACCOUNT(accno:int, branch-name:string, balance:real)
DEPOSITOR(customer-name:string, accno:int)
CUSTOMER(customer-name:string, customer-street:string,
customer-city:string)
LOAN(loan-number:int, branch-name:string, amount:real)
BORROWER(customer-name:string, loan-number:int)

Write each of the following QUERIES in SQL:
i. Create the above tables by properly specifying the primary keys and the foreign keys
ii. Enter at least five tuples for each relation
iii. Find all the customers who have at least two accounts at the Main branch.
iv. Find all the customers who have an account at all the branches located in a specific city.
v. Demonstrate how you delete all account tuples at every branch located in a specific city.
vi. Find the names of all depositors of a specific branch.
vii. Find the details of all loan holder of a specific branch.

TABLE CREATION

BRANCH
```
CREATE TABLE branch
( branch_name char(15) primary key,
branch_city char(15),
assets numeric(10,2));
```

ACCOUNT
```
CREATE TABLE account
( accno int primary key,
branch_name varchar(15),
balance numeric(10,2),
foreign key(branch_name) references branch(branch_name)ON DELETE CASCADE );
```

DEPOSITOR
```
CREATE TABLE customer
( customer_name varchar(15) PRIMARY KEY,
customer_street varchar(15),
customer_city varchar(15)
);
```

CUSTOMER

```
CREATE TABLE loan
( loan_number int primary key,
branch_name varchar(15),
amount numeric(10,2),
foreign key(branch_name) REFERENCES branch(branch_name) );
```

LOAN

```
CREATE TABLE depositor
( customer_name varchar(15),
accno int, primary key(customer_name, accno),
foreign key(customer_name) REFERENCES customer(customer_name) on delete cascade,
foreign key(accno) REFERENCES account(accno) on delete cascade);
```

BORROWER

```
CREATE TABLE borrower
( customer_name varchar(15),
loan_number int,
primary key(customer_name, loan_number),
foreign key(customer_name) REFERENCES customer(customer_name),
foreign key(loan_number) REFERENCES loan(loan_number)
);
```

Tables data entry

```
insert into branch values      ('mgroad','bangalore',20000),
('borivali','mumbai',2000),
('banashankari','bangalore',45000),
('eastwing','bombay',1000),
('indiranagar','bangalore',500);
```

```
  branch_name   |   branch_city   |  assets
----------------+-----------------+----------
 mgroad         | bangalore       | 20000.00
 borivali       | mumbai          |  2000.00
 banashankari   | bangalore       | 45000.00
 eastwing       | bombay          |  1000.00
 indiranagar    | bangalore       |   500.00
(5 rows)
```

```
insert into account values(4,'borivali', 50000),
(22,'indiranagar', 500),
(13, 'indiranagar', 10000),
(45, 'banashankari', 30000),
(12, 'mgroad', 200000),
(15, 'banashankari', 25000);

 accno | branch_name  |  balance
-------+--------------+-----------
     4 | borivali     |  50000.00
    22 | indiranagar  |    500.00
            13 | indiranagar  |  10000.00
            45 | banashankari |  30000.00
           12 | mgroad       | 200000.00
           15 | banashankari |  25000.00
   (6 rows)

insert into customer values('Tarun', 'spooner st.', 'bangalore'),
('saqeeb', 'main road', 'shimoga'),('arjun', 'biker st.', 'mumbai'),
('john', 'waltress. st', 'bangalore'),('ashish', 'noName st.', 'bangalore');

 customer_name | customer_street | customer_city
---------------+-----------------+---------------
 Tarun         | spooner st.     | bangalore
 saqeeb        | main road       | shimoga
 arjun         | biker st.       | mumbai
 john          | waltress. st    | bangalore
 ashish        | noName st.      | bangalore
(5 rows)

insert into loan values(1, 'borivali', 20000),(2, 'borivali', 300000),(3, 'mgroad', 4500000),
(4, 'banashankari', 900000),(5, 'mgroad', 400000);
 loan_number | branch_name  |   amount
-------------+--------------+------------
           1 | borivali     |   20000.00
           2 | borivali     |  300000.00
           3 | mgroad       | 4500000.00
           4 | banashankari |  900000.00
           5 | mgroad       |  400000.00
(5 rows)
```

```
insert into depositor values('Tarun', 22), ('saqeeb', 13), ('arjun', 45), ('john', 12),
('ashish', 4),('‘arjun’, 15);
 customer_name | accno
---------------+-------
 Tarun         |    22
 saqeeb        |    13
 arjun         |    45
 john          |    12
 ashish        |     4
 arjun         |    15
(6 rows)

insert into borrower values('Tarun', 2),('saqeeb', 3), ('saqeeb', 4), ('john', 5), ('ashish', 1);
 customer_name | loan_number
---------------+-------------
 Tarun         |           2
 saqeeb        |           3
 saqeeb        |           4
 john          |           5
 ashish        |           1
(5 rows)
```

QUERIES:->

Find all the customers who have at least two accounts at the Main branch.

```
SELECT customer_name
FROM depositor d,account a
 where a.accno = d.accno and
 a.branch_name = 'banashankari'
 group by customer_name
 having count(*) >= 2;

 customer_name
---------------
 arjun
(1 row)
```

Find all the customers who have an account at all the branches located in a specific city.

```
SELECT d.customer_name
FROM account a,branch b,depositor d
WHERE b.branch_name=a.branch_name AND
a.accno=d.accno AND
b.branch_city='mumbai'
GROUP BY d.customer_name
HAVING COUNT(distinct b.branch_name)=(
SELECT COUNT(branch_name)
FROM branch
WHERE branch_city='mumbai');

 customer_name
---------------
 ashish
(1 row)
```

Find the names of all depositors of a specific branch.

```
SELECT distinct d.customer_name
From depositor d, account a
WHERE a.accno=d.accno and a.branch_name='mgroad';

 customer_name
---------------
 john
(1 row)
```

Find the details of all loan holder of a specific branch.

```
SELECT distinct b.customer_name
From loan l, borrower b
WHERE l.loan_number=b.loan_number and l.branch_name='mgroad';

 customer_name
---------------
 john
 saqeeb
(2 rows)
```

Demonstrate how you delete all account tuples at every branch located in a specific city.

```
delete from account
where branch_name in ( select branch_name
from branch  where branch_city ='mumbai');

select * from account;

 accno | branch_name  |  balance
-------+--------------+-----------
  22 | indiranagar  |    500.00
  13 | indiranagar  |  10000.00
  45 | banashankari |  30000.00
  12 | mgroad       | 200000.00
  15 | banashankari |  25000.00
(5 rows)
```