PART – A

EXPERIMENT 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 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

Solution :

Table creation:

```
CREATE TABLE PUBLISHER
                        (
                                NAME CHAR(15),
                                ADDRESS VARCHAR(20),
                                PHONE VARCHAR(10),
                                PRIMARY KEY(NAME)
                        );
CREATE TABLE LIBRARY_BRANCH
                        (
                                BRANCH_ID VARCHAR(5),
                                BRANCH_NAME CHAR(5),
                                ADDRESS VARCHAR(20),
                                PRIMARY KEY(BRANCH_ID)
                        );
CREATE TABLE BORROWERS
                        (
                                CARD_NO VARCHAR(10),
                                NAME CHAR(15),
                                ADDRESS VARCHAR(20),
                                PHONE VARCHAR(10),
                                PRIMARY KEY(CARD_NO)
                        );
CREATE TABLE BOOK
                (
                        BOOK_ID VARCHAR(5),
                        TITLE CHAR(20),
                        PUBLISHER_NAME CHAR(15),
                        PUB_YEAR INT,
                        (BOOK_ID),
                        FOREIGN KEY(PUBLISHER_NAME)
                        REFERENCES PUBLISHER(NAME)
                        ON DELETE CASCADE )
                        PARTITION BY RANGE(PUB_YEAR)
                        (PARTITION T1 VALUES LESS THAN(2016),
                        PARTITION T2 VALUES LESS THAN(2018)
                );
CREATE TABLE BOOK_AUTHORS
                        (
                                BOOK_ID VARCHAR(5),
                                AUTHOR_NAME CHAR(15),
                                PRIMARY KEY(BOOK_ID),
                                FOREIGN KEY(BOOK_ID)
                                REFERENCES BOOK(BOOK_ID)
                                ON DELETE CASCADE
                        );
CREATE TABLE BOOK_COPIES
                        (
                                BOOK_ID VARCHAR(5),
                                BRANCH_ID VARCHAR(5),
                                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
                        );
CREATE TABLE BOOK_LENDING
                        (
                                BOOK_ID VARCHAR(5),
                                BRANCH_ID VARCHAR(5),
                                CARD_NO VARCHAR(10),
                                DUE_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,
                                FOREIGN KEY(CARD_NO)
                                REFERENCES BORROWERS(CARD_NO)
                                ON DELETE CASCADE
                          );
```

```
INSERT INTO PUBLISHER  VALUES('&NAME','&ADDRESS','&PHONE');
INSERT INTO LIBRARY_BRANCH VALUES('&BRANCH_ID','&BRANCH_NAME','&ADDRESS');
INSERT INTO BORROWERS  VALUES('&CARD_NO','&NAME','&ADDRESS','&PHONE');
INSERT INTO BOOK VALUES('&BOOK_ID','&TITLE','&PUBLISHER_NAME','&PUB_YEAR');
INSERT INTO BOOK_AUTHORS VALUES('&BOOK_ID','&AUTHOR_NAME');
INSERT INTO BOOK_COPIES VALUES('&BOOK_ID','&BRANCH_ID','&NO_OF_COPIES');
INSERT INTO BOOK_LENDINGVALUES('&BOOK_ID','&BRANCH_ID','&CARD_NO','&DUE_OUT','&DUE_DATE');
```

```
select table_name from dba_tables where owner ='15IS048';
SET LINESIZE 120 PAGESIZE 2000;
```

QUERY 1:

```
@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@
SQL> select a.book_id,a.title,a.publisher_name,b.author_name,c.branch_id,c.no_of_copies
  2  from book a,book_authors b,book_copies c
  3  where a.book_id=b.book_id and a.book_id=c.book_id;
```

```
BOOK_ TITLE              PUBLISHER_NAME   AUTHOR_NAME     BRANC NO_OF_COPIES
----- -------------------- --------------- -------------- ----- ------------
611   MANAGEMENT         OXFORD              JOHN E                   03                  20
611   MANAGEMENT         OXFORD              JOHN E                   01                 10
614   JAVA                     NAROSA               PETER LINZ             05                 50
616   DBMS                     PEARSON            C K NAGPAL             05                 30
617   CO                       PEARSON              RAJKUMAR               03                 40
```

```
SELECT CARD_NO,NAME,ADDRESS,PHONE FROM BORROWERS WHERE CARD_NO IN
(SELECT DISTINCT CARD_NO FROM BOOK_LENDING
WHERE DUE_OUT BETWEEN '01-JAN-2017' AND '30-JUN-2017' GROUP BY(CARD_NO)
HAVING COUNT(DUE_OUT)>3);
```

```
SQL> UPDATE BOOK_LENDING SET CARD_NO='1011' WHERE BOOK_ID=615;
2 rows updated.

SQL>  SELECT * FROM BOOK_LENDING;

BOOK_ BRANC CARD_NO    DUE_OUT   DUE_DATE
----- ----- ---------- --------- ------------------------------
614   05           1011               12-AUG-15 12-AUG-16
616   02            1011              26-JAN-11 26-JAN-16
611   03           1014               15-MAR-15 15-APR-16
615   04           1011               19-DEC-16 16-JAN-17
615   01           1011               27-MAR-97 27-OCT-99

SQL>  UPDATE BOOK_LENDING SET DUE_OUT='20-JAN-2017' WHERE CARD_NO=1011;
4 rows updated.
SQL>  UPDATE BOOK_LENDING SET DUE_OUT='20-JUN-2017' WHERE CARD_NO='1011';
4 rows updated.
```

```
SQL>  SELECT * FROM BOOK_LENDING;

BOOK_ BRANC  CARD_NO    DUE_OUT   DUE_DATE
----- ----- ---------- --------- ---------
614   05           1011      20-JUN-17 12-AUG-16
616   02           1011      20-JUN-17 26-JAN-16
611   03          1014      15-MAR-15 15-APR-16
615   04          1011       20-JUN-17 16-JAN-17
615   01           1011      20-JUN-17 27-OCT-99
```

QUERY 2:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@

```
SQL> SELECT CARD_NO,NAME,ADDRESS,PHONE FROM BORROWERS WHERE CARD_NO IN
  2  (SELECT DISTINCT CARD_NO FROM BOOK_LENDING
  3  WHERE DUE_OUT BETWEEN '01-JAN-2017' AND '30-JUN-2017' GROUP BY(CARD_NO)
  4  HAVING COUNT(DUE_OUT)>3);

CARD_NO    NAME          ADDRESS           PHONE
---------- --------------- -------------------- ----------
1011      SHARIQE       MANGALORE         9964999012
SQL>  SELECT CARD_NO,NAME,ADDRESS,PHONE FROM BORROWERS WHERE CARD_NO IN
  2   (SELECT DISTINCT CARD_NO FROM BOOK_LENDING
  3   WHERE DUE_OUT BETWEEN '01-JAN-2017' AND '30-JUN-2017' GROUP BY(CARD_NO)
  4   HAVING COUNT(*)>3);

CARD_NO    NAME          ADDRESS           PHONE
---------- --------------- ------------------- ----------------
1011      SHARIQE       MANGALORE         9964999012
```

QUERY 3:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@

```
SQL> SELECT * FROM BOOK;


BOOK_ TITLE              PUBLISHER_NAME    PU
----- ------------------- --------------- ----
612   ATCI               CENGAGE
613   CN                 OXFORD
614   JAVA               NAROSA
615   CLOUD              CENGAGE
616   DBMS               PEARSON
617   CO                 PEARSON
6 rows selected.
SQL>  SELECT * FROM BOOK_COPIES;

BOOK_ BRANC NO_OF_COPIES
----- ----- ------------
616   05           30
617   03           40
614   05           50
SQL> SELECT * FROM BOOK_LENDING;
BOOK_ BRANC CARD_NO    DUE_OUT   DUE_DATE
----- ----- ---------- --------- ---------
614   05    1011       20-JUN-17 12-AUG-16
616   02    1011       20-JUN-17 26-JAN-16
615   04    1011       20-JUN-17 16-JAN-17
615   01    1011       20-JUN-17 27-OCT-99

SQL>  DELETE FROM BOOK WHERE BOOK_ID='614';
1 row deleted.
SQL>  SELECT * FROM BOOK_COPIES;
```

```
BOOK_ BRANC NO_OF_COPIES
----- ----- ------------
616   05          30
617   03          40



SQL> SELECT * FROM BOOK_LENDING;
BOOK_ BRANC CARD_NO    DUE_OUT   DUE_DATE
----- ----- ---------- --------- ---------
616   02    1011       20-JUN-17 26-JAN-16
615   04    1011       20-JUN-17 16-JAN-17
615   01    1011       20-JUN-17 27-OCT-99
```

QUERY 4:

```
@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@
SQL> INSERT INTO BOOK 2  VALUES('&BOOK_ID','&TITLE','&PUBLISHER_NAME','&PUB_YEAR');
Enter value for book_id: 511
Enter value for title: RAT
Enter value for publisher_name: CAT
Enter value for pub_year: 2018
old   2: VALUES('&BOOK_ID','&TITLE','&PUBLISHER_NAME','&PUB_YEAR')
new   2: VALUES('511','RAT','CAT','2018')

INSERT INTO BOOK
            *
ERROR at line 1:
ORA-14400: inserted partition key does not map to any partition
```

QUERY 5:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@

```
CREATE VIEW VV AS
SELECT A.BOOK_ID,A.TITLE,B.BRANCH_ID,B.NO_OF_COPIES
FROM BOOK A,BOOK_COPIES B
WHERE A.BOOK_ID=B.BOOK_ID;

SELECT * FROM VV;

BOOK_ TITLE              BRANC NO_OF_COPIES
----- ------------------- ----- ------------
616   DBMS               05           30
617   CO                 03           40

SQL> SELECT * FROM BOOK_COPIES;

BOOK_ BRANC NO_OF_COPIES
----- ----- ------------
616   05           30
617   03           40

SQL> SELECT * FROM LIBRARAY_BRANCH;
SELECT * FROM LIBRARAY_BRANCH
              *
ERROR at line 1:
ORA-00942: table or view does not exist

SQL>  SELECT * FROM LIBRARY_BRANCH;
```

```
BRANC BRANC ADDRESS
----- ----- --------------------
01    ISE         SAHYADRI
02    EC          SRINIVAS
03    MECH        JOSEPH
04    CIVIL       BEARYS
05    CS          PESIT

SQL> INSERT INTO BOOK_COPIES 2  VALUES('&BOOK_ID','&BRANCH_ID','&NO_OF_COPIES');
Enter value for book_id: 614
Enter value for branch_id: 05
Enter value for no_of_copies: 10
old   2: VALUES('&BOOK_ID','&BRANCH_ID','&NO_OF_COPIES')
new   2: VALUES('614','05','10')
INSERT INTO BOOK_COPIES
*
ERROR at line 1:
ORA-02291: integrity constraint (15IS048.SYS_C008120) violated - p
SQL> INSERT INTO BOOK_COPIES  2  VALUES('&BOOK_ID','&BRANCH_ID','&NO_OF_COPIES');
Enter value for book_id: 613
Enter value for branch_id: 05
Enter value for no_of_copies: 10
old   2: VALUES('&BOOK_ID','&BRANCH_ID','&NO_OF_COPIES')
new   2: VALUES('613','05','10')

1 row created.
SQL> SELCT ;L
SP2-0042: unknown command "SELCT ;L" - rest of line ignored.
SQL> SELECT SUM(NO_OF_COPIES)
  2  FROM BOOK_COPIES
  3  GROUP BY BRANCH_ID;
```

```
SUM(NO_OF_COPIES)
-----------------
       40
       40

SQL> SELECT BRANCH_ID, SUM(NO_OF_COPIES) FROM BOOK_COPIES
  2   GROUP BY BRANCH_ID;


BRANC SUM(NO_OF_COPIES)
----- -----------------
05                 40
03                 40

SQL>  SELECT BRANCH_ID, SUM(NO_OF_COPIES) AS TOTAL
  2   FROM BOOK_COPIES GROUP BY BRANCH_ID;

BRANC      TOTAL
----- ----------
05            40
03            40
```

@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@

EXPERIMENT 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 salesman who had more than one customer.

3. List all the salesman 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.

Solution :

Table creation

```
CREATE TABLE SALESMAN
                    (
                            SALESMAN_ID VARCHAR(20),
                            NAME CHAR(15),
                            CITY CHAR(15),
                            COMMISSION INT,
                            PRIMARY KEY(SALESMAN_ID)
                    );
CREATE TABLE CUSTOMER
                    (
                            CUSTOMER_ID VARCHAR(20),
                            CUST_NAME CHAR(15),
                            CITY CHAR(15),
                            GRADE FLOAT,
                            SALESMAN_ID VARCHAR(20),
                            PRIMARY KEY(CUSTOMER_ID),
                            FOREIGN KEY(SALESMAN_ID) REFERENCES
                            SALESMAN(SALESMAN_ID) ON DELETE CASCADE
                    );
CREATE TABLE ORDERS
                    (
                            ORDER_NO VARCHAR(15),
                            PURCHSE_AMT INT,
                            ORD_DATE DATE,
                            CUSTOMER_ID VARCHAR(20),
                            SALESMAN_ID VARCHAR(20),
                            PRIMARY KEY(ORDER_NO),
                            FOREIGN KEY(CUSTOMER_ID)
                            REFERENCES CUSTOMER(CUSTOMER_ID)
                            ON DELETE CASCADE,
                                        FOREIGN KEY(SALESMAN_ID)
                                        REFERENCES SALESMAN(SALESMAN_ID)
                                        ON DELETE CASCADE
                              );
INSERT INTO SALESMAN
VALUES('&SALESMAN_ID','&NAME','&CITY','&COMMISSION');
INSERT INTO CUSTOMER
VALUES('&CUSTOMER_ID','&CUST_NAME','&CITY','&GRADE','&SALESMAN_ID');
INSERT INTO ORDERS
VALUES('&ORDER_NO','&PURCHASE_AMT','&ORD_DATE','&CUSTOMER_ID','&SALESMAN_ID');

SQL> SELECT * FROM SALESMAN;

SALESMAN_ID      NAME       CITY       COMMISSION
-------------------- --------------- --------------- ----------
1000            RAVI        BANGALORE          12
1001            SOORAJ      DELHI              20
1002            PREM        LUCKNOW            15
1003            JOHN        BANGALORE          20
1004            RAJU        MYSORE             18

SQL>  SELECT * FROM CUSTOMER;

CUSTOMER_ID      CUST_NAME    CITY           GRADE SALESMAN_ID
-------------------- --------------- --------------- ---------- --------------------
C1                SHERYL       BANGALORE        4.5 1000
C2                DIYA         DELHI              5 1000
C3                PRIYA        MUMBAI           5.5 1001
C4                JACK         LUCKNOW          4.5 1002
C5                JILL         BANGALORE          9 1003
```

```
SQL> SELECT * FROM ORDERS
  2  ;

ORDER_NO       PURCHSE_AMT ORD_DATE  CUSTOMER_ID        SALESMAN_ID
--------------- ----------- --------- -------------------- --------------------
OR1              25000         25-MAY-17      C1             1000
OR2             15000          25-MAY-17       C2             1000
OR3              17000         25-MAY-17      C5             1003
OR4              30000         17-FEB-17       C4             1002
OR5              32000         17-FEB-17       C3             1001
OR6              14000         05-JUN-17       C1             1000
OR7             50000          10-JUL-17       C1             1000
```

QUERY1:

@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@

```
SELECT COUNT(CUSTOMER_ID)FROM CUSTOMER WHERE GRADE>
(SELECT AVG(GRADE)FROM CUSTOMER WHERE CITY='BANGALORE');
COUNT(CUSTOMER_ID)
------------------
             1
```

QUERY 2:

@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@

```
SELECT SALESMAN_ID,NAME FROM SALESMAN WHERE SALESMAN_ID IN
(SELECT SALESMAN_ID FROM CUSTOMER GROUP BY SALESMAN_ID HAVING COUNT(SALESMAN_ID)>1);

SALESMAN_ID        NAME
-------------------- ---------------
1000             RAVI
```

QUERY 3:

@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@

```
 (SELECT DISTINCT A.SALESMAN_ID,A.NAME,A.CITY FROM SALESMAN A,CUSTOMER B WHERE
A.SALESMAN_ID=B.SALESMAN_ID AND A.CITY=B.CITY)
UNION
(SELECT DISTINCT A.SALESMAN_ID,A.NAME,A.CITY FROM SALESMAN A,CUSTOMER B WHERE
A.SALESMAN_ID=B.SALESMAN_ID AND A.CITY!=B.CITY);

SALESMAN_ID        NAME          CITY
-------------------- --------------- ---------------
1000             RAVI          BANGALORE
1001             SOORAJ          DELHI
1002             PREM           LUCKNOW
1003             JOHN          BANGALORE
```

QUERY 4:

@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@

```
CREATE VIEW MAX_ORDER AS
SELECT A.SALESMAN_ID,A.NAME,B.ORD_DATE FROM SALESMAN A,ORDERS B WHERE
A.SALESMAN_ID=B.SALESMAN_ID AND
PURCHSE_AMT=(SELECT MAX(PURCHSE_AMT)FROM ORDERS C WHERE C.ORD_DATE=B.ORD_DATE);

View created.
SQL> SELECT * FROM MAX_ORDER;
SALESMAN_ID        NAME          ORD_DATE
-------------------- --------------- ---------
1000             RAVI          25-MAY-17
1001             SOORAJ          17-FEB-17
1000             RAVI          05-JUN-17
1000             RAVI          10-JUL-17
```

QUERY 5

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

```
SQL> SELECT * FROM SALESMAN;

SALESMAN_ID        NAME         CITY         COMMISSION
------------------- --------------- --------------- ----------
1000            RAVI           BANGALORE          12
1001            SOORAJ         DELHI              20
1002            PREM           LUCKNOW            15
1003            JOHN          BANGALORE           20
1004            RAJU          MYSORE              18

SQL> SELECT * FROM CUSTOMER;

CUSTOMER_ID        CUST_NAME      CITY            GRADE SALESMAN_ID
------------------- --------------- --------------- ---------- ----------------------------
C1                          SHERYL          BANGALORE           4.5 1000
C2                          DIYA             DELHI              5 1000
C3                          PRIYA           MUMBAI              5.5 1001
C4                           JACK            LUCKNOW            4.5 1002
C5                          JILL            BANGALORE            9 1003

SQL> SELECT *FROM OREDRS;
SELECT *FROM OREDRS


        *
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> SELECT * FROM ORDERS;
```

```
ORDER_NO       PURCHSE_AMT ORD_DATE  CUSTOMER_ID        SALESMAN_ID
--------------- ----------- --------- ------------------- --------------------
OR1            25000           25-MAY-17       C1                 1000
OR2            15000           25-MAY-17      C2                  1000
OR3            17000           25-MAY-17      C5                  1003
OR4            30000           17-FEB-17      C4                   1002
OR5            32000           17-FEB-17      C3                   1001
OR6            14000           05-JUN-17       C1                 1000
OR7            50000           10-JUL-17      C1                  1000
7 rows selected.
SQL> DELETE FROM SALESMAN WHERE SALESMAN_ID=1000;
1 row deleted.

SQL> SELECT * FROM SALESMAN
  2  ;
SALESMAN_ID        NAME         CITY         COMMISSION
-------------------- --------------- --------------- ----------
1001             SOORAJ        DELHI                20
1002             PREM          LUCKNOW            15
1003             JOHN          BANGALORE           20
1004             RAJU          MYSORE               18

SQL> SELECT * FROM CUSTOMER;

CUSTOMER_ID        CUST_NAME      CITY             GRADE SALESMAN_ID
-------------------- --------------- --------------- ---------- --------------------
C3               PRIYA         MUMBAI            5.5 1001
C4               JACK          LUCKNOW           4.5 1002
C5               JILL          BANGALORE           9 1003

SQL> SLECT * FROM ORDERS;
SP2-0734: unknown command beginning "SLECT * FR..." - rest of line ignored.
```

```
SQL> SELECT * FROM ORDERS;

ORDER_NO       PURCHSE_AMT ORD_DATE  CUSTOMER_ID         SALESMAN_ID
--------------- ----------- --------- ------------------- --------------------
OR3              17000 25-MAY-17 C5               1003
OR4              30000 17-FEB-17 C4               1002
OR5              32000 17-FEB-17 C3               1001
```

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

EXPERIMENT 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 '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' to 5.

Solution :

Table creation

```
CREATE TABLE ACTOR
        (
            ACT_ID VARCHAR(10),
            ACT_NAME CHAR(20),
            ACT_GENDER CHAR(6),
            PRIMARY KEY(ACT_ID)
        );
CREATE TABLE DIRECTOR
        (
            DIR_ID VARCHAR(10),
            DIR_NAME CHAR(20),
            DIR_PHONE VARCHAR(10),
            PRIMARY KEY(DIR_ID)
        );
CREATE TABLE MOVIES
        (
            MOV_ID VARCHAR(10),
            MOV_TITLE VARCHAR(20),
            MOV_YEAR INT,
            MOV_LANG CHAR(10),
            DIR_ID VARCHAR(10),
            PRIMARY KEY(MOV_ID),
            FOREIGN KEY(DIR_ID)
            REFERENCES DIRECTOR(DIR_ID)
            ON DELETE CASCADE
        );
```

```
CREATE TABLE MOVIE_CAST
        (
            ACT_ID VARCHAR(10),
            MOV_ID VARCHAR(10),
            ROLE CHAR(10),
            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
        );
CREATE TABLE RATING
        (
            MOV_ID VARCHAR(10),
            REV_STARS INT,
            MOVIE_REVIEWER VARCHAR(20),
            PRIMARY KEY(MOV_ID,REV_STARS,MOVIE_REVIEWER),
            FOREIGN KEY(MOV_ID)
            REFERENCES MOVIES(MOV_ID)
            ON DELETE CASCADE
        );

INSERT INTO ACTOR VALUES('&ACT_ID','&ACT_NAME','&ACT_GENDER');
INSERT INTO DIRECTOR VALUES('&DIR_ID','&DIR_NAME','&DIR_PHONE');
INSERT INTO MOVIES VALUES('&MOV_ID','&MOV_TITLE','&MOV_YEAR','&MOV_LANG','&DIR_ID');
INSERT INTO MOVIE_CAST VALUES('&ACT_ID','&MOV_ID','&ROLE');
INSERT INTO RATING VALUES('&MOV_ID','&REV_STARS','&MOVIE_REVIEWER');
```

```
SQL> SELECT * FROM ACTOR;
ACT_ID     ACT_NAME            ACT_GE
---------- ------------------- -------------
A1         SHARUKH KHAN        MALE
A2         SALMAN KHAN         MALE
A3         AMIR KHAN           MALE
A4         KAREENA KAPOOR      FEMALE
A5         KAJOL               FEMALE

SQL> SELECT * FROM DIRECTOR;

DIR_ID     DIR_NAME            DIR_PHONE
---------- ------------------- ----------------------
D1         HITCHCOCK           9945464951
D2         YASHRAJ             9964586855
D3         ROHIT SHETTY        7899744311
D4         STEVEN SPIELBERG    9844347348
D5         MADHUR BHANDARKAR   9964999012

SQL> SELECT * FROM MOVIES;

MOV_ID     MOV_TITLE           MOV_YEAR MOV_LANG   DIR_ID
---------- ------------------- ---------- ---------- --------
M1         HEROES              1990 ENGLISH    D4
M2         HORIZON             2001 ENGLISH    D3
M3         VERTIGO             2002 SPANISH    D1
M4         SPARROWS            1995 HEBREW     D2
M5         BIRDS               2017 ENGLISH    D1

SQL> SELECT * FROM MOVIE_CAST;
```

```
ACT_ID     MOV_ID     ROLE
---------- ---------- ----------
A1         M1         FATHER
A2         M2         SISTER
A3         M3         HERO
A1         M4         HERO
A1         M5         SUPPORT
A1         M2         SUPPORT

6 rows selected.

SQL> SELECT * FROM RATING;

MOV_ID      REV_STARS MOVIE_REVIEWER
---------- ---------- --------------------
M1                  7 BBC
M2                  6 BBC
M3                  7 RAM GOPAL
M4                  8 RAM GOPAL
M5                  9 TOI
```

QUERY 1:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

```
SELECT A.MOV_TITLE FROM MOVIES A,DIRECTOR B
WHERE A.DIR_ID=B.DIR_ID AND B.DIR_NAME='HITCHCOCK';

MOV_TITLE
--------------------
VERTIGO
BIRDS
```

QUERY 2:

@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@

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

MOV_TITLE
--------------------
BIRDS
SPARROWS
HORIZON
HEROES
```

QUERY 3:

@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@

```
 (SELECT DISTINCT A.ACT_NAME 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<2000)
INTERSECT
(SELECT DISTINCT A.ACT_NAME 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>2015);

ACT_NAME
--------------------
SHARUKH KHAN
```

QUERY 4:

```
@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@
SELECT A.MOV_TITLE,MAX(R.REV_STARS)FROM RATING R
JOIN MOVIES A ON A.MOV_ID=R.MOV_ID GROUP BY
A.MOV_TITLE ORDER BY A.MOV_TITLE;
MOV_TITLE          MAX(R.REV_STARS)
------------------- ----------------
BIRDS                    9
HEROES                   7
HORIZON                  6
SPARROWS                 8
VERTIGO                  7
```

QUERY 5:

```
@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@
UPDATE RATING SET REV_STARS=5 WHERE MOV_ID IN
(SELECT MOV_ID FROM MOVIES A,DIRECTOR D WHERE
D.DIR_NAME='STEVEN SPIELBERG' AND D.DIR_ID=A.DIR_ID);


MOV_ID     REV_STARS MOVIE_REVIEWER
---------- ---------- --------------------
M1              5 BBC
M2              6 BBC
M3              7 RAM GOPAL
M4              8 RAM GOPAL
M5              9 TOI
```

EXPERIMENT 4

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.

Solution :

Table creation

```
CREATE TABLE STUDENT
                (
                        USN VARCHAR(20),
                        SNAME CHAR(15),
                        ADDRESS VARCHAR(20),
                        PHONE NUMBER(10),
                        GENDER CHAR(6),
                        PRIMARY KEY(USN)
                );
CREATE TABLE SEMSEC
                (
                        SSID VARCHAR(20),
                        SEM NUMBER(3),
                        SEC CHAR(3),
                        PRIMARY KEY(SSID)
                );
CREATE TABLE CLASS
                (
                        USN VARCHAR(20),
                        SSID VARCHAR(20),
                        PRIMARY KEY(USN,SSID),
                        FOREIGN KEY(USN)
                        REFERENCES STUDENT(USN)
                        ON DELETE CASCADE,
                        FOREIGN KEY(SSID)
                        REFERENCES SEMSEC(SSID)
                        ON DELETE CASCADE
                );
```

```
CREATE TABLE SUBJECT
                    (
                            SUBCODE VARCHAR(15),
                            TITLE CHAR(20),
                            SEM NUMBER(3),
                            CREDITS INT,
                            PRIMARY KEY(SUBCODE)
                    );

CREATE TABLE IAMARKS
                    (
                            USN VARCHAR(20),
                            SUBCODE VARCHAR(15),
                            SSID VARCHAR(20),
                            TEST1 NUMBER(2),
                            TEST2 NUMBER(2),
                            TEST3 NUMBER(2),
                            FINALIA NUMBER(2),
                            PRIMARY KEY(USN,SUBCODE,SSID),
                            FOREIGN KEY(USN)
                            REFERENCES STUDENT(USN)
                            ON DELETE CASCADE,
                            FOREIGN KEY(SUBCODE)
                            REFERENCES SUBJECT(SUBCODE)
                            ON DELETE CASCADE,
                            FOREIGN KEY(SSID)
                            REFERENCES SEMSEC(SSID)
                            ON DELETE CASCADE
                    );
```

```
INSERT INTO STUDENT VALUES('&USN','&SNAME','&ADDRESS','&PHONE','&GENDER');
INSERT INTO SEMSECVALUES('&SSID','&SEM','&SEC');
INSERT INTO CLASS VALUES('&USN','&SSID');
INSERT INTO SUBJECT VALUES('&SUBCODE','&TITLE','&SEM','&CREDITS');
INSERT INTO IAMARKS VALUES('&USN','&SUBCODE','&SSID','&TEST1','&TEST2','&TEST3','&FINALIA');

SQL> SELECT * FROM STUDENT;

USN                  SNAME          ADDRESS             PHONE      GENDER
-------------------- -------------- ------------------- ---------- -------------------------------
1BI15CS101           SHARIQE        MANGALORE           9964999012 M
IS001                SUHAIL         BANGALORE           7899744311 M
CS048                SHAHANA        MYSORE              9844347348 F
12CV057              RAMNATH        OOTY                9945464951 M
16CS103              FARHEEN        DELHI               9844984220 F
12EC095              SHIFA          AGRA                7844784550 F
14ME76               FARZEEN        GOA                 9644984450 M
13IS023              FARHAN         RAJASTHAN           8866945321 M
10IS049              NAZEER         PUNE                9844651035 M
13CS092              MUNAZZA        CHENNAI             6514827981 F

10 rows selected.
SQL> SELECT * FROM SEMSEC;
SSID                 SEM SEC
-------------------- ---------- ---
S1                   3 A
S2                   8 B
S3                   8 C
S4                   8 A
S5                   4 C
S6                   4 A
6 rows selected.
```

```
SQL> SELECT * FROM CLASS;

USN                SSID
-------------------- --------------------
13CS092            S4
13IS023            S2
16CS103            S5
1BI15CS101         S1
1BI15CS101         S3
CS048              S3

6 rows selected.
SQL> SELECT * FROM SUBJECT;
SUBCODE       TITLE                 SEM    CREDITS
--------------- -------------------- ---------- ----------
IS010         UNIX                    3        4
IS020         DBMS                    4        4
IS030         OOMD                    8        3
IS040         JAVA                    8        2
IS050         FLAT                    5        4

SQL>  SELECT * FROM IAMARKS;

USN                SUBCODE      SSID                TEST1     TEST2     TEST3    FINALIA
-------------------- --------------- -------------------- ---------- ---------- ---------- ---------
13CS092            IS010        S4                  15         17          11
1BI15CS101         IS030        S3                  19          20         17
CS048              IS020        S3                  11         8           14
13IS023            IS010        S2                  14         8           10
16CS103            IS030        S5                   5         10           15
```

QUERY 1:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

```
SELECT A.*,B.SEM,B.SEC FROM
STUDENT A,SEMSEC B,CLASS C
WHERE A.USN=C.USN
AND B.SSID=C.SSID
AND B.SEM=4
AND B.SEC='C';
USN             SNAME        ADDRESS            PHONE GENDER     SEM SEC
------------------- -------------- ------------------- ---------- ------ ---------- ---
16CS103         FARHEEN      DELHI          9844984220 F         4 C
```

QUERY 2:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

```
SELECT A.SEM,A.SEC,COUNT(
CASE WHEN B.GENDER='M'THEN A.SID END)
AS MALECOUNT,
COUNT(CASE WHEN B.GENDER='F'THEN A.SID END)
AS FEMALECOUNT FROM STUDENT B,SEMSEC A,CLASS C
WHERE A.SSID=C.SSID AND B.USN=C.USN GROUP BY A.SEM,A.SEC;
```

QUERY 3:

@@@@@@@@@@@@@@@@@@@@@@-------***-------***---@@@@@@@@@@@@@@@@@@@@@@

```
CREATE VIEW STU_MARKS_TEST1 AS
SELECT SUBCODE,TEST1 FROM IAMARKS WHERE USN='1BI15CS101';
View created.
SQL>
SQL> SELECT * FROM STU_MARKS_TEST1;

SUBCODE         TEST1
--------------- ----------
IS030           19
```

QUERY 4:

```
@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@
UPDATE IAMARKS S,(
SELECT USN,MAX(T1)T4 FROM(
SELECT USN,AVG((TEST1+TEST2)/2)T1 FROM
STUDENT S1 GROUP BY USN UNION
SELECT USN,AVG((TEST2+TEST3)/2)T1 FROM
STUDENT S2 GROUP BY USN UNION
SELECT USN,AVG((TEST3+TEST1)/2)T1 FROM
STUDENT S3 GROUP BY USN )
AVGSCORS GROUP BY USN)UPD
SET FINALIA=UPD.T4 WHERE
S.USN=UPD.USN;
```

QUERY 5:

```
@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@
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 SU
WHERE S.USN=IA.USN AND SS.SSID=IA.SSID
AND SU.SUBCODE=IA.SUBCODE
AND SEC.SEM=8;
```

EXPERIMENT 5

Consider the schema for Company Database:

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)

Write SQL queries to

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.

2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise

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

4. Retrieve the name of each employee who works on all the projects controlledby department number 5 (use NOT EXISTS operator).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.

Solution :

Table creation

```
CREATE TABLE DEPARTMENT
                    (
                          DNO INT,
                          DNAME VARCHAR(20),
                          MGR_SSN VARCHAR(10),
                          MGR_START_DATE DATE,
                          PRIMARY KEY(DNO)
                    );
CREATE TABLE EMPLOYEE
                    (
                          SSN VARCHAR(10),
                          NAME VARCHAR(20),
                          ADDRESS VARCHAR(20),
                          SEX CHAR(1),
                          SALARY DECIMAL(10,3),
                          SUPER_SSN VARCHAR(10),
                          DNO INT,
                          PRIMARY KEY(SSN),
                          FOREIGN KEY(SUPER_SSN)
                          REFERENCES EMPLOYEE(SSN)
                          ON DELETE CASCADE,
                          FOREIGN KEY(DNO)
                          REFERENCES DEPARTMENT(DNO)
                          ON DELETE CASCADE
                    );
CREATE TABLE DLOCATION
                    (
                          DNO INT,
                          DLOC VARCHAR(20),
                          PRIMARY KEY(DNO,DLOC),
                          FOREIGN KEY(DNO)
                          REFERENCES DEPARTMENT(DNO)
                          ON DELETE CASCAD
                    );
CREATE TABLE PROJECT
                    (
                          PNO INT,
                          PNAME  VARCHAR(20),
                          PLOCATION  VARCHAR(20),
                          DNO INT,
                          PRIMARY KEY(PNO),
                          FOREIGN KEY(DNO)
                          REFERENCES DEPARTMENT(DNO)
                          ON DELETE CASCADE
                    );
CREATE TABLE WORKS_ON
                    (
                          SSN VARCHAR(20),
                          PNO INT,
                          HOURS INT,
                          PRIMARY KEY(SSN,PNO),
                          FOREIGN KEY(SSN)
                          REFERENCES EMPLOYEE(SSN)
                          ON DELETE CASCADE,
                          FOREIGN KEY(PNO)
                          REFERENCES PROJECT(PNO)
                          ON DELETE CASCADE
                    );
```

```
ALTER TABLE DEPARTMENT
ADD CONSTRAINT C_MSSN
FOREIGN KEY(MGR_SSN)
REFERENCES EMPLOYEE(SSN)
ON DELETE CASCADE;
UPDATE DEPARTMENT
SET MGR_SSN='&MGR_SSN'
WHERE DNAME='&DNAME';

INSERT INTO DEPARTMENT VALUES('&DNO','&DNAME','&MGRSSN','&MGRSTARTDATE');
INSERT INTO EMPLOYEEVALUES('&SSN','&NAME','&ADDRESS','&SEX','&SALARY','&SUPERSSN','&DNO');
INSERT INTO DLOCATION VALUES('&DNO','&DLOC');
INSERT INTO PROJECT  VALUES('&PNO','&PNAME','&PLOCATION','&DNO');
INSERT INTO WORKS_ON VALUES('&SSN','&PNO','&HOURS');

SELECT * FROM DEPARTMENT;
  DNO DNAME              MGR_SSN    MGR_START
----- ------------------- ---------- ---------
    1 ACCOUNTS                      01-JAN-00
    2 FINANCE                       01-FEB-01
    3 RESEARCH                      03-MAR-01
    4 HR                            04-APR-02
    5 SOFTWARE                      05-FEB-03

SQL> SELECT * FROM EMPLOYEE;
SSN        NAME                ADDRESS             S    SALARY SUPER_SSN
---------- ------------------- ------------------- - ---------- ---------- -------
E1         SHAWN               FRANCE              M    600000   E1
E2         SCOTT               LONDON              M    700000   E2
E3         ROBERT              ENGLAND             M    500000   E3
E4         ROBERT              FRANCE              M    400000   E4
```

```
E6      TOM           LONDON       M   200000   E6
E7      JOHN          USA          M   100000   E7
E8      TEENA         NEWYORK      F   500000   E8
E5      MICKY         LONDON       M   300000   E5
E9      ROSHEL        KOREA        F   700000   E9
E10     SMITH         LONDON       F   400000   E10
10 rows selected.
```

AFTER UPDATING :

```
SQL> SELECT * FROM DEPARTMENT;

      DNO DNAME              MGR_SSN    MGR_START
---------- -------------------- ---------- ---------
        1 ACCOUNTS           E1         01-JAN-00
        2 FINANCE            E2         01-FEB-01
        3 RESEARCH           E3         03-MAR-01
        4 HR                 E4         04-APR-02
        5 SOFTWARE           E5         05-FEB-03

SQL> SELECT * FROM DLOCATION;

      DNO DLOC
---------- --------------------
        1 FRANCE
        2 LONDON
        3 NEWYORK
        4 ENGLAND
        5 KOREA
```

```
SQL> SELECT * FROM PROJECT;

      PNO PNAME                PLOCATION            DNO
---------- -------------------- -------------------- --------------------
        1  IOT                  FRANCE                1
        2  CLOUD COMPUTING      LONDON                2
        3  MACHINE LEARNING     NEWYORK               3
        4  DATA MINING          ENGLAND               4
        5  ANDROID APP          KOREA                 5


SQL> SELECT * FROM WORKS_ON;

SSN                  PNO     HOURS
-------------------- ---------- ----------
E1                   1       2
E2                   2       4
E3                   3       5
E4                   4       6
E5                   5       4
E6                   5       2

6 rows selected.
```

QUERY 1:

@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@

```
 (SELECT DISTINCT P.PNO FROM  PROJECT P,EMPLOYEE E,DEPARTMENT D WHERE
E.DNO=D.DNO AND
D.DNO=P.DNO AND
D.MGR_SSN=E.SSN AND
NAME='SCOTT') UNION
(SELECT DISTINCT P.PNO FROM  PROJECT P,EMPLOYEE E,WORKS_ON W WHERE
E.SSN=W.SSN AND
P.DNO=W.PNO AND
NAME='SCOTT');

      PNO
     --------
        2
```

QUERY 2:

@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@

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

NAME                 INCREASEDSALARY
-------------------- ---------------
SHAWN                         660000
```

QUERY 3:

@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@

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

SUM(SALARY) MAX(SALARY) MIN(SALARY) AVG(SALARY)
----------- ----------- ----------- -----------
    2500000      700000      100000  416666.667
```

QUERY 4:

```
@@@@@@@@@@@@@@@@@@@@@@-------***-------***----@@@@@@@@@@@@@@@@@@@@@@
SELECT NAME FROM EMPLOYEE E WHERE NOT EXISTS
((SELECT PNO FROM PROJECT WHERE DNO=5)MINUS
(SELECT PNO FROM WORKS_ON W WHERE E.SSN=W.SSN));

NAME
--------------------
TOM
MICKY
```

QUERY 5:

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

       DNO   COUNT(*)
    --------- ----------
         1         1
```

******************** END ********************