Dayananda Sagar College of Engineering
Department of MCA
LABORATORY MANUAL

PART-A

SQL Programming

• Design, develop, and implement the specified queries for the following problems using Oracle, MySQL, MS SQL Server, or any other DBMS under LINUX/Windows environment.

• Create Schema and insert at least 5 records for each table. Add appropriate database constraints.

PART-B

PL/SQL Programming

- Performing Essential PL/SQL Tasks
- Conditional and Sequential and iterative statements in PL/SQL
- Working with stored procedures and functions
- Working with Implicit and explicit Cursors
- Handling exception in PL/SQL
- Implementing trigger in PL/SQL

PART A: SQL PROGRAMMING

1. 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)

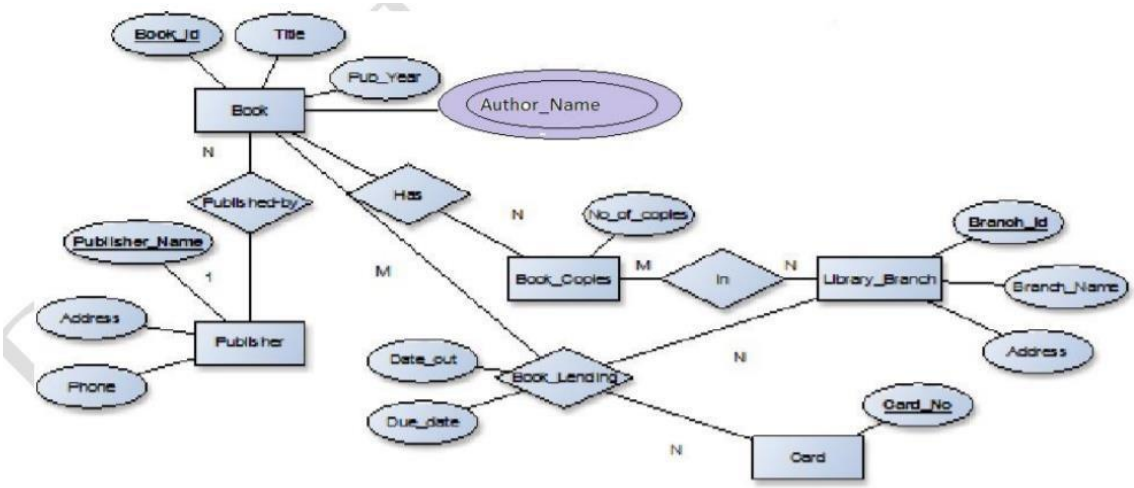
   LIBRARY_BRANCH (Branch_id, Branch_Name, Address)

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 Jun2017**
3. **Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulationoperation.**
4. **Partition the BOOK table based on year of publication. Demonstrate its working with a simplequery.**
5. **Create a view of all books and its number of copies that are currently available in the Library.**

Solution:

Entity-Relationship Diagram



Schema Diagram

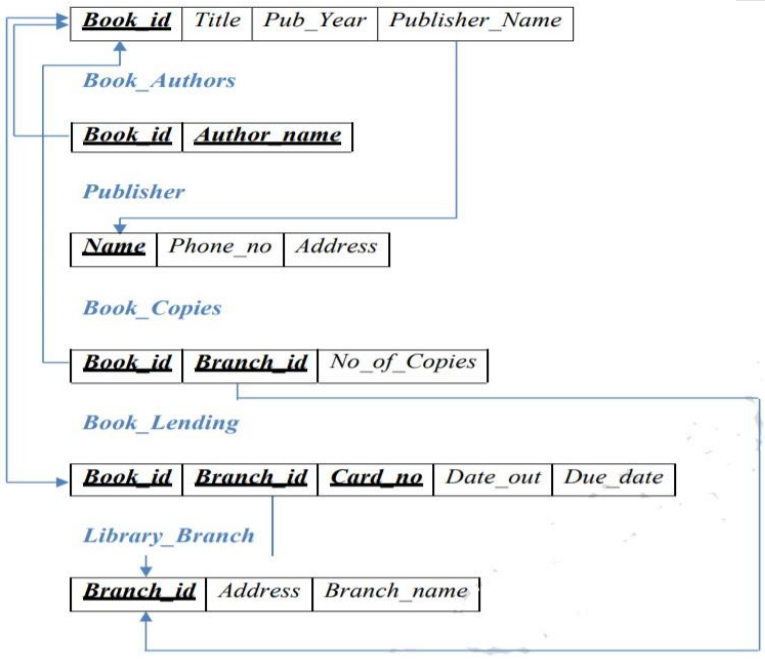


Table Creation

```
CREATE TABLE PUBLISHER
(NAME VARCHAR (20) PRIMARY KEY,
PHONE BIGINT,
ADDRESS VARCHAR (20));

CREATE TABLE BOOK
(BOOK_ID INTEGER 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 INTEGER,
FOREIGN KEY(BOOK_ID) REFERENCES BOOK (BOOK_ID) ON DELETE CASCADE,
PRIMARY KEY (BOOK_ID, AUTHOR_NAME));

CREATE TABLE LIBRARY_BRANCH
(BRANCH_ID INTEGER PRIMARY KEY,
BRANCH_NAME VARCHAR (50),
ADDRESS VARCHAR (50));

CREATE TABLE BOOK_COPIES
(NO_OF_COPIES INTEGER,
BOOK_ID INTEGER,
FOREIGN KEY(BOOK_ID) REFERENCES BOOK (BOOK_ID) ON DELETE CASCADE,
BRANCH_ID INTEGER,
FOREIGN KEY(BRANCH_ID) REFERENCES LIBRARY_BRANCH (BRANCH_ID) ON DELETE CASCADE,
PRIMARY KEY (BOOK_ID, BRANCH_ID));

CREATE TABLE CARD
(CARD_NO INTEGER PRIMARY KEY);
```

```
CREATE TABLE BOOK_LENDING
(DATE_OUT DATE,
DUE_DATE DATE,
BOOK_ID INTEGER,
FOREIGN KEY(BOOK_ID) REFERENCES BOOK (BOOK_ID) ON DELETE CASCADE,
BRANCH_ID INTEGER,
FOREIGN KEY(BRANCH_ID) REFERENCES LIBRARY_BRANCH (BRANCH_ID) ON DELETE CASCADE,
CARD_NO INTEGER,
FOREIGN KEY(CARD_NO) REFERENCES CARD (CARD_NO) ON DELETE CASCADE,
PRIMARY KEY (BOOK_ID, BRANCH_ID, CARD_NO));
```

Table Descriptions

```
mysql> DESC PUBLISHER;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| NAME    | varchar(20) | NO   | PRI | NULL    |       |
| PHONE   | bigint(20)  | YES  |     | NULL    |       |
| ADDRESS | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> DESC BOOK;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| BOOK_ID        | int(11)     | NO   | PRI | NULL    |       |
| TITLE          | varchar(20) | YES  |     | NULL    |       |
| PUB_YEAR       | varchar(20) | YES  |     | NULL    |       |
| PUBLISHER_NAME | varchar(20) | YES  | MUL | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
4 rows in set (0.04 sec)
```

```
mysql> DESC BOOK_AUTHORS;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| AUTHOR_NAME | varchar(20) | NO   | PRI | NULL    |       |
| BOOK_ID     | int(11)     | NO   | PRI | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
2 rows in set (0.04 sec)
```

```
mysql> DESC LIBRARY_BRANCH;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| BRANCH_ID   | int(11)     | NO   | PRI | NULL    |       |
| BRANCH_NAME | varchar(50) | YES  |     | NULL    |       |
| ADDRESS     | varchar(50) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> DESC BOOK_COPIES;
+--------------+---------+------+-----+---------+-------+
| Field        | Type    | Null | Key | Default | Extra |
+--------------+---------+------+-----+---------+-------+
| NO_OF_COPIES | int(11) | YES  |     | NULL    |       |
| BOOK_ID      | int(11) | NO   | PRI | NULL    |       |
| BRANCH_ID    | int(11) | NO   | PRI | NULL    |       |
+--------------+---------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> DESC CARD;
+---------+---------+------+-----+---------+-------+
| Field   | Type    | Null | Key | Default | Extra |
+---------+---------+------+-----+---------+-------+
| CARD_NO | int(11) | NO   | PRI | NULL    |       |
+---------+---------+------+-----+---------+-------+
1 row in set (0.00 sec)
```

```
mysql> DESC BOOK_LENDING;
+-----------+---------+------+-----+---------+-------+
| Field     | Type    | Null | Key | Default | Extra |
+-----------+---------+------+-----+---------+-------+
| DATE_OUT  | date    | YES  |     | NULL    |       |
| DUE_DATE  | date    | YES  |     | NULL    |       |
| BOOK_ID   | int(11) | NO   | PRI | NULL    |       |
| BRANCH_ID | int(11) | NO   | PRI | NULL    |       |
| CARD_NO   | int(11) | NO   | PRI | NULL    |       |
+-----------+---------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Insertion of Values into Tables

INSERT INTO PUBLISHER VALUES ('MCGRAW-HILL', 9989076587, 'BANGALORE'); INSERT INTO PUBLISHER VALUES ('PEARSON', 9889076565, 'NEWDELHI'); INSERT INTO PUBLISHER VALUES ('RANDOM HOUSE', 7455679345, 'HYDRABAD');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,'DSCE','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);
```

Display all Values

```
mysql> SELECT * FROM PUBLISHER;
+----------------+------------+-----------+
| NAME           | PHONE      | ADDRESS   |
+----------------+------------+-----------+
| GRUPO PLANETA  | 7756120238 | BANGALORE |
| HACHETTE LIVRE | 8970862340 | CHENNAI   |
| MCGRAW-HILL    | 9989076587 | BANGALORE |
| PEARSON        | 9889076565 | NEWDELHI  |
| RANDOM HOUSE   | 7455679345 | HYDRABAD  |
+----------------+------------+-----------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM BOOK;
+---------+-------+----------+----------------+
| BOOK_ID | TITLE | PUB_YEAR | PUBLISHER_NAME |
+---------+-------+----------+----------------+
|       1 | DBMS  | JAN-2017 | MCGRAW-HILL    |
|       2 | ADBMS | JUN-2016 | MCGRAW-HILL    |
|       3 | CN    | SEP-2016 | PEARSON        |
|       4 | CG    | SEP-2015 | GRUPO PLANETA  |
|       5 | OS    | MAY-2016 | PEARSON        |
+---------+-------+----------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM BOOK_AUTHORS;
+--------------+---------+
| AUTHOR_NAME  | BOOK_ID |
+--------------+---------+
| NAVATHE      |       1 |
| NAVATHE      |       2 |
| TANENBAUM    |       3 |
| EDWARD ANGEL |       4 |
| GALVIN       |       5 |
+--------------+---------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM LIBRARY_BRANCH;
+-----------+--------------+-----------+
| BRANCH_ID | BRANCH_NAME  | ADDRESS   |
+-----------+--------------+-----------+
|        10 | RR NAGAR     | BANGALORE |
|        11 | DSCE         | BANGALORE |
|        12 | RAJAJI NAGAR | BANGALORE |
|        13 | NITTE        | MANGALORE |
|        14 | MANIPAL      | UDUPI     |
+-----------+--------------+-----------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM BOOK_COPIES;
+--------------+---------+-----------+
| NO_OF_COPIES | BOOK_ID | BRANCH_ID |
+--------------+---------+-----------+
|           10 |       1 |        10 |
|            5 |       1 |        11 |
|            2 |       2 |        12 |
|            5 |       2 |        13 |
|            7 |       3 |        14 |
|            3 |       4 |        11 |
|            1 |       5 |        10 |
+--------------+---------+-----------+
7 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM CARD;
+---------+
| CARD_NO |
+---------+
|     100 |
|     101 |
|     102 |
|     103 |
|     104 |
+---------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM BOOK_LENDING;
+------------+------------+---------+-----------+---------+
| DATE_OUT   | DUE_DATE   | BOOK_ID | BRANCH_ID | CARD_NO |
+------------+------------+---------+-----------+---------+
| 2017-01-01 | 2017-06-01 |       1 |        10 |     101 |
| 2017-04-12 | 2017-05-12 |       1 |        11 |     104 |
| 2017-02-21 | 2017-04-21 |       2 |        13 |     101 |
| 2017-01-11 | 2017-03-11 |       3 |        14 |     101 |
| 2017-03-15 | 2017-07-15 |       4 |        11 |     101 |
+------------+------------+---------+-----------+---------+
5 rows in set (0.00 sec)
```

Queries

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

```
mysql> 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;
+---------+-------+----------------+-------------+--------------+-----------+
| BOOK_ID | TITLE | PUBLISHER_NAME | AUTHOR_NAME | NO_OF_COPIES | BRANCH_ID |
+---------+-------+----------------+-------------+--------------+-----------+
|       1 | DBMS  | MCGRAW-HILL    | NAVATHE     |           10 |        10 |
|       1 | DBMS  | MCGRAW-HILL    | NAVATHE     |            5 |        11 |
|       2 | ADBMS | MCGRAW-HILL    | NAVATHE     |            2 |        12 |
|       2 | ADBMS | MCGRAW-HILL    | NAVATHE     |            5 |        13 |
|       3 | CN    | PEARSON        | TANENBAUM   |            7 |        14 |
|       4 | CG    | GRUPO PLANETA  | EDWARD ANGEL|            3 |        11 |
|       5 | OS    | PEARSON        | GALVIN      |            1 |        10 |
+---------+-------+----------------+-------------+--------------+-----------+
7 rows in set (0.03 sec)
```

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

```
mysql> SELECT CARD_NO
    -> FROM BOOK_LENDING
    -> WHERE DATE_OUT BETWEEN '2017-01-01' AND '2017-07-01'
    -> GROUP BY CARD_NO
    -> HAVING COUNT(*) > 3;
+---------+
| CARD_NO |
+---------+
|     101 |
+---------+
1 row in set (0.00 sec)
```

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

```
mysql> DELETE FROM BOOK
    -> WHERE BOOK_ID=3;
Query OK, 1 row affected (0.05 sec)

mysql> SELECT * FROM BOOK;
+---------+-------+----------+----------------+
| BOOK_ID | TITLE | PUB_YEAR | PUBLISHER_NAME |
+---------+-------+----------+----------------+
|       1 | DBMS  | JAN-2017 | MCGRAW-HILL    |
|       2 | ADBMS | JUN-2016 | MCGRAW-HILL    |
|       4 | CG    | SEP-2015 | GRUPO PLANETA  |
|       5 | OS    | MAY-2016 | PEARSON        |
+---------+-------+----------+----------------+
4 rows in set (0.00 sec)
```

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

```
mysql> CREATE VIEW V_PUBLICATION AS
    -> SELECT PUB_YEAR
    -> FROM BOOK;
Query OK, 0 rows affected (0.04 sec)

mysql> SELECT * FROM V_PUBLICATION;
+----------+
| PUB_YEAR |
+----------+
| JAN-2017 |
| JUN-2016 |
| SEP-2015 |
| MAY-2016 |
+----------+
4 rows in set (0.03 sec)
```

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

```
mysql> 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;
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM V_BOOKS;
+---------+-------+--------------+
| BOOK_ID | TITLE | NO_OF_COPIES |
+---------+-------+--------------+
|       1 | DBMS  |           10 |
|       1 | DBMS  |            5 |
|       2 | ADBMS |            2 |
|       2 | ADBMS |            5 |
|       4 | CG    |            3 |
|       5 | OS    |            1 |
+---------+-------+--------------+
6 rows in set (0.02 sec)
```

2. Consider the following schema for OrderDatabase:

SALESMAN (Salesman_id, Name, City, Commission)

CUSTOMER (Customer_id, Cust_Name, City, Grade,Salesman_id)

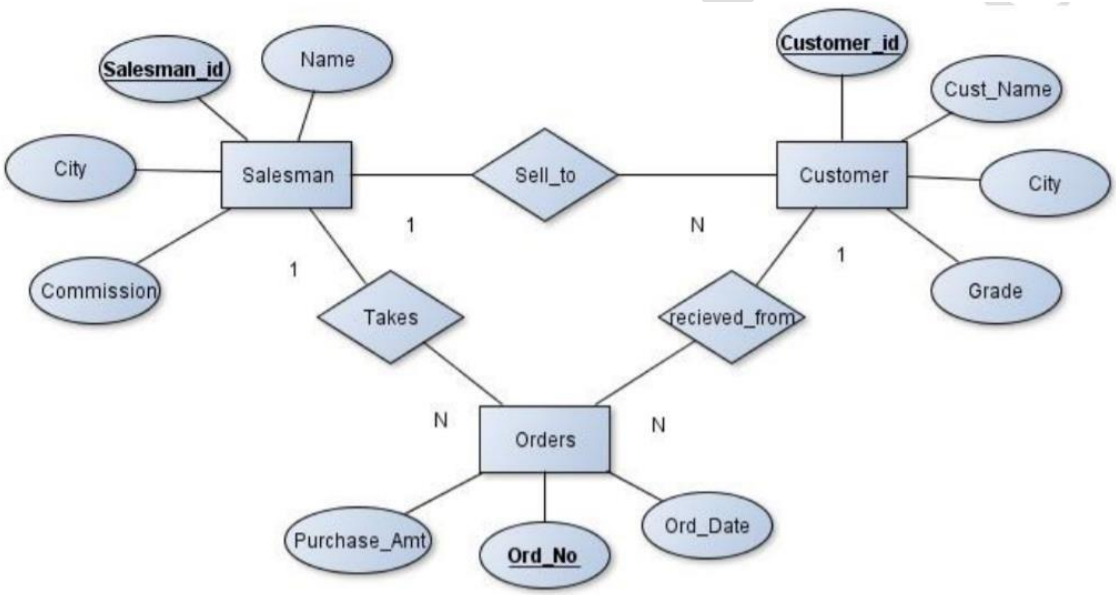
ORDERS (Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id)

Write SQL queries to

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.**

Solution:

Entity-Relationship Diagram



Schema Diagram

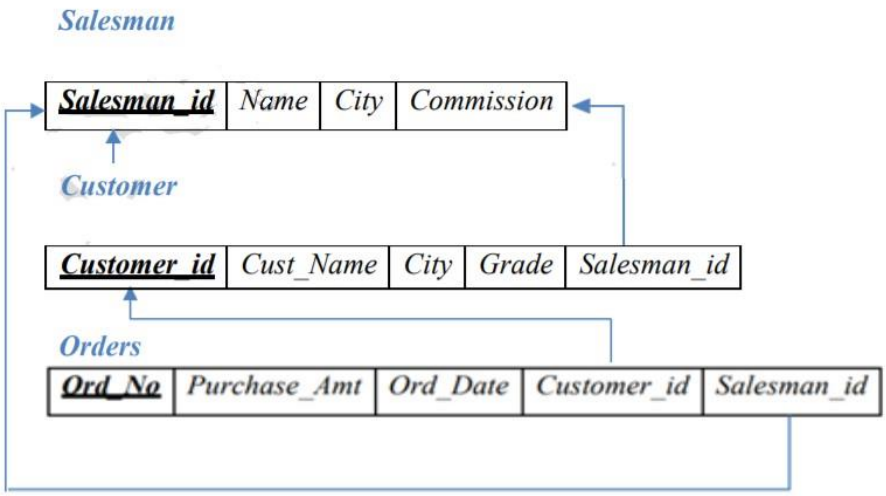


Table Creation

```
CREATE TABLE SALESMAN
(SALESMAN_ID INTEGER (4),
NAME VARCHAR (20),
CITY VARCHAR (20),
COMMISSION VARCHAR (20),
PRIMARY KEY (SALESMAN_ID));

CREATE TABLE CUSTOMER
(CUSTOMER_ID INTEGER (4),
CUST_NAME VARCHAR (20),
CITY VARCHAR (20),
GRADE INTEGER (3),
PRIMARY KEY (CUSTOMER_ID),
SALESMAN_ID INTEGER (4),
FOREIGN KEY (SALESMAN_ID) REFERENCES SALESMAN (SALESMAN_ID) ON DELETE SET NULL);
```

```
CREATE TABLE ORDERS
(ORD_NO INTEGER (5),
PURCHASE_AMT FLOAT (10, 2),
ORD_DATE DATE,
PRIMARY KEY (ORD_NO),
CUSTOMER_ID INTEGER (4),
FOREIGN KEY (CUSTOMER_ID) REFERENCES CUSTOMER1 (CUSTOMER_ID) ON DELETE CASCADE,
SALESMAN_ID INTEGER (4),
FOREIGN KEY (SALESMAN_ID) REFERENCES SALESMAN (SALESMAN_ID) ON DELETE CASCADE);
```

Table Descriptions

```
mysql> DESC SALESMAN;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| SALESMAN_ID | int(4)      | NO   | PRI | NULL    |       |
| NAME        | varchar(20) | YES  |     | NULL    |       |
| CITY        | varchar(20) | YES  |     | NULL    |       |
| COMMISSION  | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
4 rows in set (0.05 sec)
```

```
mysql> DESC CUSTOMER;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| CUSTOMER_ID | int(4)      | NO   | PRI | NULL    |       |
| CUST_NAME   | varchar(20) | YES  |     | NULL    |       |
| CITY        | varchar(20) | YES  |     | NULL    |       |
| GRADE       | int(3)      | YES  |     | NULL    |       |
| SALESMAN_ID | int(4)      | YES  | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.05 sec)
```

```
mysql> DESC ORDERS;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| ORD_NO       | int(5)      | NO   | PRI | NULL    |       |
| PURCHASE_AMT | float(10,2) | YES  |     | NULL    |       |
| ORD_DATE     | date        | YES  |     | NULL    |       |
| CUSTOMER_ID  | int(4)      | YES  | MUL | NULL    |       |
| SALESMAN_ID  | int(4)      | YES  | MUL | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
5 rows in set (0.03 sec)
```

Insertion of Values into Tables

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','HYDRABAD','15%');
INSERT INTO CUSTOMER VALUES (10, 'PREETHI','BANGALORE', 100, 1000);
INSERT INTO CUSTOMER VALUES (11, 'VIVEK','MANGALORE', 300, 1000);
INSERT INTO CUSTOMER VALUES (12, 'BHASKAR','CHENNAI', 400, 2000);
INSERT INTO CUSTOMER VALUES (13, 'CHETHAN','BANGALORE', 200, 2000);
INSERT INTO CUSTOMER VALUES (14, 'MAMATHA','BANGALORE', 400, 3000);

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);

Display all Values

```
mysql> SELECT * FROM SALESMAN;
+-------------+--------+-----------+------------+
| SALESMAN_ID | NAME   | CITY      | COMMISSION |
+-------------+--------+-----------+------------+
|        1000 | JOHN   | BANGALORE | 25 %       |
|        2000 | RAVI   | BANGALORE | 20 %       |
|        3000 | KUMAR  | MYSORE    | 15 %       |
|        4000 | SMITH  | DELHI     | 30 %       |
|        5000 | HARSHA | HYDRABAD  | 15%        |
+-------------+--------+-----------+------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM CUSTOMER;
+-------------+-----------+-----------+-------+-------------+
| CUSTOMER_ID | CUST_NAME | CITY      | GRADE | SALESMAN_ID |
+-------------+-----------+-----------+-------+-------------+
|          10 | PREETHI   | BANGALORE |   100 |        1000 |
|          11 | VIVEK     | MANGALORE |   300 |        1000 |
|          12 | BHASKAR   | CHENNAI   |   400 |        2000 |
|          13 | CHETHAN   | BANGALORE |   200 |        2000 |
|          14 | MAMATHA   | BANGALORE |   400 |        3000 |
+-------------+-----------+-----------+-------+-------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM ORDERS;
+--------+--------------+------------+-------------+-------------+
| ORD_NO | PURCHASE_AMT | ORD_DATE   | CUSTOMER_ID | SALESMAN_ID |
+--------+--------------+------------+-------------+-------------+
|     50 |      5000.00 | 2017-05-04 |          10 |        1000 |
|     51 |       450.00 | 2017-01-20 |          10 |        2000 |
|     52 |      1000.00 | 2017-02-24 |          13 |        2000 |
|     53 |      3500.00 | 2017-04-13 |          14 |        3000 |
|     54 |       550.00 | 2017-03-09 |          12 |        2000 |
+--------+--------------+------------+-------------+-------------+
5 rows in set (0.04 sec)
```

Queries

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

```
mysql> SELECT GRADE, COUNT(DISTINCT CUSTOMER_ID) FROM CUSTOMER
    -> GROUP BY GRADE
    -> HAVING GRADE > (SELECT AVG(GRADE) FROM CUSTOMER WHERE CITY='BANGALORE');
+-------+----------------------------+
| GRADE | COUNT(DISTINCT CUSTOMER_ID) |
+-------+----------------------------+
|   300 |                          1 |
|   400 |                          2 |
+-------+----------------------------+
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT SALESMAN_ID, NAME
    -> FROM SALESMAN S
    -> WHERE 1 < (SELECT COUNT(*) FROM CUSTOMER WHERE SALESMAN_ID = S.SALESMAN_ID);
+-------------+------+
| SALESMAN_ID | NAME |
+-------------+------+
|        1000 | JOHN |
|        2000 | RAVI |
+-------------+------+
2 rows in set (0.04 sec)
```

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

```
mysql> SELECT SALESMAN.SALESMAN_ID, NAME, CUST_NAME, COMMISSION
    -> FROM SALESMAN, CUSTOMER
    -> WHERE SALESMAN.CITY = CUSTOMER.CITY
    -> UNION
    -> SELECT SALESMAN_ID, NAME, 'NO MATCH', COMMISSION
    -> FROM SALESMAN
    -> WHERE NOT CITY = ANY
    -> (SELECT CITY FROM CUSTOMER)
    -> ORDER BY 2 DESC;
+-------------+--------+-----------+------------+
| SALESMAN_ID | NAME   | CUST_NAME | COMMISSION |
+-------------+--------+-----------+------------+
|        4000 | SMITH  | NO MATCH  | 30 %       |
|        2000 | RAVI   | CHETHAN   | 20 %       |
|        2000 | RAVI   | PREETHI   | 20 %       |
|        2000 | RAVI   | MAMATHA   | 20 %       |
|        3000 | KUMAR  | NO MATCH  | 15 %       |
|        1000 | JOHN   | PREETHI   | 25 %       |
|        1000 | JOHN   | MAMATHA   | 25 %       |
|        1000 | JOHN   | CHETHAN   | 25 %       |
|        5000 | HARSHA | NO MATCH  | 15%        |
+-------------+--------+-----------+------------+
9 rows in set (0.04 sec)
```

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

```
mysql> 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(PURCHASE_AMT) FROM ORDERS C WHERE C.ORD_DATE = B.ORD_DATE);
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM ELITSALESMAN;
+------------+-------------+-------+
| ORD_DATE   | SALESMAN_ID | NAME  |
+------------+-------------+-------+
| 2017-05-04 |        1000 | JOHN  |
| 2017-01-20 |        2000 | RAVI  |
| 2017-02-24 |        2000 | RAVI  |
| 2017-04-13 |        3000 | KUMAR |
| 2017-03-09 |        2000 | RAVI  |
+------------+-------------+-------+
5 rows in set (0.00 sec)
```

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

Use ON DELETE CASCADE at the end of foreign key definitions while creating child table orders and then execute the following:

Use ON DELETE SET NULL at the end of foreign key definitions while creating child table customers and then executes the following:

```
mysql> DELETE FROM SALESMAN
    -> WHERE SALESMAN_ID=1000;
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM SALESMAN;
+-------------+--------+-----------+------------+
| SALESMAN_ID | NAME   | CITY      | COMMISSION |
+-------------+--------+-----------+------------+
|        2000 | RAVI   | BANGALORE | 20 %       |
|        3000 | KUMAR  | MYSORE    | 15 %       |
|        4000 | SMITH  | DELHI     | 30 %       |
|        5000 | HARSHA | HYDRABAD  | 15%        |
+-------------+--------+-----------+------------+
4 rows in set (0.00 sec)
```

3. Consider the schema for MovieDatabase:

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)

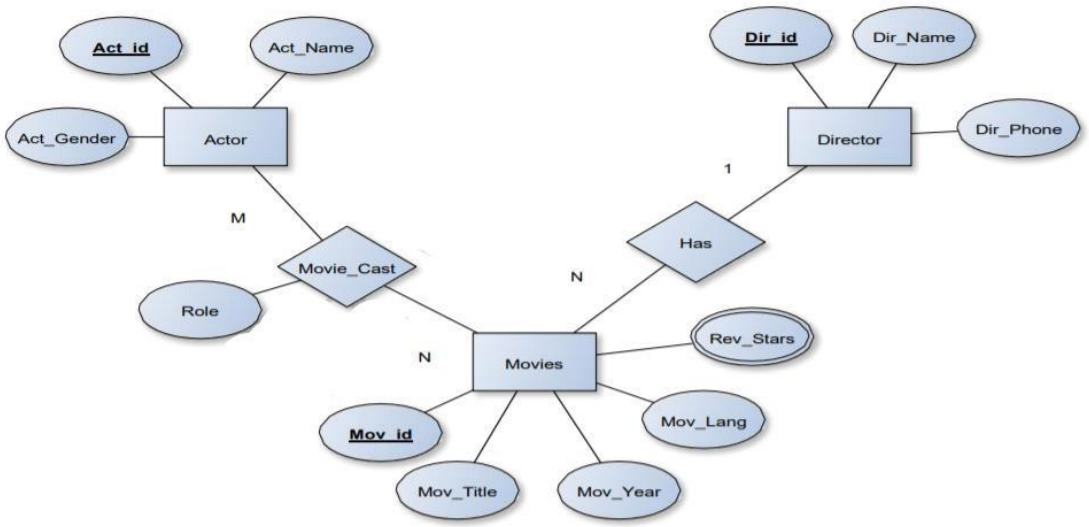
RATING (Mov_id, Rev_Stars)

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.**

Solution:

Entity-Relationship Diagram



Schema Diagram

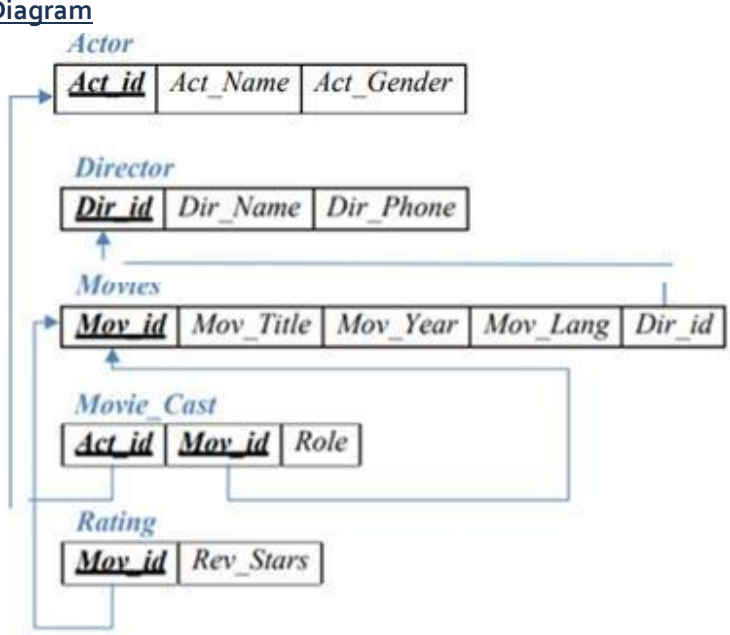


Table Creation

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

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

```
CREATE TABLE MOVIES
( MOV_ID INTEGER (4),
MOV_TITLE VARCHAR (25),
MOV_YEAR INTEGER (4),
MOV_LANG VARCHAR (12),
DIR_ID INTEGER (3),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR (DIR_ID));

CREATE TABLE MOVIE_CAST
( ACT_ID INTEGER (3),
MOV_ID INTEGER (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 INTEGER (4),
REV_STARS VARCHAR (25),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));
```

Table Descriptions

```
mysql> DESC ACTOR;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| ACT_ID     | int(3)      | NO   | PRI | NULL    |       |
| ACT_NAME   | varchar(20) | YES  |     | NULL    |       |
| ACT_GENDER | char(1)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 rows in set (0.02 sec)
```

```
mysql> DESC DIRECTOR;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| DIR_ID    | int(3)      | NO   | PRI | NULL    |       |
| DIR_NAME  | varchar(20) | YES  |     | NULL    |       |
| DIR_PHONE | bigint(10)  | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> DESC MOVIES;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| MOV_ID    | int(4)      | NO   | PRI | NULL    |       |
| MOV_TITLE | varchar(25) | YES  |     | NULL    |       |
| MOV_YEAR  | int(4)      | YES  |     | NULL    |       |
| MOV_LANG  | varchar(12) | YES  |     | NULL    |       |
| DIR_ID    | int(3)      | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> DESC MOVIE_CAST;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| ACT_ID | int(3)      | NO   | PRI | NULL    |       |
| MOV_ID | int(4)      | NO   | PRI | NULL    |       |
| ROLE   | varchar(10) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> DESC RATING;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| MOV_ID    | int(4)      | NO   | PRI | NULL    |       |
| REV_STARS | varchar(25) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.03 sec)
```

Insertion of Values into 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, 'TELAGU', 60);

INSERT INTO MOVIES VALUES (1002,'BAHUBALI-1', 2015, 'TELAGU', 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);

Display all Values

```
mysql> SELECT * FROM ACTOR;
+--------+----------+------------+
| ACT_ID | ACT_NAME | ACT_GENDER |
+--------+----------+------------+
|    301 | ANUSHKA  | F          |
|    302 | PRABHAS  | M          |
|    303 | PUNITH   | M          |
|    304 | JERMY    | M          |
+--------+----------+------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM DIRECTOR;
+--------+------------------+------------+
| DIR_ID | DIR_NAME         | DIR_PHONE  |
+--------+------------------+------------+
|     60 | RAJAMOULI        | 8751611001 |
|     61 | HITCHCOCK        | 7766138911 |
|     62 | FARAN            | 9986776531 |
|     63 | STEVEN SPIELBERG | 8989776530 |
+--------+------------------+------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM MOVIES;
+--------+-----------+----------+----------+--------+
| MOV_ID | MOV_TITLE | MOV_YEAR | MOV_LANG | DIR_ID |
+--------+-----------+----------+----------+--------+
|   1001 | BAHUBALI-2 |     2017 | TELAGU   |     60 |
|   1002 | BAHUBALI-1 |     2015 | TELAGU   |     60 |
|   1003 | AKASH      |     2008 | KANNADA  |     61 |
|   1004 | WAR HORSE  |     2011 | ENGLISH  |     63 |
+--------+-----------+----------+----------+--------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM MOVIE_CAST;
+--------+--------+----------+
| ACT_ID | MOV_ID | ROLE     |
+--------+--------+----------+
|    301 |   1002 | HEROINE  |
|    301 |   1001 | HEROINE  |
|    303 |   1003 | HERO     |
|    303 |   1002 | GUEST    |
|    304 |   1004 | HERO     |
+--------+--------+----------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM RATING;
+--------+-----------+
| MOV_ID | REV_STARS |
+--------+-----------+
|   1001 | 4         |
|   1002 | 2         |
|   1003 | 5         |
|   1004 | 4         |
+--------+-----------+
4 rows in set (0.00 sec)
```

Queries

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

```
mysql> SELECT MOV_TITLE
    -> FROM MOVIES
    -> WHERE DIR_ID IN (SELECT DIR_ID
    -> FROM DIRECTOR
    -> WHERE DIR_NAME = 'HITCHCOCK');
+-----------+
| MOV_TITLE |
+-----------+
| AKASH     |
+-----------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT MOV_TITLE
    -> FROM MOVIES M, MOVIE_CAST MV
    -> WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN (SELECT ACT_ID
    -> FROM MOVIE_CAST GROUP BY ACT_ID
    -> HAVING COUNT(ACT_ID) > 1)
    -> GROUP BY MOV_TITLE
    -> HAVING COUNT(*) > 1;
+------------+
| MOV_TITLE  |
+------------+
| BAHUBALI-1 |
+------------+
1 row in set (0.04 sec)
```

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

```
mysql> SELECT ACT_NAME, MOV_TITLE, MOV_YEAR
    -> FROM ACTOR A
    -> JOIN MOVIE_CAST C
    -> ON A.ACT_ID=C.ACT_ID
    -> JOIN MOVIES M
    -> ON C.MOV_ID=M.MOV_ID
    -> WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
+----------+------------+----------+
| ACT_NAME | MOV_TITLE  | MOV_YEAR |
+----------+------------+----------+
| ANUSHKA  | BAHUBALI-2 |     2017 |
+----------+------------+----------+
1 row in set (0.02 sec)
```

OR

```
mysql> SELECT A.ACT_NAME, A.ACT_NAME, C.MOV_TITLE, C.MOV_YEAR
    -> FROM ACTOR A, MOVIE_CAST B, MOVIES C
    -> WHERE A.ACT_ID=B.ACT_ID
    -> AND B.MOV_ID=C.MOV_ID
    -> AND C.MOV_YEAR NOT BETWEEN 2000 AND 2015;
+----------+----------+------------+----------+
| ACT_NAME | ACT_NAME | MOV_TITLE  | MOV_YEAR |
+----------+----------+------------+----------+
| ANUSHKA  | ANUSHKA  | BAHUBALI-2 |     2017 |
+----------+----------+------------+----------+
1 row in set (0.00 sec)
```

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

```
mysql> SELECT MOV_TITLE, MAX(REV_STARS)
    -> FROM MOVIES
    -> INNER JOIN RATING USING (MOV_ID)
    -> GROUP BY MOV_TITLE
    -> HAVING MAX(REV_STARS) > 0
    -> ORDER BY MOV_TITLE;
+-------------+-----------------+
| MOV_TITLE   | MAX(REV_STARS) |
+-------------+-----------------+
| AKASH       | 5               |
| BAHUBALI-1 | 2               |
| BAHUBALI-2 | 4               |
| WAR HORSE   | 4               |
+-------------+-----------------+
4 rows in set (0.04 sec)
```

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

```
mysql> UPDATE RATING
    -> SET REV_STARS=5
    -> WHERE MOV_ID IN (SELECT MOV_ID FROM MOVIES
    -> WHERE DIR_ID IN (SELECT DIR_ID
    -> FROM DIRECTOR
    -> WHERE DIR_NAME = 'STEVEN SPIELBERG'));
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM RATING;
+--------+-----------+
| MOV_ID | REV_STARS |
+--------+-----------+
|   1001 | 4         |
|   1002 | 2         |
|   1003 | 5         |
|   1004 | 5         |
+--------+-----------+
4 rows in set (0.00 sec)
```

4. Consider the schema for CollegeDatabase:

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 '1DS15MCA101' 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'

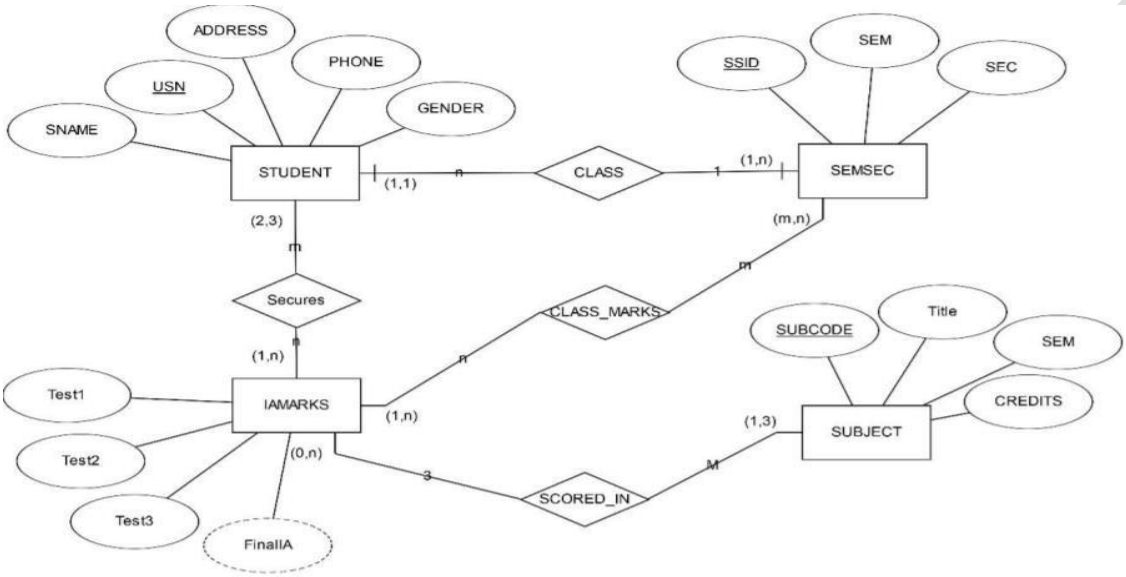
   If FinalIA< 12 then

   CAT = 'Weak'

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

Solution:

Entity-Relationship Diagram



Schema Diagram

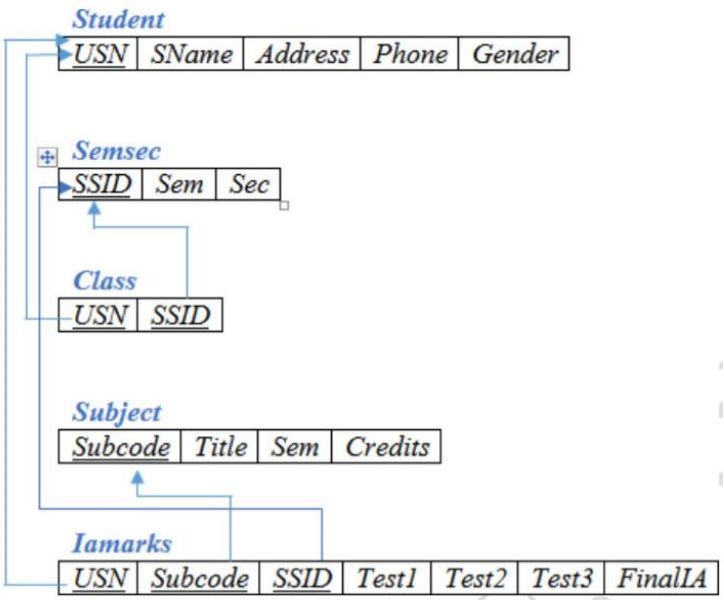


Table Creation

```
CREATE TABLE STUDENT
( USN VARCHAR (11) 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 (11),
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),
TITLE VARCHAR (20),
SEM INT (2),
CREDITS INT (2),
PRIMARY KEY (SUBCODE));

CREATE TABLE IAMARKS
( USN VARCHAR (11),
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));

Table Descriptions

```
mysql> DESC STUDENT;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| USN     | varchar(11) | NO   | PRI | NULL    |       |
| SNAME   | varchar(25) | YES  |     | NULL    |       |
| ADDRESS | varchar(25) | YES  |     | NULL    |       |
| PHONE   | bigint(10)  | YES  |     | NULL    |       |
| GENDER  | char(1)     | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
ysql> DESC SEMSEC;
-------+------------+------+-----+---------+-------+
 Field | Type       | Null | Key | Default | Extra |
-------+------------+------+-----+---------+-------+
 SSID  | varchar(5) | NO   | PRI | NULL    |       |
 SEM   | int(2)     | YES  |     | NULL    |       |
 SEC   | char(1)    | YES  |     | NULL    |       |
-------+------------+------+-----+---------+-------+
 rows in set (0.00 sec)
```

```
mysql> DESC CLASS;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| USN   | varchar(11) | NO   | PRI | NULL    |       |
| SSID  | varchar(5)  | NO   | PRI | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
ysql> DESC SUBJECT;
---------+-------------+------+-----+---------+-------+
 Field   | Type        | Null | Key | Default | Extra |
---------+-------------+------+-----+---------+-------+
 SUBCODE | varchar(8)  | NO   | PRI | NULL    |       |
 TITLE   | varchar(20) | YES  |     | NULL    |       |
 SEM     | int(2)      | YES  |     | NULL    |       |
 CREDITS | int(2)      | YES  |     | NULL    |       |
---------+-------------+------+-----+---------+-------+
 rows in set (0.00 sec)
```

```
mysql> DESC IAMARKS;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| USN     | varchar(11) | NO   | PRI | NULL    |       |
| SUBCODE | varchar(8)  | NO   | PRI | NULL    |       |
| SSID    | varchar(5)  | NO   | PRI | NULL    |       |
| TEST1   | int(2)      | YES  |     | NULL    |       |
| TEST2   | int(2)      | YES  |     | NULL    |       |
| TEST3   | int(2)      | YES  |     | NULL    |       |
| FINALIA | int(2)      | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

Insertion of Values into Tables

```
INSERT INTO STUDENT VALUES ('1DS13MCA020','AKSHAY','BELAGAVI', 8877881122,'M');
INSERT INTO STUDENT VALUES ('1DS13MCA062','SANDHYA','BENGALURU', 7722829912,'F');
INSERT INTO STUDENT VALUES ('1DS13MCA091','TEESHA','BENGALURU', 7712312312,'F');
INSERT INTO STUDENT VALUES ('1DS13MCA066','SUPRIYA','MANGALURU', 8877881122,'F');
INSERT INTO STUDENT VALUES ('1DS14MCA010','ABHAY','BENGALURU', 9900211201,'M');
INSERT INTO STUDENT VALUES ('1DS14MCA032','BHASKAR','BENGALURU',
9923211099,'M'); INSERT INTO STUDENT VALUES ('1DS14MCA025','ASMI','BENGALURU',
7894737377,'F'); INSERT INTO STUDENT VALUES ('1DS15MCA011','AJAY','TUMKUR',
9845091341,'M');
INSERT INTO STUDENT VALUES ('1DS15MCA029','CHITRA','DAVANGERE', 7696772121,'F');
INSERT INTO STUDENT VALUES ('1DS15MCA045','JEEVA','BELLARY', 9944850121,'M');
INSERT INTO STUDENT VALUES ('1DS15MCA091','SANTOSH','MANGALURU', 8812332201,'M');
INSERT INTO STUDENT VALUES ('1DS16MCA045','ISMAIL','KALBURGI', 9900232201,'M');
INSERT INTO STUDENT VALUES ('1DS16MCA088','SAMEERA','SHIMOGA', 9905542212,'F');
INSERT INTO STUDENT VALUES ('1DS16MCA122','VINAYAKA','CHIKAMAGALUR', 8800880011,'M');

INSERT INTO SEMSEC VALUES ('MCA4A', 4,'A');
INSERT INTO SEMSEC VALUES ('MCA4B', 4,'B');
INSERT INTO SEMSEC VALUES ('MCA4C', 4,'C');
INSERT INTO SEMSEC VALUES ('MCA3A', 3,'A');
INSERT INTO SEMSEC VALUES ('MCA3B', 3,'B');
INSERT INTO SEMSEC VALUES ('MCA3C', 3,'C');
INSERT INTO SEMSEC VALUES ('MCA2A', 2,'A');
INSERT INTO SEMSEC VALUES ('MCA2B', 2,'B');
INSERT INTO SEMSEC VALUES ('MCA2C', 2,'C');
INSERT INTO SEMSEC VALUES ('MCA1A', 1,'A');
INSERT INTO SEMSEC VALUES ('MCA1B', 1,'B');
INSERT INTO SEMSEC VALUES ('MCA1C', 1,'C');

INSERT INTO CLASS VALUES ('1DS15MCA011','MCA4A');
INSERT INTO CLASS VALUES ('1DS15MCA029','MCA4A');
INSERT INTO CLASS VALUES ('1DS15MCA045','MCA4B');
```

INSERT INTO CLASS VALUES ('1DS15MCA091','MCA4C');

INSERT INTO CLASS VALUES ('1DS16MCA045','MCA3A'); INSERT INTO CLASS VALUES ('1DS16MCA088','MCA3B'); INSERT INTO CLASS VALUES ('1DS16MCA122','MCA3C');

INSERT INTO SUBJECT VALUES ('15MCA41','M4', 4, 4);
INSERT INTO SUBJECT VALUES ('15MCA42','SE', 4, 4);
INSERT INTO SUBJECT VALUES ('15MCA43','DAA', 4, 4);
INSERT INTO SUBJECT VALUES ('15MCA44','MPMC', 4, 4);
INSERT INTO SUBJECT VALUES ('15MCA45','OOC', 4, 3);
INSERT INTO SUBJECT VALUES ('15MCA46','DC', 4, 3);
INSERT INTO SUBJECT VALUES ('15MCA31','M3', 3, 4);
INSERT INTO SUBJECT VALUES ('15MCA32','ADE', 3, 4);
INSERT INTO SUBJECT VALUES ('15MCA33','DSA', 3, 4);
INSERT INTO SUBJECT VALUES ('15MCA34','CO', 3, 4);
INSERT INTO SUBJECT VALUES ('15MCA35','USP', 3, 3);
INSERT INTO SUBJECT VALUES ('15MCA36','DMS', 3, 3);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1DS15MCA091','15MCA41','MCA4C', 15, 16, 18); INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1DS15MCA091','15MCA42','MCA4C', 12, 19, 14); INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1DS15MCA091','15MCA43','MCA4C', 19, 15, 20); INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1DS15MCA091','15MCA44','MCA4C', 20, 16, 19); INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1DS15MCA091','15MCA45','MCA4C', 15, 15, 12);

Display all Values

```
mysql> SELECT * FROM STUDENT;
+-------------+----------+--------------+------------+--------+
| USN         | SNAME    | ADDRESS      | PHONE      | GENDER |
+-------------+----------+--------------+------------+--------+
| 1DS13MCA020 | AKSHAY   | BELAGAVI     | 8877881122 | M      |
| 1DS13MCA062 | SANDHYA  | BENGALURU    | 7722829912 | F      |
| 1DS13MCA091 | TEESHA   | BENGALURU    | 7712312312 | F      |
| 1DS13MCA066 | SUPRIYA  | MANGALURU    | 8877881122 | F      |
| 1DS14MCA010 | ABHAY    | BENGALURU    | 9900211201 | M      |
| 1DS14MCA032 | BHASKAR  | BENGALURU    | 9923211099 | M      |
| 1DS14MCA025 | ASMI     | BENGALURU    | 7894737377 | F      |
| 1DS15MCA011 | AJAY     | TUMKUR       | 9845091341 | M      |
| 1DS15MCA029 | CHITRA   | DAVANGERE    | 7696772121 | F      |
| 1DS15MCA045 | JEEVA    | BELLARY      | 9944850121 | M      |
| 1DS15MCA091 | SANTOSH  | MANGALURU    | 8812332201 | M      |
| 1DS16MCA045 | ISMAIL   | KALBURGI     | 9900232201 | M      |
| 1DS16MCA088 | SAMEERA  | SHIMOGA      | 9905542212 | F      |
| 1DS16MCA122 | VINAYAKA | CHIKAMAGALUR | 8800880011 | M      |
+-------------+----------+--------------+------------+--------+
14 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM SEMSEC;
+-------+------+------+
| SSID  | SEM  | SEC  |
+-------+------+------+
| MCA4A |    4 | A    |
| MCA4B |    4 | B    |
| MCA4C |    4 | C    |
| MCA3A |    3 | A    |
| MCA3B |    3 | B    |
| MCA3C |    3 | C    |
| MCA2A |    2 | A    |
| MCA2B |    2 | B    |
| MCA2C |    2 | C    |
| MCA1A |    1 | A    |
| MCA1B |    1 | B    |
| MCA1C |    1 | C    |
+-------+------+------+
12 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM CLASS;
+-------------+-------+
| USN         | SSID  |
+-------------+-------+
| 1DS15MCA011 | MCA4A |
| 1DS15MCA029 | MCA4A |
| 1DS15MCA045 | MCA4B |
| 1DS15MCA091 | MCA4C |
| 1DS16MCA045 | MCA3A |
| 1DS16MCA088 | MCA3B |
| 1DS16MCA122 | MCA3C |
+-------------+-------+
7 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM SUBJECT;
+---------+-------+------+---------+
| SUBCODE | TITLE | SEM  | CREDITS |
+---------+-------+------+---------+
| 15MCA41 | M4    |    4 |       4 |
| 15MCA42 | SE    |    4 |       4 |
| 15MCA43 | DAA   |    4 |       4 |
| 15MCA44 | MPMC  |    4 |       4 |
| 15MCA45 | OOC   |    4 |       3 |
| 15MCA46 | DC    |    4 |       3 |
| 15MCA31 | M3    |    3 |       4 |
| 15MCA32 | ADE   |    3 |       4 |
| 15MCA33 | DSA   |    3 |       4 |
| 15MCA34 | CO    |    3 |       4 |
| 15MCA35 | USP   |    3 |       3 |
| 15MCA36 | DMS   |    3 |       3 |
+---------+-------+------+---------+
12 rows in set (0.00 sec)
```

```
mysql> select * from iamarks;
+-------------+---------+-------+-------+-------+-------+----------+
| USN         | SUBCODE | SSID  | TEST1 | TEST2 | TEST3 | FINALIA  |
+-------------+---------+-------+-------+-------+-------+----------+
| 1DS15MCA091 | 15MCA41 | MCA4C |    15 |    16 |    18 |     NULL |
| 1DS15MCA091 | 15MCA42 | MCA4C |    12 |    19 |    14 |     NULL |
| 1DS15MCA091 | 15MCA43 | MCA4C |    19 |    15 |    20 |     NULL |
| 1DS15MCA091 | 15MCA44 | MCA4C |    20 |    16 |    19 |     NULL |
| 1DS15MCA091 | 15MCA45 | MCA4C |    15 |    15 |    12 |     NULL |
+-------------+---------+-------+-------+-------+-------+----------+
5 rows in set (0.00 sec)
```

Queries

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

```
mysql> 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';
+-------------+---------+-----------+------------+--------+------+------+
| USN         | SNAME   | ADDRESS   | PHONE      | GENDER | SEM  | SEC  |
+-------------+---------+-----------+------------+--------+------+------+
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      |    4 | C    |
+-------------+---------+-----------+------------+--------+------+------+
1 row in set (0.00 sec)
```

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

```
mysql> 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 SEM;
+------+------+--------+-------+
| SEM  | SEC  | GENDER | COUNT |
+------+------+--------+-------+
|    3 | A    | M      |     1 |
|    3 | B    | F      |     1 |
|    3 | C    | M      |     1 |
|    4 | A    | F      |     1 |
|    4 | A    | M      |     1 |
|    4 | B    | M      |     1 |
|    4 | C    | M      |     1 |
+------+------+--------+-------+
7 rows in set (0.00 sec)
```

3. Create a view of Test1 marks of student USN '1DS15MCA091' in all subjects.

```
mysql> CREATE VIEW STU_TEST1_MARKS_VIEW
    -> AS
    -> SELECT TEST1, SUBCODE
    -> FROM IAMARKS
    -> WHERE USN = '1DS15MCA091';
Query OK, 0 rows affected (0.05 sec)

mysql> SELECT * FROM STU_TEST1_MARKS_VIEW;
+-------+---------+
| TEST1 | SUBCODE |
+-------+---------+
|    15 | 10MCA41 |
|    12 | 10MCA42 |
|    19 | 10MCA43 |
|    20 | 10MCA44 |
|    15 | 10MCA45 |
+-------+---------+
5 rows in set (0.00 sec)
```

4. Calculate the FinalIA (average of test marks) and update the corresponding table for all students.

```
mysql> select * from iamarks;
+-------------+---------+-------+-------+-------+-------+---------+
| USN         | SUBCODE | SSID  | TEST1 | TEST2 | TEST3 | FINALIA |
+-------------+---------+-------+-------+-------+-------+---------+
| 1DS15MCA091 | 15MCA41 | MCA4C |    15 |    16 |    18 |      16 |
| 1DS15MCA091 | 15MCA42 | MCA4C |    12 |    19 |    14 |      15 |
| 1DS15MCA091 | 15MCA43 | MCA4C |    19 |    15 |    20 |      18 |
| 1DS15MCA091 | 15MCA44 | MCA4C |    20 |    16 |    19 |      18 |
| 1DS15MCA091 | 15MCA45 | MCA4C |    15 |    15 |    12 |      14 |
+-------------+---------+-------+-------+-------+-------+---------+
5 rows in set (0.00 sec)
```

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.

```
mysql>  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 = 4;
+-------------+---------+-----------+------------+--------+-------------+
| USN         | SNAME   | ADDRESS   | PHONE      | GENDER | CAT         |
+-------------+---------+-----------+------------+--------+-------------+
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      | AVERAGE     |
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      | AVERAGE     |
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      | OUTSTANDING |
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      | OUTSTANDING |
| 1DS15MCA091 | SANTOSH | MANGALURU | 8812332201 | M      | AVERAGE     |
+-------------+---------+-----------+------------+--------+-------------+
5 rows in set (0.00 sec)
```

5. Consider the schema for Company Database:

EMPLOYEE (SSN, Name, Address, Gender, Salary, SuperSSN, DNo)

DEPARTMENT (DNo, DName, MgrSSN, MgrStartDate)

DLOCATION (DNo,DLoc)

PROJECT (PNo, PName, PLocation, DNo)

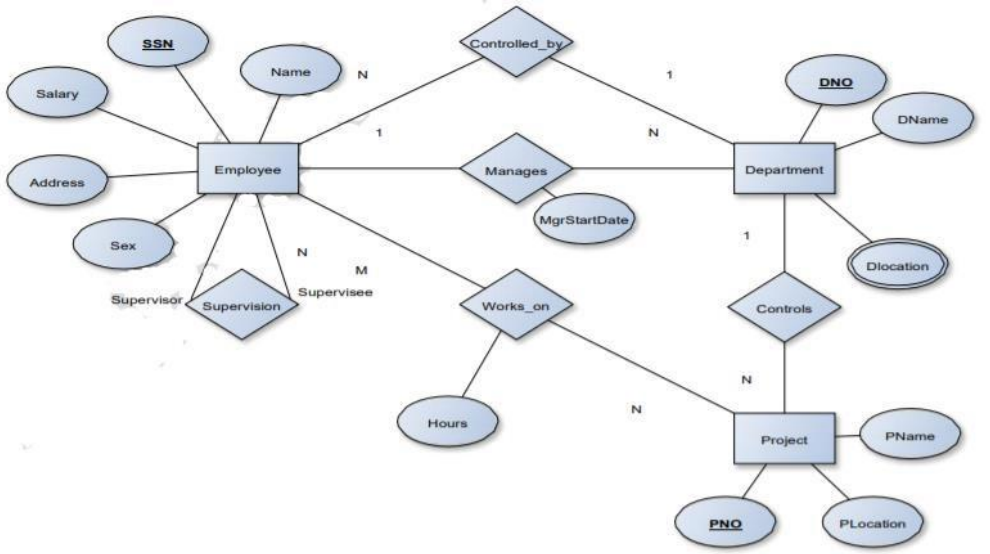
WORKS_ON (SSN, PNo, Hours)

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 controlled by department number 5 (use NOT EXISTS operator). 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:

Entity-Relationship Diagram



Schema Diagram

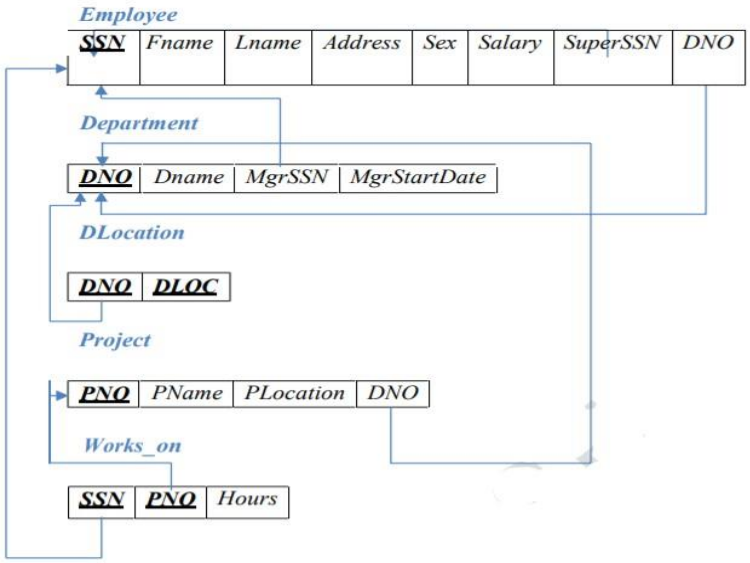


Table Creation

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

```
CREATE TABLE EMPLOYEE
(SSN VARCHAR (20) PRIMARY KEY,
FNAME VARCHAR (20),
LNAME VARCHAR (20),
ADDRESS VARCHAR (20),
GENDER CHAR (1),
SALARY INTEGER,
SUPERSSN VARCHAR
(20),
FOREIGN KEY(SUPERSSN) REFERENCES EMPLOYEE (SSN),
DNO VARCHAR (20),
FOREIGN KEY(DNO) REFERENCES DEPARTMENT (DNO));
```

NOTE: Once DEPARTMENT and EMPLOYEE tables are created, we must alter department table to add foreign constraint MGRSSN using SQL command

```
ALTER TABLE DEPARTMENT
ADD CONSTRAINT FOREIGN KEY(MGRSSN) REFERENCES EMPLOYEE (SSN);
```

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

```
CREATE TABLE PROJECT
(PNO INTEGER PRIMARY KEY,
PNAME VARCHAR (20),
PLOCATION VARCHAR (20),
DNO VARCHAR(20),
```

```
FOREIGN KEY(DNO) REFERENCES DEPARTMENT (DNO));
```

CREATE TABLE
WORKS_ON (HOURS
INTEGER (2),
SSN VARCHAR (20),
FOREIGN KEY(SSN) REFERENCES EMPLOYEE
(SSN),PNO VARCHAR (20),
FOREIGN KEY(PNO) REFERENCES PROJECT(PNO),
PRIMARY KEY (SSN, PNO));

Table Descriptions

```
mysql> DESC EMPLOYEE;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| SSN      | varchar(20) | NO   | PRI | NULL    |       |
| FNAME    | varchar(20) | YES  |     | NULL    |       |
| LNAME    | varchar(20) | YES  |     | NULL    |       |
| ADDRESS  | varchar(20) | YES  |     | NULL    |       |
| GENDER   | char(1)     | YES  |     | NULL    |       |
| SALARY   | int(11)     | YES  |     | NULL    |       |
| SUPERSSN | varchar(20) | YES  | MUL | NULL    |       |
| DNO      | varchar(20) | YES  | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

```
mysql> DESC DEPARTMENT;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| DNO          | varchar(20) | NO   | PRI | NULL    |       |
| DNAME        | varchar(20) | YES  |     | NULL    |       |
| MGRSTARTDATE | date        | YES  |     | NULL    |       |
| MGRSSN       | varchar(20) | YES  | MUL | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> DESC DLOCATION;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| DLOC  | varchar(20) | NO   | PRI | NULL    |       |
| DNO   | varchar(20) | NO   | PRI | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

```
mysql> DESC PROJECT;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| PNO       | int(11)     | NO   | PRI | NULL    |       |
| PNAME     | varchar(20) | YES  |     | NULL    |       |
| PLOCATION | varchar(20) | YES  |     | NULL    |       |
| DNO       | varchar(20) | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.03 sec)
```

```
mysql> DESC WORKS_ON;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| HOURS | int(2)      | YES  |     | NULL    |       |
| SSN   | varchar(20) | NO   | PRI | NULL    |       |
| PNO   | varchar(20) | NO   | PRI | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.04 sec)
```

Insertion of Values into Tables

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEECE01','JOHN','SCOTT','BANGALORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA01','JAMES','SMITH','BANGALORE','M', 500000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA02','HEARN','BAKER','BANGALORE','M', 700000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA03','EDWARD','SCOTT','MYSORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA04','PAVAN','HEGDE','MANGALORE','M', 650000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA05','GIRISH','MALYA','MYSORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEMCA06','NEHA','SN','BANGALORE','F', 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEACC01','AHANA','K','MANGALORE','F', 350000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEACC02','SANTHOSH','KUMAR','MANGALORE','M', 300000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEISE01','VEENA','M','MYSORE','M', 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY) VALUES ('DSCEIT01','NAGESH','HR','BANGALORE','M', 500000);

INSERT INTO DEPARTMENT VALUES ('1','ACCOUNTS','2021-01-01','DSCEACC02');
INSERT INTO DEPARTMENT VALUES ('2','IT','2016-08-01','DSCEIT01');
INSERT INTO DEPARTMENT VALUES ('3','ECE','2018-06-01','DSCEECE01');
INSERT INTO DEPARTMENT VALUES ('4','ISE','2015-08-01','DSCEISE01');
INSERT INTO DEPARTMENT VALUES ('5','MCA','2002-06-01','DSCEMCA05');

Note: update entries of employee table to fill missing fields SUPERSSN and DNO

UPDATE EMPLOYEE SET SUPERSSN=NULL, DNO='3' WHERE SSN='DSCEECE01';
UPDATE EMPLOYEE SET SUPERSSN='DSCEMCA02', DNO='5' WHERE SSN='DSCEMCA01'; UPDATE EMPLOYEE SET SUPERSSN='DSCEMCA03', DNO='5' WHERE SSN='DSCEMCA02';
UPDATE EMPLOYEE SET SUPERSSN='DSCEMCA04', DNO='5' WHERE SSN='DSCEMCA03';UPDATE EMPLOYEE SET DNO='5', SUPERSSN='DSCEMCA05' WHERE SSN='DSCEMCA04';UPDATE EMPLOYEE SET DNO='5', SUPERSSN='DSCEMCA06' WHERE SSN='DSCEMCA05'; UPDATE EMPLOYEE SET

DNO='5', SUPERSSN=NULL WHERE SSN='DSCEMCA06'; UPDATE EMPLOYEE SET
DNO='1', SUPERSSN='DSCEACC02' WHERE SSN='DSCEACC01';

```
UPDATE EMPLOYEE SET DNO='1', SUPERSSN=NULL WHERE SSN='DSCEACC02'; UPDATE EMPLOYEE SET DNO='4', SUPERSSN=NULL WHERE SSN='DSCEISE01'; UPDATE EMPLOYEE SET DNO='2', SUPERSSN=NULL WHERE SSN='DSCEIT01';

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,'SMART CITY','BANGALORE','2');

INSERT INTO WORKS_ON VALUES (4, 'DSCEMCA01', 100);
INSERT INTO WORKS_ON VALUES (6, 'DSCEMCA01', 101);
INSERT INTO WORKS_ON VALUES (8, 'DSCEMCA01', 102);
INSERT INTO WORKS_ON VALUES (10, 'DSCEMCA02',
100); INSERT INTO WORKS_ON VALUES (3, 'DSCEMCA04',
100); INSERT INTO WORKS_ON VALUES (4, 'DSCEMCA05',
101); INSERT INTO WORKS_ON VALUES (5, 'DSCEMCA06',
102); INSERT INTO WORKS_ON VALUES (6, 'DSCEMCA03',
102); INSERT INTO WORKS_ON VALUES (7, 'DSCEECE01',
103); INSERT INTO WORKS_ON VALUES (5, 'DSCEACC01',
104); INSERT INTO WORKS_ON VALUES (6, 'DSCEACC02',
105); INSERT INTO WORKS_ON VALUES (4, 'DSCEISE01',
106); INSERT INTO WORKS_ON VALUES (10, 'DSCEIT01',
107);
```

Display all Values

```
mysql> SELECT * FROM EMPLOYEE;
+-----------+----------+---------+-----------+--------+--------+-----------+------+
| SSN       | FNAME    | LNAME   | ADDRESS   | GENDER | SALARY | SUPERSSN  | DNO  |
+-----------+----------+---------+-----------+--------+--------+-----------+------+
| DSCEECE01 | JOHN     | SCOTT   | BANGALORE | M      | 450000 | NULL      | 3    |
| DSCEMCA01 | JAMES    | SCSMITH | BANGALORE | M      | 500000 | DSCEMCA02 | 5    |
| DSCEMCA02 | HEARN    | BAKER   | BANGALORE | M      | 700000 | DSCEMCA03 | 5    |
| DSCEMCA03 | EDWARD   | SCOTT   | MYSORE    | M      | 500000 | DSCEMCA04 | 5    |
| DSCEMCA04 | PAVAN    | HEGDE   | MANGALORE | M      | 650000 | DSCEMCA05 | 5    |
| DSCEMCA05 | GIRISH   | MALYA   | MYSORE    | M      | 450000 | DSCEMCA06 | 5    |
| DSCEMCA06 | NEHA     | SN      | BANGALORE | F      | 800000 | NULL      | 5    |
| DSCEACC01 | AHANA    | K       | MANGALORE | F      | 350000 | DSCEACC02 | 1    |
| DSCEACC02 | SANTHOSH | KUMAR   | MANGALORE | M      | 300000 | NULL      | 1    |
| DSCEISE01 | VEENA    | M       | MYSORE    | M      | 600000 | NULL      | 4    |
| DSCEIT01  | NAGESH   | HR      | BANGALORE | M      | 500000 | NULL      | 2    |
+-----------+----------+---------+-----------+--------+--------+-----------+------+
11 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM DEPARTMENT;
+-----+----------+--------------+-----------+
| DNO | DNAME    | MGRSTARTDATE | MGRSSN    |
+-----+----------+--------------+-----------+
| 1   | ACCOUNTS | 2021-01-01   | DSCEACC02 |
| 2   | IT       | 2016-08-01   | DSCEIT01  |
| 3   | ECE      | 2018-06-01   | DSCEECE01 |
| 4   | ISE      | 2015-08-01   | DSCEISE01 |
| 5   | MCA      | 2002-06-01   | DSCEMCA05 |
+-----+----------+--------------+-----------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM DLOCATION;
+-----------+-----+
| DLOC      | DNO |
+-----------+-----+
| BANGALORE | 1   |
| BANGALORE | 2   |
| BANGALORE | 3   |
| MANGALORE | 4   |
| MANGALORE | 5   |
+-----------+-----+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM PROJECT;
+-----+------------------+-----------+------+
| PNO | PNAME            | PLOCATION | DNO  |
+-----+------------------+-----------+------+
| 100 | IOT              | BANGALORE | 5    |
| 101 | CLOUD            | BANGALORE | 5    |
| 102 | BIGDATA          | BANGALORE | 5    |
| 103 | SENSORS          | BANGALORE | 3    |
| 104 | BANK MANAGEMENT  | BANGALORE | 1    |
| 105 | SALARY MANAGEMENT| BANGALORE | 1    |
| 106 | OPENSTACK        | BANGALORE | 4    |
| 107 | SMART CITY       | BANGALORE | 2    |
+-----+------------------+-----------+------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM WORKS_ON;
+-------+-----------+-----+
| HOURS | SSN       | PNO |
+-------+-----------+-----+
|     4 | DSCEMCA01 | 100 |
|     6 | DSCEMCA01 | 101 |
|     8 | DSCEMCA01 | 102 |
|    10 | DSCEMCA02 | 100 |
|     3 | DSCEMCA04 | 100 |
|     4 | DSCEMCA05 | 101 |
|     5 | DSCEMCA06 | 102 |
|     6 | DSCEMCA03 | 102 |
|     7 | DSCEECE01 | 103 |
|     5 | DSCEACC01 | 104 |
|     6 | DSCEACC02 | 105 |
|     4 | DSCEISE01 | 106 |
|    10 | DSCEIT01  | 107 |
+-------+-----------+-----+
13 rows in set (0.00 sec)
```

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.

```
mysql> (SELECT DISTINCT P.PNO FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
    -> WHERE E.DNO=D.DNO AND D.MGRSSN=E.SSN AND E.LNAME='SCOTT')
    -> UNION
    ->  (SELECT DISTINCT P1.PNO FROM PROJECT P1, WORKS_ON W, EMPLOYEE E1
    -> WHERE P1.PNO=W.PNO AND E1.SSN=W.SSN AND E1.LNAME='SCOTT');
+-----+
| PNO |
+-----+
| 100 |
| 101 |
| 102 |
| 103 |
| 104 |
| 105 |
| 106 |
| 107 |
+-----+
8 rows in set (0.00 sec)
```

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

```
mysql> SELECT 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';
+-------+---------+----------+
| FNAME | LNAME   | INCR_SAL |
+-------+---------+----------+
| JAMES | SCSMITH | 550000.0 |
| HEARN | BAKER   | 770000.0 |
| PAVAN | HEGDE   | 715000.0 |
+-------+---------+----------+
3 rows in set (0.00 sec)
```

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.

```
mysql> 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';
+---------------+---------------+---------------+---------------+
| SUM(E.SALARY) | MAX(E.SALARY) | MIN(E.SALARY) | AVG(E.SALARY) |
+---------------+---------------+---------------+---------------+
|        650000 |        350000 |        300000 |   325000.0000 |
+---------------+---------------+---------------+---------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT E.FNAME, E.LNAME FROM EMPLOYEE E
    -> WHERE NOT EXISTS
    -> (SELECT PNO FROM PROJECT WHERE DNO='5'
    -> AND PNO NOT IN(SELECT PNO FROM WORKS_ON WHERE E.SSN=SSN));
+--------+----------+
| FNAME  | LNAME    |
+--------+----------+
| JAMES  | SCSMITH  |
+--------+----------+
1 row in set (0.00 sec)
```

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.

```
mysql> 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;
+-----+----------+
| DNO | COUNT(*) |
+-----+----------+
| 5   |        3 |
+-----+----------+
1 row in set (0.00 sec)
```

Part B: PL/SQL

1. Demonstrate creating a simple PL/SQL procedure which will compute and return the student details who have marks greater than 70 and also return the count.

```
mysql> create table student_info(usn varchar(10) primary key, name varchar(25), subject varchar(25), marks integer(4));
Query OK, 0 rows affected (0.04 sec)

mysql> insert into student_info values('1DS18MCA01','RAJ','C++',90);
Query OK, 1 row affected (0.03 sec)

mysql> insert into student_info values('1DS18MCA02','RAJ','STATISTICS',75);
Query OK, 1 row affected (0.00 sec)

mysql> insert into student_info values('1DS18MCA03','ANKITH','DBMS',64);
Query OK, 1 row affected (0.00 sec)

mysql> insert into student_info values('1DS18MCA04','PRIYA','DBMS',86);
Query OK, 1 row affected (0.00 sec)

mysql> insert into student_info values('1DS18MCA05','LAKSHMI','STATISTICS',69);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM STUDENT_INFO;
+------------+---------+------------+-------+
| usn        | name    | subject    | marks |
+------------+---------+------------+-------+
| 1DS18MCA01 | RAJ     | C++        |    90 |
| 1DS18MCA02 | RAJ     | STATISTICS |    75 |
| 1DS18MCA03 | ANKITH  | DBMS       |    64 |
| 1DS18MCA04 | PRIYA   | DBMS       |    86 |
| 1DS18MCA05 | LAKSHMI | STATISTICS |    69 |
+------------+---------+------------+-------+
5 rows in set (0.00 sec)
```

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE get_merit_student ()
    -> BEGIN
    ->  SELECT * FROM STUDENT_INFO WHERE MARKS>70;
    ->  SELECT COUNT(USN) AS TOTAL_STUDENT FROM STUDENT_INFO;
    -> END
    -> //
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> CALL get_merit_student();
+------------+-------+------------+-------+
| usn        | name  | subject    | marks |
+------------+-------+------------+-------+
| 1DS18MCA01 | RAJ   | C++        |    90 |
| 1DS18MCA02 | RAJ   | STATISTICS |    75 |
| 1DS18MCA04 | PRIYA | DBMS       |    86 |
+------------+-------+------------+-------+
3 rows in set (0.00 sec)

+---------------+
| TOTAL_STUDENT |
+---------------+
|             5 |
+---------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
```

After showing the execution, Stored procedure can be deleted by the following query:

```
mysql> drop procedure get_merit_student;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

2. Create a function that returns the customer occupation based on the age using the below statement.

```
mysql>  create table customer_details(cust_id integer primary key, name varchar(25), occupation varchar(
25), age integer);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into customer_details values(101,'peter','Engineer',32);
Query OK, 1 row affected (0.04 sec)

mysql> insert into customer_details values(102,'Rama','Developer',30);
Query OK, 1 row affected (0.00 sec)

mysql> insert into customer_details values(103,'Sudha','Leader',28);
Query OK, 1 row affected (0.00 sec)

mysql> insert into customer_details values(104,'radhika','Scientist',45);
Query OK, 1 row affected (0.00 sec)

mysql> insert into customer_details values(105,'prakash','Carpenter',26);
Query OK, 1 row affected (0.00 sec)

mysql> insert into customer_details values(106,'raju','Actor',25);
Query OK, 1 row affected (0.02 sec)

mysql> select * from customer_details;
+---------+---------+------------+------+
| cust_id | name    | occupation | age  |
+---------+---------+------------+------+
|     101 | peter   | Engineer   |   32 |
|     102 | Rama    | Developer  |   30 |
|     103 | Sudha   | Leader     |   28 |
|     104 | radhika | Scientist  |   45 |
|     105 | prakash | Carpenter  |   26 |
|     106 | raju    | Actor      |   25 |
+---------+---------+------------+------+
6 rows in set (0.00 sec)

mysql>
```

```
mysql> delimiter //
mysql> create function customer_occupation(age integer)
    -> returns varchar(20)
    -> deterministic
    -> begin
    ->  DECLARE customer_occupation varchar(20);
    ->  IF age>35 THEN
    ->          SET customer_occupation='Scientist';
    ->  ELSEIF (age<=35 AND age>=30) THEN
    ->          SET customer_occupation='Engineer';
    ->  ELSEIF age<30 THEN
    ->          SET customer_occupation='Actor';
    ->  END IF;
    -> -- RETURN THE CUSTOMER OCCUPATION
    -> RETURN (customer_occupation);
    -> end
    -> //
Query OK, 0 rows affected (0.03 sec)

mysql> delimiter ;
mysql>
```

To Display all the available functions in the database:

```
mysql> SHOW FUNCTION STATUS WHERE DB='chandrika';
+-----------+--------------------+----------+----------------+---------------------+---------------------+---------------+---------+----------------------+----------------------+--------------------+
| Db        | Name               | Type     | Definer        | Modified            | Created             | Security_type | Comment | character_set_client | collation_connection | Database Collation |
+-----------+--------------------+----------+----------------+---------------------+---------------------+---------------+---------+----------------------+----------------------+--------------------+
| chandrika | customer_occupation | FUNCTION | root@localhost | 2021-03-24 15:51:18 | 2021-03-24 15:51:18 | DEFINER       |         | cp850                | cp850_general_ci     | latin1_swedish_ci  |
+-----------+--------------------+----------+----------------+---------------------+---------------------+---------------+---------+----------------------+----------------------+--------------------+
1 row in set (0.08 sec)

mysql>
```

To call the function:

```
mysql> SELECT name, age, customer_occupation(age)
    -> from customer_details order by age;
+---------+------+--------------------------+
| name    | age  | customer_occupation(age) |
+---------+------+--------------------------+
| raju    |   25 | Actor                    |
| prakash |   26 | Actor                    |
| Sudha   |   28 | Actor                    |
| Rama    |   30 | Engineer                 |
| peter   |   32 | Engineer                 |
| radhika |   45 | Scientist                |
+---------+------+--------------------------+
6 rows in set (0.02 sec)

mysql>
```

3. Write a PL/SQL program to demonstrate the creation and usage of cursor.

```
mysql> create table curser1(id integer, name varchar(25), class varchar(25));
Query OK, 0 rows affected (0.04 sec)

mysql> insert into curser1 values(101,'srishti','MCA');
Query OK, 1 row affected (0.03 sec)

mysql> insert into curser1 values(102,'Pragya','MBA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(103,'Archana','MCA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(104,'Namrata','MBA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(105,'Kiran','MCA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(106,'Anirudh','MBA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(107,'Sanvi','MCA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(108,'Vinay','MBA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(109,'Tarun','MCA');
Query OK, 1 row affected (0.00 sec)

mysql> insert into curser1 values(110,'Pooja','MBA');
Query OK, 1 row affected (0.00 sec)

mysql> select * from curser1;
+------+---------+-------+
| id   | name    | class |
+------+---------+-------+
|  101 | srishti | MCA   |
|  102 | Pragya  | MBA   |
|  103 | Archana | MCA   |
|  104 | Namrata | MBA   |
|  105 | Kiran   | MCA   |
|  106 | Anirudh | MBA   |
|  107 | Sanvi   | MCA   |
|  108 | Vinay   | MBA   |
|  109 | Tarun   | MCA   |
|  110 | Pooja   | MBA   |
+------+---------+-------+
10 rows in set (0.00 sec)
```

Now, Create the cursor:

```
mysql> delimiter //
mysql> CREATE PROCEDURE PRO1 (INOUT name_list Varchar(4000))
    -> BEGIN
    ->  DECLARE is_done INTEGER DEFAULT 0;
    ->  DECLARE s_name VARCHAR(100) DEFAULT "";
    ->  DECLARE stud_cursor CURSOR FOR
    ->          SELECT name FROM curser1;
    ->  DECLARE CONTINUE HANDLER FOR NOT FOUND SET is_done = 1;
    ->  OPEN stud_cursor;
    ->  get_list: LOOP
    -> FETCH stud_cursor INTO s_name;
    ->  IF is_done=1 THEN
    -> LEAVE get_list;
    -> END IF;
    -> SET name_list=CONCAT(s_name,";",name_list);
    -> END LOOP get_list;
    -> CLOSE stud_cursor;
    -> END
    -> //
Query OK, 0 rows affected (0.04 sec)

mysql> DELIMITER ;
mysql>
```

Now, call the cursor:

```
mysql> SET @name_list="";
Query OK, 0 rows affected (0.02 sec)

mysql> CALL PRO1(@name_list);
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT @name_list;
+-------------------------------------------------------------------+
| @name_list                                                        |
+-------------------------------------------------------------------+
| Pooja;Tarun;Vinay;Sanvi;Anirudh;Kiran;Namrata;Archana;Pragya;srishti; |
+-------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

4. Write a PL/SQL program to illustrate how trigger can be used to set values if inserted values are invalid.

Create table and insert values:

```
mysql> CREATE TABLE emp(
    ->     name varchar(45) NOT NULL,
    ->     occupation varchar(35) NOT NULL,
    ->     working_date date,
    ->     working_hours varchar(10)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO emp VALUES
    -> ('Robin', 'Scientist', '2020-10-04', 12),
    -> ('Warner', 'Engineer', '2020-10-04', 10),
    -> ('Peter', 'Actor', '2020-10-04', 13),
    -> ('Marco', 'Doctor', '2020-10-04', 14),
    -> ('Brayden', 'Teacher', '2020-10-04', 12),
    -> ('Antonio', 'Business', '2020-10-04', 11);
Query OK, 6 rows affected (0.02 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from emp;
+---------+------------+--------------+---------------+
| name    | occupation | working_date | working_hours |
+---------+------------+--------------+---------------+
| Robin   | Scientist  | 2020-10-04   | 12            |
| Warner  | Engineer   | 2020-10-04   | 10            |
| Peter   | Actor      | 2020-10-04   | 13            |
| Marco   | Doctor     | 2020-10-04   | 14            |
| Brayden | Teacher    | 2020-10-04   | 12            |
| Antonio | Business   | 2020-10-04   | 11            |
+---------+------------+--------------+---------------+
6 rows in set (0.00 sec)

mysql>
```

Create Trigger:

```
mysql> delimiter //
mysql>  Create Trigger before_insert_empworkinghours
    -> BEFORE INSERT ON emp FOR EACH ROW
    -> BEGIN
    -> IF NEW.working_hours < 0 THEN SET NEW.working_hours = 0;
    -> END IF;
    -> END
    -> //
Query OK, 0 rows affected (0.04 sec)
```

Now, check the output by inserting invalid values:

```
mysql> delimiter ;
mysql> INSERT INTO emp VALUES ('Markus','Farmer','2020-10-08',14);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO emp VALUES ('Alex','Actor','2020-10-012',-13);
Query OK, 1 row affected (0.03 sec)

mysql> select * from emp;
+---------+------------+--------------+---------------+
| name    | occupation | working_date | working_hours |
+---------+------------+--------------+---------------+
| Robin   | Scientist  | 2020-10-04   | 12            |
| Warner  | Engineer   | 2020-10-04   | 10            |
| Peter   | Actor      | 2020-10-04   | 13            |
| Marco   | Doctor     | 2020-10-04   | 14            |
| Brayden | Teacher    | 2020-10-04   | 12            |
| Antonio | Business   | 2020-10-04   | 11            |
| Markus  | Farmer     | 2020-10-08   | 14            |
| Alex    | Actor      | 2020-10-12   | 0             |
+---------+------------+--------------+---------------+
8 rows in set (0.00 sec)

mysql>
```

5. Write a PL/SQL program to illustrate exception handling in stored procedures.

First, create a new table named ***SupplierProducts*** for the demonstration:

```
mysql> CREATE TABLE SupplierProducts (
    ->     supplierId INT,
    ->     productId INT,
    ->     PRIMARY KEY (supplierId , productId)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql>
```

The table ***SupplierProducts*** stores the relationships between the table suppliers and products. Each supplier may provide many products and each product can be provided by many suppliers. For the sake of simplicity, we don't create Products and Suppliers tables, as well as the foreign keys in the ***SupplierProducts*** table.

Second, create a stored procedure that inserts product id and supplier id into the SupplierProducts table:

```
mysql> delimiter //
mysql> CREATE PROCEDURE InsertSupplierProduct(
    ->     IN inSupplierId INT,
    ->     IN inProductId INT
    -> )
    -> BEGIN
    ->     -- exit if the duplicate key occurs
    ->     DECLARE EXIT HANDLER FOR 1062
    ->     BEGIN
    ->         SELECT CONCAT('Duplicate key (',inSupplierId,',',inProductId,') occurred') AS message;
    ->     END;
    ->
    ->     -- insert a new row into the SupplierProducts
    ->     INSERT INTO SupplierProducts(supplierId,productId)
    ->     VALUES(inSupplierId,inProductId);
    ->
    ->     -- return the products supplied by the supplier id
    ->     SELECT COUNT(*)
    ->     FROM SupplierProducts
    ->     WHERE supplierId = inSupplierId;
    ->
    -> END
    -> //
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Now, we will start inserting the values:

```
mysql> delimiter ;
mysql> select * from supplierproducts;
+------------+-----------+
| supplierId | productId |
+------------+-----------+
|          1 |         1 |
|          1 |         2 |
|          1 |         3 |
+------------+-----------+
3 rows in set (0.00 sec)

mysql> CALL InsertSupplierProduct(1,1);
+-----------------------------+
| message                     |
+-----------------------------+
| Duplicate key (1,1) occurred |
+-----------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> CALL InsertSupplierProduct(1,2);
+-----------------------------+
| message                     |
+-----------------------------+
| Duplicate key (1,2) occurred |
+-----------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> CALL InsertSupplierProduct(1,3);
+-----------------------------+
| message                     |
+-----------------------------+
| Duplicate key (1,3) occurred |
+-----------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> CALL InsertSupplierProduct(1,3);
+-----------------------------+
| message                     |
+-----------------------------+
| Duplicate key (1,3) occurred |
+-----------------------------+
1 row in set (0.00 sec)
```

Output: the contents of the supplierproducts relation:

```
mysql> select * from supplierproducts;
+------------+-----------+
| supplierId | productId |
+------------+-----------+
|          1 |         1 |
|          1 |         2 |
|          1 |         3 |
+------------+-----------+
3 rows in set (0.00 sec)

mysql>
```