# VISVESVARAYA TECHNOLOGICAL UNIVERSITY
# BELGAUM

# DBMS LABORATORY WITH MINI PROJECT
# (Subject Code: 18CSL58)

## STUDENT MANUAL

### V-SEMESTER (ISE)

**Mrs Shilpa Shetty A**

Assistant Professor
Department of Information Science & Engineering

## A J INSTITUTE OF ENGINEERING & TECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
(A unit of Laxmi Memorial Education Trust. (R))
NH - 66, Kottara Chowki, Kodical Cross, Mangaluru- 575 006

## COURSE OBJECTIVE

**This course will enable students to:**

- Design and implement various algorithms in JAVA
- Employ various design strategies for problem solving.
- Measure and compare the performance of different algorithms.

# COURSE DESCRIPTION

Design, develop, and implement the specified algorithms for the following problems using Java language under LINUX /Windows environment. NetBeans /Eclipse IDE tool can be used for development and demonstration.

# COURSE OUTCOMES

**After studying this course, students will be able to:**

- Design algorithms using appropriate design techniques (brute-force, greedy, dynamic programming, etc.)
- Implement a variety of algorithms such assorting, graph related, combinatorial, etc., in a high level language.
- Analyze and compare the performance of algorithms using language features.
- Apply and implement learned algorithm design techniques and data structures to solve real world problems.

# CONTENTS

**DBMS LABORATORY WITH MINI PROJECT**
**[As per Choice Based Credit System (CBCS) scheme]**
**(Effective from the academic year 2018 -2019)**
**SEMESTER – V**

| | | | |
|---|---|---|---|
| Subject Code | 18CSL58 | IA Marks | 40 |
| Number of Lecture Hours/Week | 0:2:2 | Exam Marks | 60 |
| Total Number of Lecture Hours | 36 | Exam Hours | 03 |

**CREDITS – 02**

**Course objectives:** This course will enable students to

- Foundation knowledge in database concepts, technology and practice to groom students into well-informed database application developers.
- Strong practice in SQL programming through a variety of database problems.
- Develop database applications using front-end tools and back-end DBMS.

**Description (If any):**

**PART-A: SQL Programming (Max. Exam Mks. 50)**

- 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: Mini Project (Max. Exam Mks. 30)**

- Use Java, C#, PHP, Python, or any other similar front-end tool. All applications must be demonstrated on desktop/laptop as a stand-alone or web based application (Mobile apps on Android/IOS are not permitted.)

## PART A: SQL PROGRAMMING

**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, Programme_id, No of_Copies)
BOOK_LENDING(Book_id, Programme _id, Card_No, Date_Out, Due_Date) LIBRARY_PROGRAMME(Programme_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 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.

| | |
|---|---|
| **2** | Consider the following schema for Order Database:<br>SALESMAN(Salesman_id, Name, City, Commission)<br>CUSTOMER(Customer_id, Cust_Name, City, Grade, Salesman_id)<br>ORDERS(Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id)<br>Write SQL queries to<br>1. Count the customers with grades above Bangalore's average.<br>2. Find the name and numbers of all salesman who had more than one customer.<br>3. List all the salesman and indicate those who have and don't have customers in their cities (Use UNION operation.)<br>4. Create a view that finds the salesman who has the customer with the highest order of a day.<br>5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted. |
| **3** | Consider the schema for Movie Database: ACTOR(Act_id, Act_Name, Act_Gender) DIRECTOR(Dir_id, Dir_Name, Dir_Phone)<br>MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)<br>MOVIE_CAST(Act_id, Mov_id, Role)<br>RATING(Mov_id, Rev_Stars) Write SQL queries to<br>1. List the titles of all movies directed by 'Hitchcock'.<br>2. Find the movie names where one or more actors acted in two or more movies.<br>3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).<br>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.<br>5. Update rating of all movies directed by 'Steven Spielberg' to5. |
| **4** | Consider the schema for College Database: STUDENT(USN, SName, Address, Phone, Gender) SEMSEC(SSID, Sem, Sec)<br>CLASS(USN, SSID)<br>SUBJECT(Subcode, Title, Sem, Credits)<br>IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, Final IA)<br>Write SQL queries to<br>1. List all the student details studying in fourth semester 'C'section.<br>2. Compute the total number of male and female students in each semester and in each section.<br>3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.<br>4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.<br>5. Categorize students based on the following criterion:<br>If FinalIA = 17 to 20 then CAT ='Outstanding'<br>If FinalIA = 12 to 16 then CAT = 'Average' If FinalIA< 12 then CAT = 'Weak'<br>Give these details only for 8th semester A, B, and C section students. |

| | |
|---|---|
| **5** | Consider the schema for Company Database:<br>EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)<br>DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)<br>DLOCATION(DNo,DLoc)<br>PROJECT(PNo, PName, PLocation, DNo) WORKS_ON(SSN, PNo, Hours)<br>Write SQL queries to<br>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.<br>2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.<br>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<br>4. Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator).<br>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. |

**PART B: MINI PROJECT**

- For any problem selected, write the ER Diagram, apply ER-mapping rules, normalize the relations, and follow the application development process.
- Make sure that the application should have five or more tables, at least one trigger and one stored procedure, using suitable front end tool.
- Indicative areas include; health care, education, industry, transport, supply chain, etc.

**Course outcomes:** The students should be able to:

- Create, Update and query on the database.
- Demonstrate the working of different concepts of DBMS
- Implement, analyze and evaluate the project developed for an application.

**Conduction of Practical Examination:**

- Experiment distribution o For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
  - o For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only. • Marks Distribution *(Courseed to change in accoradance with university regulations)*

k) For laboratories having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks

l) For laboratories having PART A and PART B

1. i. Part A – Procedure + Execution + Viva = 6 + 28 + 6 = 40 Marks
2. ii. Part B – Procedure + Execution + Viva = 9 + 42 + 9 = 60 Marks

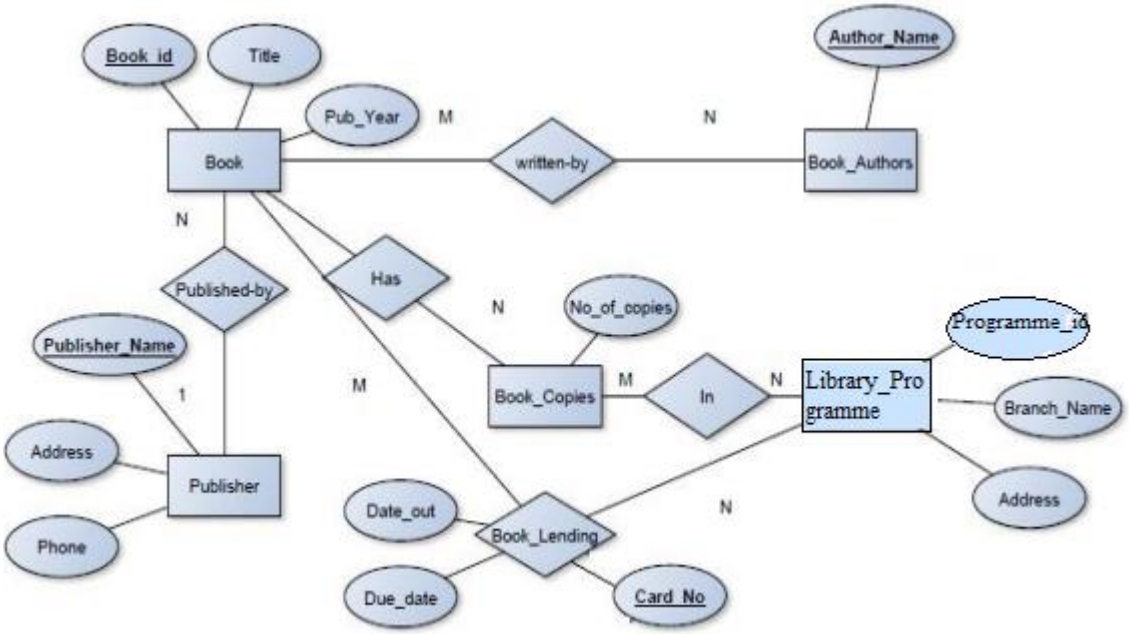
**1. Consider the following schema for a *Library Database*:**

**BOOK (*Book_id*, *Title*, *Publisher_Name*, *Pub_Year*)**
**BOOK_AUTHORS (Book_id, Author_*Name*)**
**PUBLISHER (*Name*, *Address*, *Phone*)**
**BOOK_COPIES (*Book_id*, *Programme_id*, *No-of_Copies*)**
**BOOK_LENDING (*Book_id*, *Programme_id*, *Card_No*, *Date_Out*, *Due_Date*) LIBRARY_PROGRAMME (*Programme_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 2017to 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.**

***Entity-Relationship Diagram:***



***Schema Diagram:***

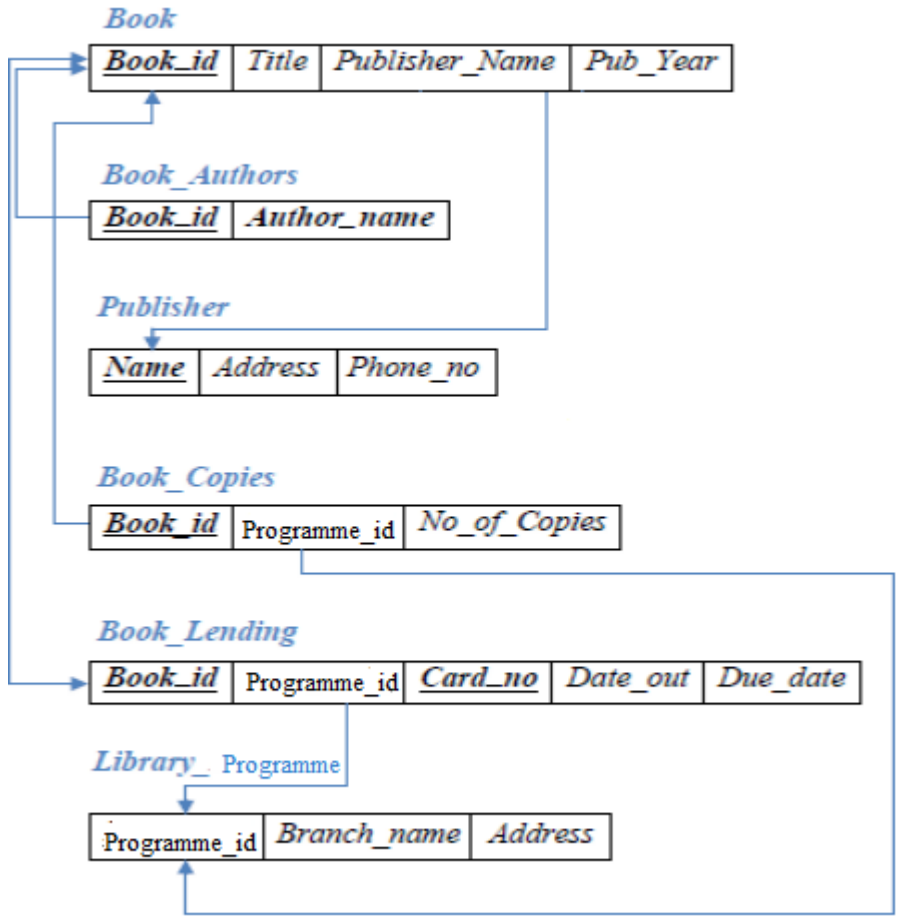



***Table Creation:***

```
create table PUBLISHER
(
Name varchar(15),
Address varchar(15),
Phone varchar(10),
Primary key (Name)
);

create table BOOK
(
Book_id varchar(4),
Title varchar(10),
Publisher_name varchar(10),
Pub_year int,
primary key (Book_id),
Foreign key(Publisher_name) references PUBLISHER(Name) on delete cascade);
```

```
create table BOOK_AUTHORS
(
Book_id varchar(4),
Author_name varchar(10),
primary key (Book_id),
Foreign key(Book_id) references BOOK(Book_id) on delete cascade
);


create table
LIBRARY_PROGRAMME
 (
Programme_id varchar(4),
Branch_name varchar(15),
Address varchar(15),
primary
key(Programme_id)
);


create table BOOK_COPIES
(
Book_id varchar(4),
Programme_id
varchar(4),
No_of_copies int,
primary key (Book_id, Programme_id),
Foreign key(Book_id) references BOOK(Book_id) on delete cascade,
Foreign key(Programme_id) references LIBRARY_PROGRAMME(Programme_id) on delete
cascade
);


create table BOOK_LENDING
(
Book_id varchar(4),
Programme_id
varchar(4),
    Card_no int,
Date_out date,
Due_date date,
primary key (Book_id, Programme_id, Card_no),
Foreign key(Book_id) references BOOK(Book_id) on delete cascade,
      Foreign key(Programme_id) references LIBRARY_PROGRAMME(Programme_id)
on delete cascade
);
```

***Table Descriptions:***

```
SQL> DESC PUBLISHER;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 NAME                                      NOT NULL VARCHAR2(15)
 ADDRESS                                            VARCHAR2(15)
 PHONE                                              VARCHAR2(10)

SQL> DESC BOOK;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 BOOK_ID                                   NOT NULL VARCHAR2(4)
 TITLE                                              VARCHAR2(10)
 PUBLISHER_NAME                                     VARCHAR2(10)
 PUB_YEAR                                           NUMBER(38)

SQL> DESC BOOK_AUTHORS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 BOOK_ID                                   NOT NULL VARCHAR2(4)
 AUTHOR_NAME                                        VARCHAR2(10)

SQL> DESC LIBRARY_PROGRAMME
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 PROGRAMME_ID                              NOT NULL VARCHAR2(4)
 BRANCH_NAME                                        VARCHAR2(15)
 ADDRESS                                            VARCHAR2(15)

SQL> DESC BOOK_COPIES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 BOOK_ID                                   NOT NULL VARCHAR2(4)
 PROGRAMME_ID                              NOT NULL VARCHAR2(4)
 NO_OF_COPIES                                       NUMBER(38)

SQL> DESC BOOK_LENDING;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 BOOK_ID                                   NOT NULL VARCHAR2(4)
 PROGRAMME_ID                              NOT NULL VARCHAR2(4)
 CARD_NO                                   NOT NULL NUMBER(38)
 DATE_OUT                                           DATE
 DUE_DATE                                           DATE
```

***Insertion of Values to Tables:***

insert into PUBLISHER values (’TMH’, ’Mangalore’,’9876543897’);
insert into PUBLISHER values (’Prism’, ’chennai’,’8756444324’);
insert into PUBLISHER values (’Himalaya’, ’Kolkata’,’9876556785’);
insert into PUBLISHER values (’Pearson’,’Delhi’,’9878987675’);
insert into PUBLISHER values (’Elsevier’,’Bangalore’,’7659876785’);

```
insert into BOOK values ('b1','DBMS', 'TMH',2015);
insert into BOOK values ('b2','DMS', 'Prism',2016);
insert into BOOK values ('b3','CN', 'Himalaya',2015);
insert into BOOK values ('b4','AI', 'Pearson',2013);
insert into BOOK values ('b5','OS', 'Elsevier',2017);

insert into BOOK_AUTHORS values ('b1','navathe');
insert into BOOK_AUTHORS values ('b2','dsc');
insert into BOOK_AUTHORS values ('b3','david');
insert into BOOK_AUTHORS values ('b4','stuart');
insert into BOOK_AUTHORS values ('b5','galvin');

insert into LIBRARY_PROGRAMME values ('bh1','Book Corner',
'Bangalore');
insert into LIBRARY_PROGRAMME values ('bh2','Book Point',
'Mangalore'); insert into LIBRARY_PROGROMME values ('bh3','Book
Cafe', 'Mumbai');

insert into BOOK_COPIES values ('b1','bh1',10);
insert into BOOK_COPIES values ('b1','bh2',15);
insert into BOOK_COPIES values ('b2','bh2',30);
insert into BOOK_COPIES values ('b2','bh3',28);
insert into BOOK_COPIES values ('b3','bh1',35);
insert into BOOK_COPIES values ('b3','bh2',22);
insert into BOOK_COPIES values ('b4','bh1',8);
insert into BOOK_COPIES values ('b5','bh3',17);
```

(**Note**: For **Mysql**use Date Format: **'YYYY-MM-DD'**. Eg: **'2017-01-02'**.)

```
insert into BOOK_LENDING values ('b1','bh1',1,'02-jan-17','10-jan-17');
insert into BOOK_LENDING values ('b2','bh2',2,'11-jan-17','11-mar-17');
insert into BOOK_LENDING values ('b3','bh1',1,'21-feb-17','21-apr-17');
insert into BOOK_LENDING values ('b5','bh3',1,'15-mar-17','15-jul-17');
insert into BOOK_LENDING values ('b3','bh2',4,'12-apr-17','12-may-17');
insert into BOOK_LENDING values ('b4','bh1',1,'21-feb-17','21-apr-17');
```

```
SQL> SELECT * FROM PUBLISHER;

NAME            ADDRESS         PHONE
--------------- --------------- ----------
TMH             Mangalore       9876543897
Prism           chennai         8756444324
Himalaya        Kolkata         9876556785
Pearson         Delhi           9878987675
Elsevier        Bangalore       7659876785
```

```
SQL> SELECT * FROM BOOK;

BOOK TITLE      PUBLISHER_   PUB_YEAR
---- ---------- ---------- ----------
b1   DBMS       TMH              2015
b2   DMS        Prism            2016
b3   CN         Himalaya         2015
b4   AI         Pearson          2013
b5   OS         Elsevier         2017
```

```
SQL> SELECT * FROM BOOK_AUTHORS;

BOOK AUTHOR_NAM
---- ----------
b1   navathe
b2   dsc
b3   david
b4   stuart
b5   galvin
```

```
SQL> SELECT * FROM LIBRARY_ PROGRAMME;

PROGRAMME_ID BRANCH_NAME     ADDRESS
---- ---------------- ---------------
bh1          Book Corner     Bangalore
bh2          Book Point      Mangalore
bh3          Book Cafe       Mumbai
```

```
SQL> SELECT * FROM BOOK_COPIES;

BOOK_ID  PROGRAMME_ID  NO_OF_COPIES
-----    ----          ------------
b1       bh1                   10
b1       bh2                   15
b2       bh2                   30
b2       bh3                   28
b3       bh1                   35
b3       bh2                   22
b4       bh1                    8
b5       bh3                   17
```

```
SQL> SELECT * FROM BOOK_LENDING;

BOOK_ID   PROGRAMME_ID    CARD_NO DATE_OUT  DUE_DATE
----      ----          --------- --------- ---------
b1        bh1                 1 02-JAN-17 10-JAN-17
b2        bh2                 2 11-JAN-17 11-MAR-17
b3        bh1                 1 21-FEB-17 21-APR-17
b5        bh3                 1 15-MAR-17 15-JUL-17
b3        bh2                 4 12-APR-17 12-MAY-17
b4        bh1                 1 21-FEB-17 21-APR-17
```

***Queries:***

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

```
Select b.book_id, b.title, b.publisher_name, a.author_name, c.no_of_copies,
lb.Programme_id from BOOK b, BOOK_AUTHORS a, BOOK_COPIES c,
LIBRARY_PROGRAMME lb
where a.book_id=b.book_id
        and b.book_id=c.book_id
        and c.Programme_id=lb.Programme_id;
```

```
SQL> select     b.book_id, b.title, b.publisher_name, a.author_name,
  2  c.no_of_copies, lb.branch_id
  3  from BOOK b, BOOK_AUTHORS a, BOOK_COPIES c,
  4  LIBRARY_PROGRAMME
  5  where a.book_id=b.book_id
  6             and b.book_id=c.book_id
  7             and c.Programme_id=lb. Programme_id;

BOOK TITLE      PUBLISHER_ AUTHOR_NAM NO_OF_COPIES  PROGRAMME_ID
---- ---------- ---------- ---------- ------------ ----
b1   DBMS       TMH        navathe              10 bh1
b1   DBMS       TMH        navathe              15 bh2
b2   DMS        Prism      dsc                  30 bh2
b2   DMS        Prism      dsc                  28 bh3
b3   CN         Himalaya   david                35 bh1
b3   CN         Himalaya   david                22 bh2
b4   AI         Pearson    stuart                8 bh1
b5   OS         Elsevier   galvin               17 bh3

8 rows selected.
```

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

   select Card_no
   from BOOK_LENDING
   where Date_out between '01-jan-2017' and '01-jul-2017'
   group by Card_no
   having count (*)>3;

```
SQL> select Card_no
  2  from BOOK_LENDING
  3  where Date_out between '01-jan-2017' and '01-jul-2017'
  4  group by Card_no
  5  having count (*)>3;

   CARD_NO
----------
         1
```

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

   delete from BOOK where Book_id='b4';

```
SQL> delete from BOOK where Book_id='b4';

1 row deleted.

SQL> SELECT * FROM BOOK;

BOOK TITLE      PUBLISHER_   PUB_YEAR
---- ---------- ---------- ----------
b1   DBMS       TMH              2015
b2   DMS        Prism            2016
b3   CN         Himalaya         2015
b5   OS         Elsevier         2017
```

   (**Note**: records corresponds to **Book_id ='b4'** is also deleted from BOOK_AUTHORS, LIBRARY_PROGRAMME, BOOK_COPIES and BOOK_LENDING tables because of the use of '***on delete CASCADE***' constraint on foreign keys).

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

```
create view V_PUBLICATION as
select Pub_year
from BOOK;

select * from V_PUBLICATION;
```

```
SQL> create view V_PUBLICATION as
  2  select Pub_year
  3  from BOOK;

View created.

SQL>
SQL> select * from V_PUBLICATION;

  PUB_YEAR
----------
      2015
      2016
      2015
      2017
```

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

```
create view V_BOOKS as
select b.Book_id, b.Title, c.Programme_id, c.No_of_copies
from BOOK b, BOOK_COPIES c, LIBRARY_PROGRAMME lb
where b.Book_id=c.Book_id
and
c.Programme_id=lb.Programm
e_id;

select * from V_BOOKS;
```

```
SQL> create view V_BOOKS as
  2  select b.Book_id, b.Title, c.Programme_id, c.No_of_copies
  3  from BOOK b, BOOK_COPIES c, LIBRARY_PROGRAMME lb
  4  where b.Book_id=c.Book_id
  5  and c.Programme_id=lb.Programme_id;

View created.

SQL>
SQL> select * from V_BOOKS;

BOOK TITLE      PROGRAMME_ID NO_OF_COPIES
---- ---------- ------------ ------------
b1   DBMS       bh1                  10
b1   DBMS       bh2                  15
b2   DMS        bh2                  30
b2   DMS        bh3                  28
b3   CN         bh1                  35
b3   CN         bh2                  22
b5   OS         bh3                  17
```

## Viva Questions

1. **What is data?**
2. **What is database?**
3. **What is DBMS?**
4. **What is a Database system?**
5. **What are the advantages of DBMS?**
6. **What is relational database?**
7. **What is Table?**
8. **What is a Tuple?**
9. **What is Columns?**
10. **What is a query?**

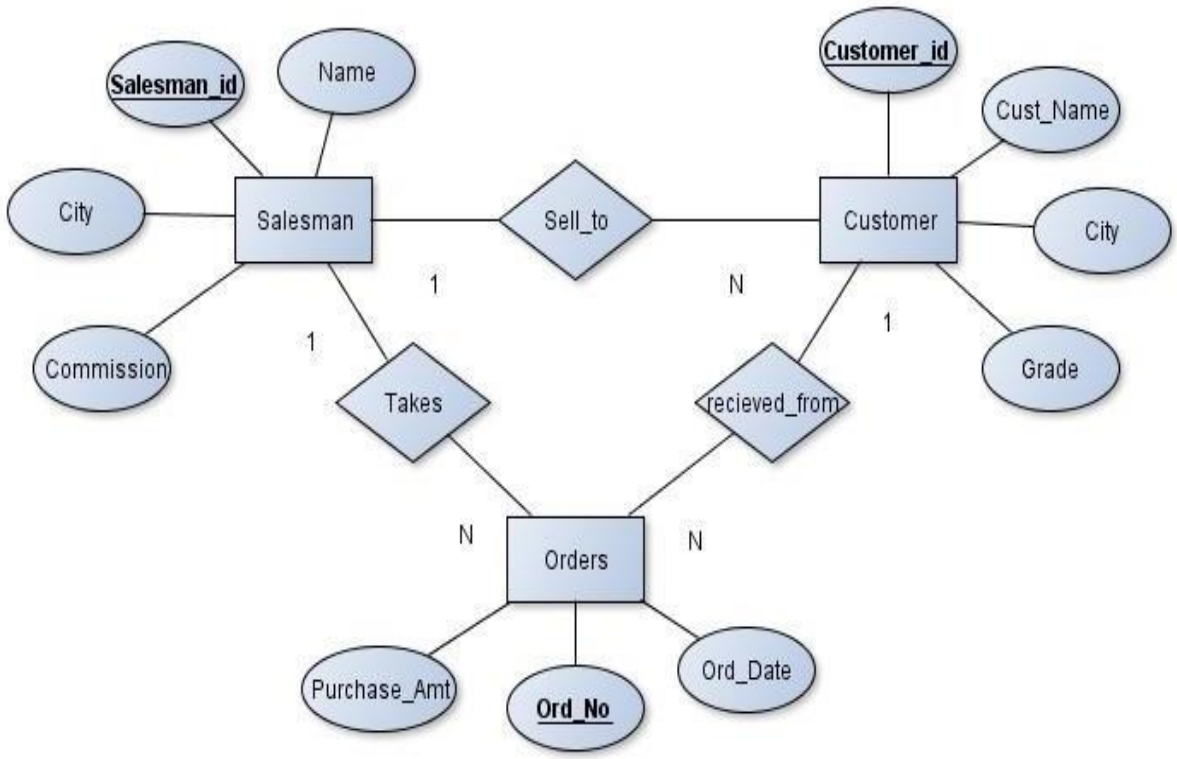
**2. Consider the following schema for Order Database:**

**SALESMAN (*Salesman_id, Name, City, Commission*)**
**CUSTOMER (*Customer_id, Cust_Name, City, Grade,Salesman_id*)**
**ORDERS (*Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id*)**

**Write SQL queries to**

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

***Entity-Relationship Diagram***



***Schema Diagram***

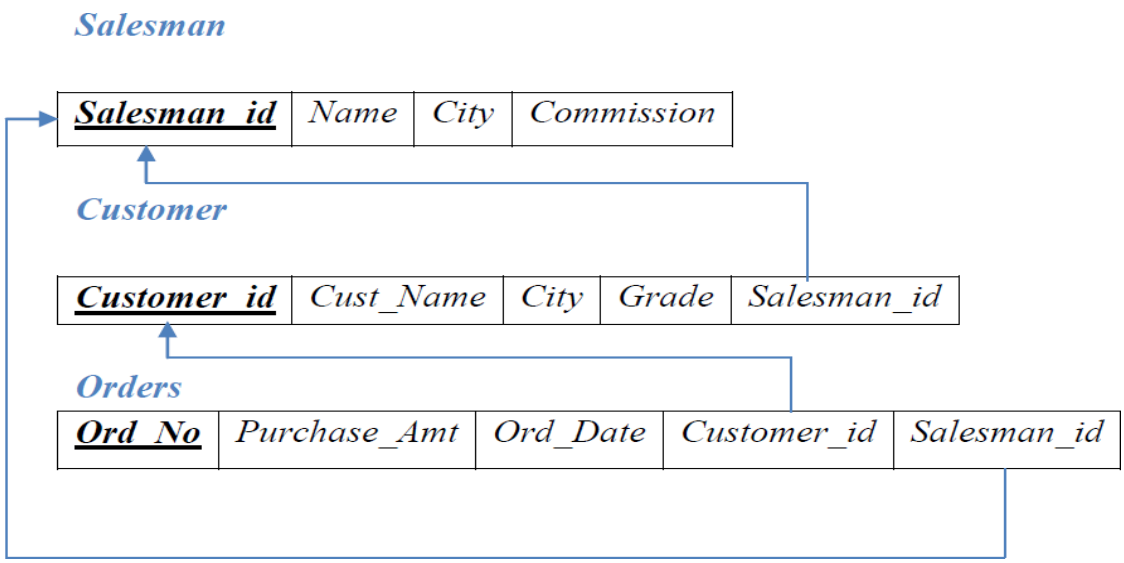



***Table Creation***

```
create table SALESMAN
(
Salesman_id int,
Name varchar(10),
City varchar(10),
Commission int,
primary key(Salesman_id)
);

create table CUSTOMER
(
Customer_id int,
Cust_name varchar(10),
City varchar(10),
Grade int,
Salesman_id int,
primary key (Customer_id),
foreign key (Salesman_id) references SALESMAN (Salesman_id) on delete set NULL
);
```

```
create table ORDERS
(
Ord_no int,
Purchase_amt int,
Ord_date date,
Customer_id int,
Salesman_id int,
primary key (Ord_no),
foreign key (Customer_id) references CUSTOMER (customer_id) on delete cascade,
foreign key (Salesman_id) references SALESMAN (salesman_id) on delete cascade
);
```

***Table Descriptions:***

```
SQL> desc SALESMAN;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 SALESMAN_ID                               NOT NULL NUMBER(38)
 NAME                                               VARCHAR2(10)
 CITY                                               VARCHAR2(10)
 COMMISSION                                         NUMBER(38)

SQL> desc CUSTOMER;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 CUSTOMER_ID                               NOT NULL NUMBER(38)
 CUST_NAME                                          VARCHAR2(10)
 CITY                                               VARCHAR2(10)
 GRADE                                              NUMBER(38)
 SALESMAN_ID                                        NUMBER(38)

SQL> desc ORDERS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 ORD_NO                                    NOT NULL NUMBER(38)
 PURCHASE_AMT                                       NUMBER(38)
 ORD_DATE                                           DATE
 CUSTOMER_ID                                        NUMBER(38)
 SALESMAN_ID                                        NUMBER(38)
```

***Insertion Of Values To Tables:***

***Salesman Details:***

insert into SALESMAN values (1000, 'joseph','mysore','13');
insert into SALESMAN values (1001, 'girish','bangalore','22');
insert into SALESMAN values (1002, 'mukund','mumbai','16');
insert into SALESMAN values (1003, 'saurabh','delhi','19');
insert into SALESMAN values (1004, 'srinivas','hydrabad','23');
insert into SALESMAN values (1005, 'mohan','ranchi','23');

***Customer Details:***

insert into CUSTOMER values (1, 'sharal','hydrabad',40,1004);
insert into CUSTOMER values (2,'meenakshi','mangalore',40,1000);
insert into CUSTOMER values (3,'vikky','mumbai',35,1002);
insert into CUSTOMER values (4, 'john','mumbai',20,1002);
insert into CUSTOMER values (5, 'george','bangalore',10,1001);
insert into CUSTOMER values (6, 'hevin','bangalore',50,1001);
insert into CUSTOMER values (7, 'roshan','delhi',45,1003);
insert into CUSTOMER values (8, 'vimala','chennai',35,1001);
insert into CUSTOMER values (9, 'nakul','ayodhya',15,1005);

***Order Details:***
*(**NOTE**: Use **'YYYY-MM-DD'** date format for **MySQL**, Example:**'2017-01-04')*

insert into ORDERS values (111, 50000, '04-jan-17', 1, 1004);
insert into ORDERS values (222, 45000, '04-jan-17', 2, 1000);
insert into ORDERS values (333, 10000, '05-feb-17', 3, 1002);
insert into ORDERS values (444, 35000, '13-mar-17', 4, 1003);
insert into ORDERS values (555, 75000, '14-mar-17', 5, 1001);
insert into ORDERS values (666, 25000, '14-mar-17', 6, 1004);
insert into ORDERS values (777, 5000, '27-jun-17', 7, 1003);
insert into ORDERS values (888, 52000, '25-aug-17', 8, 1001);
insert into ORDERS values (991, 37000, '25-aug-17', 1, 1004);
insert into ORDERS values (992, 29000, '09-sep-17', 2, 1000);
insert into ORDERS values (993, 6000, '09-sep-17', 9, 1005);

```
SQL> SELECT * FROM SALESMAN;

SALESMAN_ID NAME       CITY       COMMISSION
----------- ---------- ---------- ----------
       1000 joseph     mysore             13
       1001 girish     bangalore          22
       1002 mukund     mumbai             16
       1003 saurabh    delhi              19
       1004 srinivas   hydrabad           23
       1005 mohan      ranchi             23

6 rows selected.

SQL> SELECT * FROM CUSTOMER;

CUSTOMER_ID CUST_NAME  CITY           GRADE SALESMAN_ID
----------- ---------- ---------- ---------- -----------
          1 sharal     hydrabad           40        1004
          2 meenakshi  mangalore          40        1000
          3 vikky      mumbai             35        1002
          4 john       mumbai             20        1002
          5 george     bangalore          10        1001
          6 hevin      bangalore          50        1001
          7 roshan     delhi              45        1003
          8 vimala     chennai            35        1001
          9 nakul      ayodhya            15        1005

9 rows selected.

SQL> SELECT * FROM ORDERS;

    ORD_NO PURCHASE_AMT ORD_DATE  CUSTOMER_ID SALESMAN_ID
---------- ------------ --------- ----------- -----------
       111        50000 04-JAN-17           1        1004
       222        45000 04-JAN-17           2        1000
       333        10000 05-FEB-17           3        1002
       444        35000 13-MAR-17           4        1003
       555        75000 14-MAR-17           5        1001
       666        25000 14-MAR-17           6        1004
       777         5000 27-JUN-17           7        1003
       888        52000 25-AUG-17           8        1001
       991        37000 25-AUG-17           1        1004
       992        29000 09-SEP-17           2        1000
       993         6000 09-SEP-17           9        1005
```

***Queries:***

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

```
select Grade, COUNT (distinct Customer_id) as Total_Customers
from CUSTOMER
group by Grade
having Grade > (select AVG(Grade) from CUSTOMER
                where City='bangalore');
```

```
SQL> select AVG(Grade) from CUSTOMER where City='bangalore';

AVG(GRADE)
----------
        30

SQL> select Grade, COUNT (distinct Customer_id) as Total_Customers
  2  from CUSTOMER
  3  group by Grade
  4  having Grade > (select AVG(Grade) from CUSTOMER
  5                               where City='bangalore');

     GRADE TOTAL_CUSTOMERS
---------- ---------------
        35               2
        40               2
        45               1
        50               1
```

**2. Find the name and numbers of all salesmen who had more than onecustomer.**

```
select s.Salesman_id, s.Name
from SALESMAN s
where (select COUNT (*) from CUSTOMER c
            where c.Salesman_id=s.Salesman_id) > 1;
```

```
SQL> select s.Salesman_id, s.Name
  2  from SALESMAN s
  3  where (select COUNT (*) from CUSTOMER c
  4                    where c.Salesman_id=s.Salesman_id) > 1;

SALESMAN_ID NAME
----------- ----------
       1001 girish
       1002 mukund
```

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

```
(select a.Salesman_id, a.Name, b.Cust_name, a.Commission, a.City
from SALESMAN a, CUSTOMER b
where a.City = b.City)
UNION
(select Salesman_id, Name, 'No Match', Commission, City
from SALESMAN
where NOT City = ANY (select City from CUSTOMER))
ORDER BY 2 DESC;
```

```
SQL> select a.Salesman_id, a.Name, b.Cust_name, a.Commission, a.City
  2  from SALESMAN a, CUSTOMER b
  3  where a.City = b.City;

SALESMAN_ID NAME       CUST_NAME  COMMISSION CITY
----------- ---------- ---------- ---------- ----------
       1004 srinivas   sharal             23 hydrabad
       1002 mukund     vikky              16 mumbai
       1002 mukund     john               16 mumbai
       1001 girish     george             22 bangalore
       1001 girish     hevin              22 bangalore
       1003 saurabh    roshan             19 delhi

6 rows selected.
```

```
SQL> select Salesman_id, Name, 'No Match', Commission, City
  2  from SALESMAN
  3  where NOT City = ANY (select City from CUSTOMER);

SALESMAN_ID NAME       'NOMATCH COMMISSION CITY
----------- ---------- -------- ---------- ----------
       1000 joseph     No Match         13 mysore
       1005 mohan      No Match         23 ranchi
```

**OUTPUT OF UNION:**

```
SQL> (select a.Salesman_id, a.Name, b.Cust_name, a.Commission, a.City
  2  from SALESMAN a, CUSTOMER b
  3  where a.City = b.City)
  4  UNION
  5  (select Salesman_id, Name, 'No Match', Commission, City
  6  from SALESMAN
  7  where NOT City = ANY (select City from CUSTOMER))
  8  ORDER BY 2 DESC;

SALESMAN_ID NAME       CUST_NAME  COMMISSION CITY
----------- ---------- ---------- ---------- ----------
       1004 srinivas   sharal              23 hydrabad
       1003 saurabh    roshan             19 delhi
       1002 mukund     john               16 mumbai
       1002 mukund     vikky              16 mumbai
       1005 mohan      No Match           23 ranchi
       1000 joseph     No Match           13 mysore
       1001 girish     george             22 bangalore
       1001 girish     hevin              22 bangalore

8 rows selected.
```

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

*(NOTE: Execute **Query-1** to create a view. Then execute **Query-2** to display that on SQL Command Line console.)*

***Query-1:***

```
create view TOPSALESMAN as
select b.Ord_date,b.Purchase_amt,a.Salesman_id, a.Name
from SALESMAN a, ORDERS b
where a.Salesman_id = b.Salesman_id
and b.Purchase_amt=(select MAX(c.Purchase_amt)
                    from ORDERS c where b.Ord_date = c.Ord_date) ;
```

```
SQL> create view TOPSALESMAN as
  2  select b.Ord_date,b.Purchase_amt,a.Salesman_id, a.Name
  3  from SALESMAN a, ORDERS b
  4  where a.Salesman_id = b.Salesman_id
  5  and b.Purchase_amt=(select MAX(c.Purchase_amt)
  6                      from ORDERS c where b.Ord_date = c.Ord_date) ;

View created.
```

***Query-2:***

select * from TOPSALESMAN;

```
SQL> select * from TOPSALESMAN;

ORD_DATE  PURCHASE_AMT SALESMAN_ID NAME
--------- ------------ ----------- ----------
04-JAN-17        50000        1004 srinivas
05-FEB-17        10000        1002 mukund
13-MAR-17        35000        1003 saurabh
14-MAR-17        75000        1001 girish
27-JUN-17         5000        1003 saurabh
25-AUG-17        52000        1001 girish
09-SEP-17        29000        1000 joseph

7 rows selected.
```

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

Delete from SALESMAN Where Salesman_Id=1000;

```
SQL> Delete from SALESMAN  Where Salesman_Id=1000;

1 row deleted.
```

```
SQL> select * from SALESMAN;

SALESMAN_ID NAME       CITY       COMMISSION
----------- ---------- ---------- ----------
       1001 girish     bangalore          22
       1002 mukund     mumbai             16
       1003 saurabh    delhi              19
       1004 srinivas   hydrabad           23
       1005 mohan      ranchi             23
```

We can verify from the above snapshot that, a Salesman named 'JOSEPH' with ID 1000 has been removed from the **SALESMAN** table.

```
SQL> select * from CUSTOMER;

CUSTOMER_ID CUST_NAME  CITY           GRADE SALESMAN_ID
----------- ---------- ---------- ---------- -----------
          1 sharal     hydrabad           40        1004
          2 meenakshi  mangalore          40
          3 vikky      mumbai             35        1002
          4 john       mumbai             20        1002
          5 george     bangalore          10        1001
          6 hevin      bangalore          50        1001
          7 roshan     delhi              45        1003
          8 vimala     chennai            35        1001
          9 nakul      ayodhya            15        1005

9 rows selected.
```

We can verify from the above snapshot that, as we had assigned ON DELETE SET NULL constraint on **Salesman_id**(which is a **foreign key**) in **CUSTOMER** table, only the deleted Salesman_id is replaced by **NULL**.

```
SQL> select * from ORDERS;

    ORD_NO PURCHASE_AMT ORD_DATE  CUSTOMER_ID SALESMAN_ID
---------- ------------ --------- ----------- -----------
       111        50000 04-JAN-17           1        1004
       333        10000 05-FEB-17           3        1002
       444        35000 13-MAR-17           4        1003
       555        75000 14-MAR-17           5        1001
       666        25000 14-MAR-17           6        1004
       777         5000 27-JUN-17           7        1003
       888        52000 25-AUG-17           8        1001
       991        37000 25-AUG-17           1        1004
       993         6000 09-SEP-17           9        1005

9 rows selected.
```

We can verify from the above snapshot that, as we had assigned ON DELETE CASCADE constraint on **Salesman_id**(which is a **foreign key**) in **ORDERS** table, the complete order details of **Ord_ID**222 and 992 are deleted from **ORDERS** table (which was related to Salesman ID 1000).

## Viva Questions

1. **What is an Attribute?**
2. **What is Single valued Attributes ?**
3. **What is Multi valued Attributes?**
4. **What is Compound /Composite Attribute?**
5. **What is Simple/Atomic Attributes?**
6. **What is Stored Attribute?**
7. **What is Derived Attribute ?**
8. **What is Complex Attributes?**
9. **What is Key Attribute ?**
10. **What is Non Key Attributes ?**

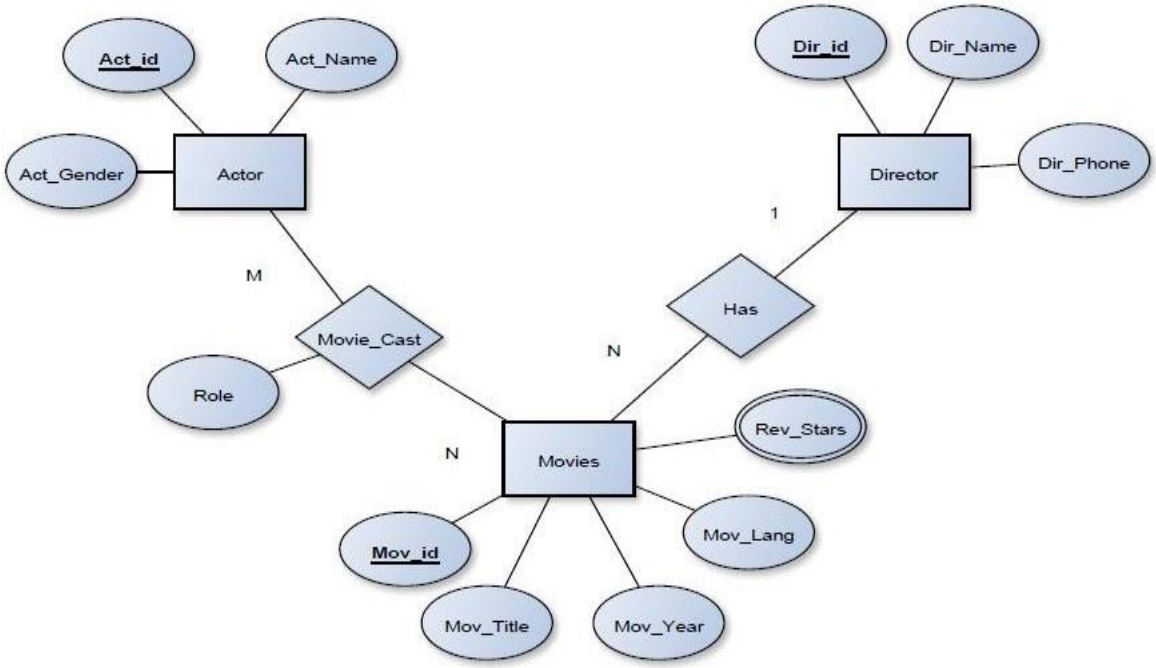
**3. Consider the schema for Movie Database:**

***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' to5.**

***Entity-Relationship Diagram:***



***Schema Diagram:***

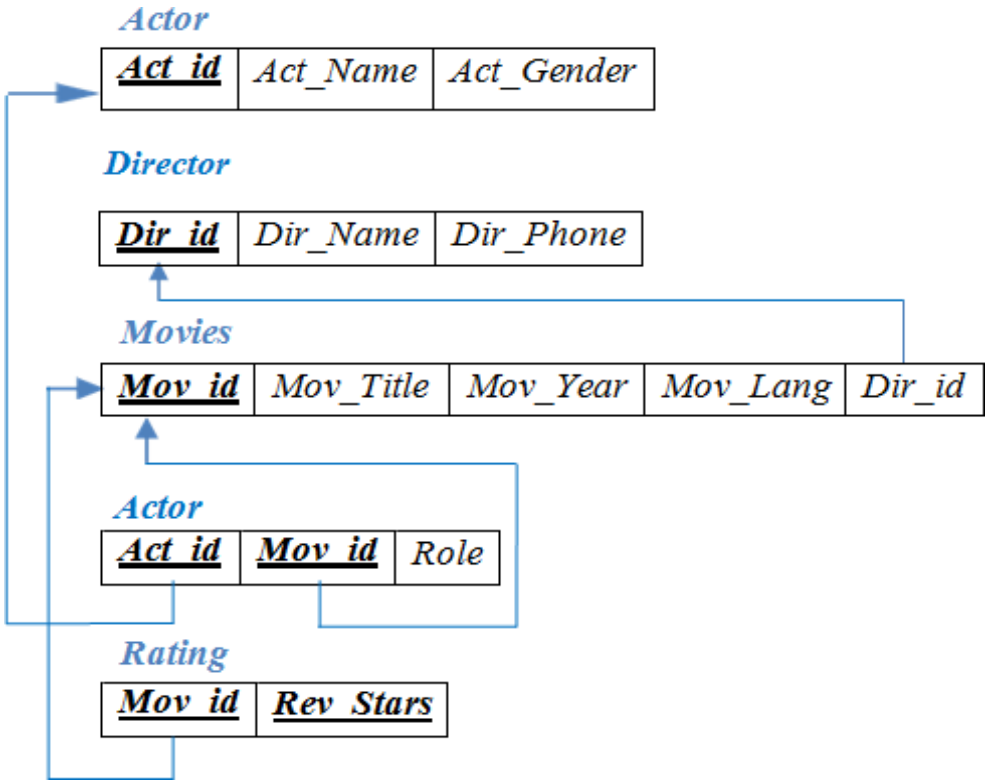



***Table Creation :***

```
create table ACTOR
(
Act_id varchar(3),
Act_name varchar(10),
Act_gender varchar(1),
primary key(Act_id)
);


create table DIRECTOR
(
Dir_id varchar(3),
Dir_name varchar(20),
Dir_phone varchar(10),
primary key(Dir_id)
);
```

```
create table MOVIES
(
Mov_id varchar(3),
Mov_title varchar(20),
Mov_year int,
Mov_lang varchar(10),
Dir_id varchar(3),
primary key(Mov_id),
foreign key(Dir_id) references DIRECTOR(Dir_id) on delete set NULL
);


create table MOVIE_CAST
(
Act_id varchar(3),
Mov_id varchar(3),
Role varchar(10),
primary key (Act_id, Mov_id),
foreign key (Act_id) references actor (Act_id) on delete set NULL,
foreign key (Mov_id) references movies (Mov_id) on delete set NULL
);

create table RATING
(
Mov_id varchar(3),
Rev_stars int,
primary key(Mov_id, Rev_stars),
foreign key(Mov_id) references MOVIES (Mov_id) on delete set NULL
);
```

**Note**: In RATINGS table **Mov_id&Rev_stars** is defined as **composite key** so that more than one rating can be assigned for a movie (Needed for query-4).

***Table Descriptions:***

```
SQL> DESC ACTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 ACT_ID                                    NOT NULL VARCHAR2(3)
 ACT_NAME                                           VARCHAR2(10)
 ACT_GENDER                                         VARCHAR2(1)

SQL> DESC DIRECTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 DIR_ID                                    NOT NULL VARCHAR2(3)
 DIR_NAME                                           VARCHAR2(20)
 DIR_PHONE                                          VARCHAR2(10)
```

```
SQL> DESC MOVIES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 MOV_ID                                    NOT NULL VARCHAR2(3)
 MOV_TITLE                                          VARCHAR2(20)
 MOV_YEAR                                           NUMBER(38)
 MOV_LANG                                           VARCHAR2(10)
 DIR_ID                                             VARCHAR2(3)

SQL> DESC MOVIE_CAST;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 ACT_ID                                    NOT NULL VARCHAR2(3)
 MOV_ID                                    NOT NULL VARCHAR2(3)
 ROLE                                               VARCHAR2(10)

SQL> DESC RATING;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 MOV_ID                                    NOT NULL VARCHAR2(3)
 REV_STARS                                 NOT NULL NUMBER(38)
```

***Insertion Of Values To the Tables:***

insert into ACTOR values (’a1’,’robert d’,’m’);
insert into ACTOR values (’a2’,’scarlett’,’f’);
insert into ACTOR values (’a3’,’puneeth’,’m’);
insert into ACTOR values (’a4’,’meera’,’f’);
insert into ACTOR values (’a5’,’prabhas’,’m’);
insert into ACTOR values (’a6’,’anushka’,’f’);

insert into DIRECTOR values (’d1’,’hitchcock’, ’7690870681’);
insert into DIRECTOR values (’d2’,’steven spielberg’, ’7986554437’);
insert into DIRECTOR values (’d3’,’mahesh babu’, ’8765675304’);
insert into DIRECTOR values (’d4’,’rajamouli’, ’9651232245’);

insert into MOVIES values (’m1’,’iron Man-1’, 1990, ’english’,’d1’);
insert into MOVIES values (’m2’,’munna’, 1998, ’telugu’,’d3’);
insert into MOVIES values (’m3’,’iron Man-2’, 2001, ’english’,’d2’);
insert into MOVIES values (’m4’,’arasu’, 2007, ’kannada’,’d3’);
insert into MOVIES values (’m5’,’iron Man-3’, 2016, ’english’,’d2’);
insert into MOVIES values (’m6’,’bahubali-2’, 2017, ’telugu’,’d4’);

```
insert into MOVIE_CAST values (’a1’, ’m1’,’hero’);
insert into MOVIE_CAST values (’a5’, ’m2’, ’hero’);
insert into MOVIE_CAST values (’a1’, ’m3’, ’hero’);
insert into MOVIE_CAST values (’a2’, ’m3’, ’heroine’);
insert into MOVIE_CAST values (’a2’, ’m5’, ’guest’);
insert into MOVIE_CAST values (’a3’, ’m4’, ’hero’);
insert into MOVIE_CAST values (’a4’, ’m4’, ’heroine’);
insert into MOVIE_CAST values (’a1’, ’m5’, ’hero’);
insert into MOVIE_CAST values (’a5’, ’m6’, ’hero’);
insert into MOVIE_CAST values (’a6’, ’m6’, ’heroine’);

insert into RATING values (’m1’,8);
insert into RATING values (’m2’,4);
insert into RATING values (’m3’,6);
insert into RATING values (’m4’,8);
insert into RATING values (’m5’,7);
insert into RATING values (’m6’,9);
insert into RATING values (’m2’,9);
insert into RATING values (’m1’,4);
```

***Initial Database With Valid Data:***

```
SQL> SELECT * FROM ACTOR;

ACT ACT_NAME   A
--- ---------- -
a1  robert d   m
a2  scarlett   f
a3  puneeth    m
a4  meera      f
a5  prabhas    m
a6  anushka    f

6 rows selected.

SQL> SELECT * FROM DIRECTOR;

DIR DIR_NAME            DIR_PHONE
--- ------------------- ----------
d1  hitchcock           7690870681
d2  steven spielberg    7986554437
d3  mahesh babu         8765675304
d4  rajamouli           9651232245
```

```
SQL> SELECT * FROM MOVIES;

MOV MOV_TITLE              MOV_YEAR MOV_LANG   DIR
--- -------------------- ---------- ---------- ---
m1  iron Man-1                 1990 english    d1
m2  munna                      1998 telugu     d3
m3  iron Man-2                 2001 english    d2
m4  arasu                      2007 kannada    d3
m5  iron Man-3                 2016 english    d2
m6  bahubali-2                 2017 telugu     d4

6 rows selected.
```

```
SQL> SELECT * FROM MOVIE_CAST;

ACT MOV ROLE
--- --- ----------
a1  m1  hero
a5  m2  hero
a1  m3  hero
a2  m3  heroine
a2  m5  guest
a3  m4  hero
a4  m4  heroine
a1  m5  hero
a5  m6  hero
a6  m6  heroine

10 rows selected.
```

```
SQL> SELECT * FROM RATING;

MOV  REV_STARS
--- ----------
m1           8
m2           4
m3           6
m4           8
m5           7
m6           9
m2           9
m1           4

8 rows selected.
```

***Queries:***

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

```
select Mov_title
from MOVIES
where Dir_id IN (select Dir_id from DIRECTOR
                 where Dir_name = 'hitchcock');
```

```
SQL> select Dir_id from DIRECTOR
  2  where Dir_name = 'hitchcock';

DIR
---
d1
```

```
SQL> select Mov_title
  2  from MOVIES
  3  where Dir_id IN (select Dir_id from DIRECTOR
  4                   where Dir_name = 'hitchcock');

MOV_TITLE
--------------------
iron Man-1
```

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

```
Select Mov_title
from MOVIES m, MOVIE_CAST mc
where m.Mov_id=mc.Mov_id
and mc.Act_id IN (select Act_id from MOVIE_CAST
group by Act_id having COUNT (Act_id)>1)
group by Mov_title;
```

```
SQL> select Act_id from MOVIE_CAST
  2  group by Act_id having COUNT (*)>1;

ACT
---
a1
a2
a5
```

```
SQL> select Mov_title
  2  from MOVIES m, MOVIE_CAST mc
  3  where m.Mov_id=mc.Mov_id
  4  and mc.Act_id IN (select Act_id from MOVIE_CAST
  5  group by Act_id having COUNT (Act_id)>1)
  6  group by Mov_title;

MOV_TITLE
--------------------
iron Man-1
iron Man-2
iron Man-3
munna
bahubali-2
```

***Alternate Query:****(with actor's name)*

```
select Mov_title, a.Act_name
from MOVIES m, MOVIE_CAST mc, ACTOR a
where m.Mov_id=mc.Mov_id
and mc.Act_id=a.Act_id
and mc.Act_id IN (select Act_id from MOVIE_CAST
group by Act_id having COUNT (*)>1);
```

```
SQL> select Mov_title, a.Act_name
  2  from MOVIES m, MOVIE_CAST mc, ACTOR a
  3  where m.Mov_id=mc.Mov_id
  4  and mc.Act_id=a.Act_id
  5  and mc.Act_id IN (select Act_id from MOVIE_CAST
  6  group by Act_id having COUNT (*)>1);

MOV_TITLE            ACT_NAME
-------------------- ----------
iron Man-1           robert d
iron Man-2           robert d
iron Man-3           robert d
iron Man-2           scarlett
iron Man-3           scarlett
munna                prabhas
bahubali-2           prabhas

7 rows selected.
```

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

```
select Act_name, Mov_title, Mov_year
from ACTOR a
JOIN MOVIE_CAST c
    ON a.Act_id=c.Act_id
JOIN MOVIESm
    ON c.Mov_id=m.Mov_id
where m.Mov_year NOT BETWEEN 2000 and 2015;
```

```
SQL> select Act_name, Mov_title, Mov_year
  2  from ACTOR a
  3  JOIN MOVIE_CAST c
  4  ON a.Act_id=c.Act_id
  5  JOIN MOVIES  m
  6  ON c.Mov_id=m.Mov_id
  7  where m.Mov_year NOT BETWEEN 2000 and 2015;

ACT_NAME   MOV_TITLE              MOV_YEAR
---------- -------------------- ----------
robert d   iron Man-1                 1990
prabhas    munna                      1998
robert d   iron Man-3                 2016
scarlett   iron Man-3                 2016
prabhas    bahubali-2                 2017
anushka    bahubali-2                 2017

6 rows selected.
```

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

```
select 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;
```

```
SQL> select Mov_title, MAX(Rev_stars)
  2  from MOVIES
  3  INNER JOIN RATING using(Mov_id)
  4  group by Mov_title
  5  having MAX(Rev_stars)>0
  6  order by Mov_title;

MOV_TITLE            MAX(REV_STARS)
-------------------- --------------
arasu                             8
bahubali-2                        9
iron Man-1                        8
iron Man-2                        6
iron Man-3                        7
munna                             9

6 rows selected.
```

***Alternate Query:***

select Mov_title, MAX(Rev_stars) as Best_Rating
from MOVIES m, RATING r
where m.Mov_id= r.Mov_id
and r.Rev_stars IS NOT NULL
group by Mov_title
order by Mov_title;

```
SQL> select Mov_title, MAX(Rev_stars) as Best_Rating
  2  from MOVIES m, RATING r
  3  where m.Mov_id= r.Mov_id
  4  and r.Rev_stars IS NOT NULL
  5  group by Mov_title
  6  order by Mov_title;

MOV_TITLE          BEST_RATING
------------------ -----------
arasu                        8
bahubali-2                   9
iron Man-1                   8
iron Man-2                   6
iron Man-3                   7
munna                        9

6 rows selected.
```

**5. Update rating of all movies directed by 'Steven Spielberg' to5.**

**NOTE**: If there are two ratings for a particular movie directed by 'Steven Spielberg' then query can't be executed, because we have defined a **composite key** (movie_id, Rev_stars)

```
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'));
```

```
SQL> update RATING
  2  set Rev_stars=5
  3  where Mov_id IN(select Mov_id from MOVIES
  4        where Dir_id IN(select Dir_id from DIRECTOR
  5              where Dir_name = 'steven spielberg'));

2 rows updated.
```

```
SQL> SELECT * FROM RATING;

MOV  REV_STARS
--- ----------
m1           8
m2           4
m3           5
m4           8
m5           5
m6           9
m2           9
m1           4

8 rows selected.
```

## Viva Questions

1. **What is a primary key?**
2. **What are the conditions for a field to be a primary key?**
3. **What is a Foreign Key ?**
4. **What is Super Key**?
5. **What is Candidate Key**
6. **What is a query?**
7. **Define SQL Insert Statement ?**
8. **Define SQL Update Statement ?**
9. **Define SQL Delete Statement ?**
10. **What is order by clause?**

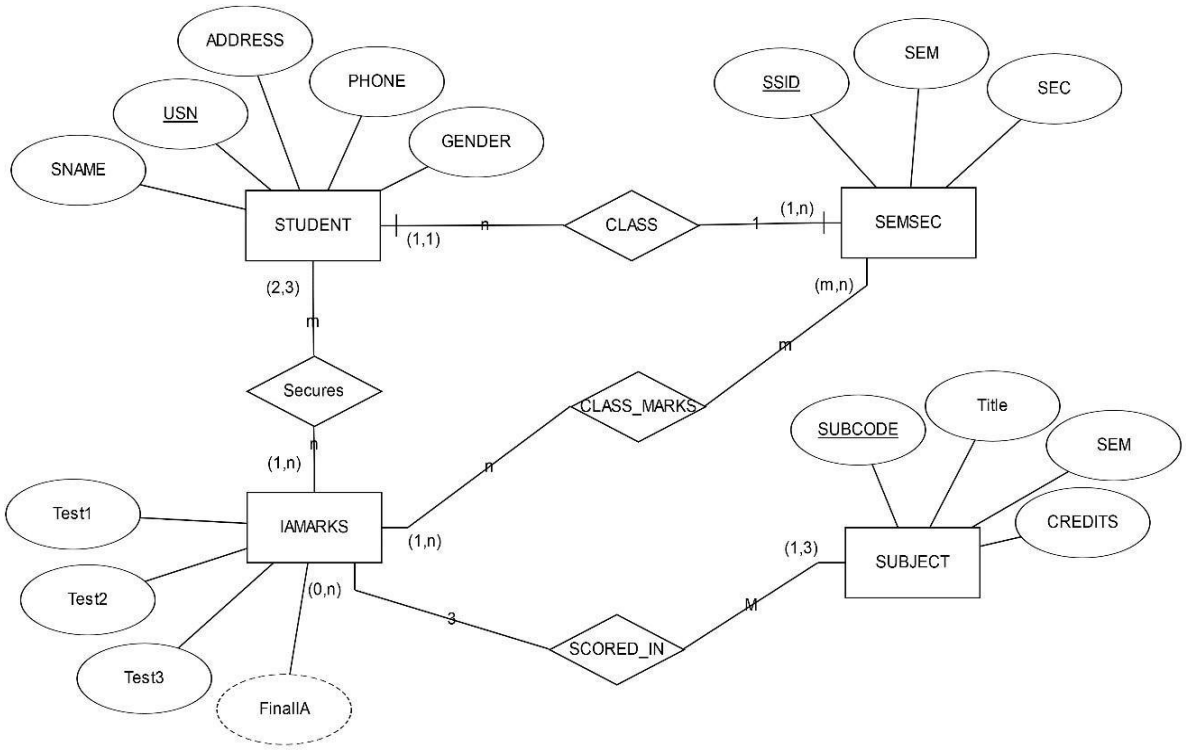
**4. Consider the schema for College Database:**

**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.**

***Entity - Relationship Diagram***



***Schema Diagram:***

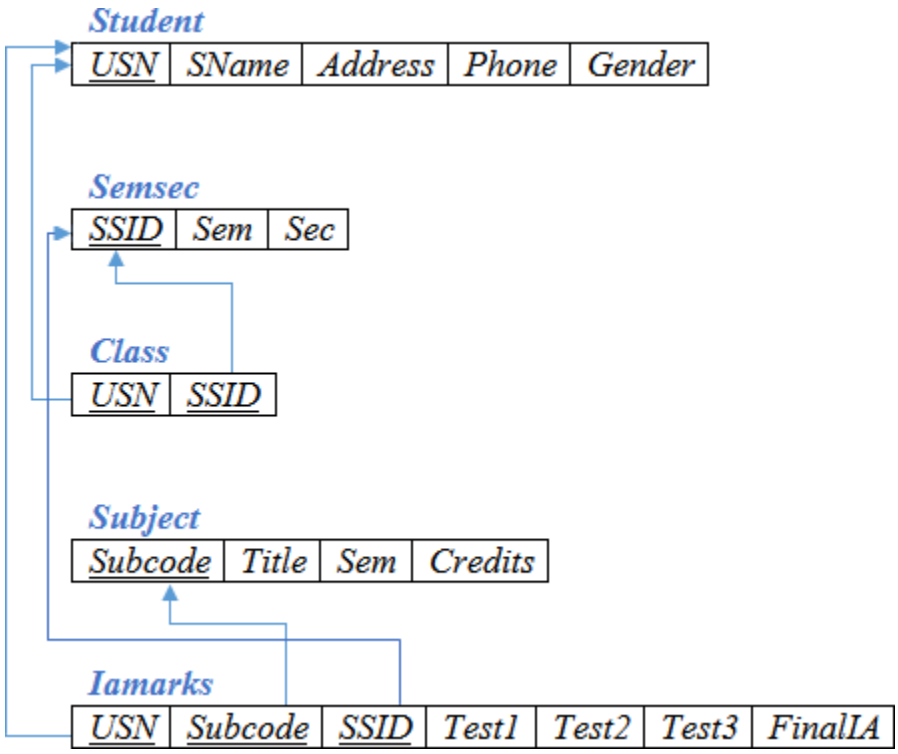



***Table creation:***

```
Create table STUDENT
(
Usn varchar(10),
Sname varchar(10),
Address varchar(10),
Phone varchar(10),
Gender varchar(1),
primary key(Usn)
);
```

```
create table SEMSEC
(
Ssid varchar(6),
Sem int,
Sec varchar(2),
primary key(Ssid)
);
```

```
create table CLASS
(
Usn varchar(10),
Ssid varchar(6),
primary key(Usn,Ssid),
foreign key(Usn) references STUDENT(Usn),
foreign key(Ssid) references SEMSEC(Ssid)
);

create table SUBJECT
(
Subcode varchar(7),
Title varchar(20),
Sem int,
Credits int,
primary key(Subcode)
);

create table IAMARKS
(
Usn varchar(10),
Subcode varchar(7),
Ssid varchar(6),
Test1 int,
Test2  int,
Test3  int,
Finalia int,
primary key(Usn,Subcode,Ssid),
foreign key(Usn) references STUDENT(Usn),
foreign key(Ssid) references SEMSEC(Ssid),
foreign key(Subcode) references SUBJECT(Subcode)
);
```

***Description of Tables:***

```
SQL> Desc STUDENT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 USN                                       NOT NULL VARCHAR2(10)
 SNAME                                              VARCHAR2(10)
 ADDRESS                                            VARCHAR2(10)
 PHONE                                              VARCHAR2(10)
 GENDER                                             VARCHAR2(1)
```

```
SQL> DESC SEMSEC;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 SSID                                      NOT NULL VARCHAR2(6)
 SEM                                                NUMBER(38)
 SEC                                                VARCHAR2(2)
```

```
SQL> DESC CLASS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 USN                                       NOT NULL VARCHAR2(10)
 SSID                                      NOT NULL VARCHAR2(6)
```

```
SQL> DESC SUBJECT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 SUBCODE                                   NOT NULL VARCHAR2(7)
 TITLE                                              VARCHAR2(20)
 SEM                                                NUMBER(38)
 CREDITS                                            NUMBER(38)
```

```
SQL> DESC IAMARKS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 USN                                       NOT NULL VARCHAR2(10)
 SUBCODE                                   NOT NULL VARCHAR2(7)
 SSID                                      NOT NULL VARCHAR2(6)
 TEST1                                              NUMBER(38)
 TEST2                                              NUMBER(38)
 TEST3                                              NUMBER(38)
 FINALIA                                            NUMBER(38)
```

***Inserting initial values into tables:***

```
insert into STUDENT values ('4al14is001','akshay','mangaluru', 8877881122,'m');
insert into STUDENT values ('4al14is002','sandhya','bengaluru', 7722829912,'f');
insert into STUDENT values ('4al14is003','trupti','bengaluru', 7712312312,'f');
insert into STUDENT values ('4al14is004','supriya','mangaluru', 8877881122,'f');
insert into STUDENT values ('4al15is010','abhay','bengaluru', 9900211201,'m');
insert into STUDENT values ('4al15is011','darshan','bengaluru', 9923211099,'m');
```

```
insert into STUDENT values ('4al15is012','ashwitha','bengaluru', 7894737377,'f');
insert into STUDENT values ('4al16is020','ajay','tumkur', 9845091341,'m');
insert into STUDENT values ('4al16is021','sanjana','kundapura', 7696772121,'f');
insert into STUDENT values ('4al16is022','krishna','bellary', 9944850121,'m');
insert into STUDENT values ('4al16is023','santosh','mangaluru', 8812332201,'m');
insert into STUDENT values ('4al17is040','lokesh','kalburgi', 9900232201,'m');
insert into STUDENT values ('4al17is041','ashika','shimoga', 9905542212,'f');
insert into STUDENT values ('4al17is042','vinayaka','bijapura', 8800880011,'m');

insert into SEMSEC values ('ise8a',8,'a');
insert into SEMSEC values ('ise8b',8,'b');
insert into SEMSEC values ('ise8c',8,'c');
insert into SEMSEC values ('ise6a',6,'a');
insert into SEMSEC values ('ise4a',4,'a');
insert into SEMSEC values ('ise4b',4,'b');
insert into SEMSEC values ('ise4c',4,'c');
insert into SEMSEC values ('ise2a',2,'a');

insert into CLASS values ('4al14is001','ise8a');
insert into CLASS values ('4al14is002','ise8a');
insert into CLASS values ('4al14is003','ise8b');
insert into CLASS values ('4al14is004','ise8c');
insert into CLASS values ('4al15is010','ise6a');
insert into CLASS values ('4al15is011','ise6a');
insert into CLASS values ('4al15is012','ise6a');
insert into CLASS values ('4al16is020','ise4a');
insert into CLASS values ('4al16is021','ise4b');
insert into CLASS values ('4al16is022','ise4c');
insert into CLASS values ('4al16is023','ise4c');
insert into CLASS values ('4al17is040','ise2a');
insert into CLASS values ('4al17is041','ise2a');
insert into CLASS values ('4al17is042','ise2a');

insert into SUBJECT values ('10is81','PW',8,4);
insert into SUBJECT values ('10is82','INS',8,4);
insert into SUBJECT values ('10isl88','PWL',8,2);
insert into SUBJECT values ('15is61','CN',6, 4);
insert into SUBJECT values ('15is62','DBMS',6,4);
insert into SUBJECT values ('15is41','DMS',4,4);
insert into SUBJECT values ('15is42','ADE',4,4);
insert into SUBJECT values ('15che21','Chemistry',2,4);
insert into SUBJECT values ('15pcd22','PCD',2,4);
```

insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al14is001', '10is81', 'ise8a',15,16,18);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al14is001', '10is82', 'ise8a',10,9,6);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al14is003', '10isl88', 'ise8b',15,5,9);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al14is004', '10is82', 'ise8c',20,15,17);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al15is011', '15is62', 'ise6a',17,10,10);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al16is022', '15is41', 'ise4c',10,9,6);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al16is023', '15is42', 'ise4c',12,11,13);
insert into IAMARKS (Usn, Subcode, Ssid, Test1, Test2, Test3) values ('4al17is042', '15pcd22', 'ise2a',9,14,13);

***Tables with values:***

```
SQL> Select * from student;

USN         SNAME      ADDRESS    PHONE      G
----------  ---------- ---------- ---------- -
4al14is001  akshay     mangaluru  8877881122 m
4al14is002  sandhya    bengaluru  7722829912 f
4al14is003  trupti     bengaluru  7712312312 f
4al14is004  supriya    mangaluru  8877881122 f
4al15is010  abhay      bengaluru  9900211201 m
4al15is011  darshan    bengaluru  9923211099 m
4al15is012  ashwitha   bengaluru  7894737377 f
4al16is020  ajay       tumkur     9845091341 m
4al16is021  sanjana    kundapura  7696772121 f
4al16is023  santosh    mangaluru  8812332201 m
4al17is040  lokesh     kalburgi   9900232201 m

USN         SNAME      ADDRESS    PHONE      G
----------  ---------- ---------- ---------- -
4al17is041  ashika     shimoga    9905542212 f
4al16is022  krishna    bellary    9944850121 m
4al17is042  vinayaka   bijapura   8800880011 m

14 rows selected.
```

```
SQL> Select * from semsec;

SSID          SEM SE
------ ---------- --
ise8a           8 a
ise8b           8 b
ise8c           8 c
ise6a           6 a
ise4a           4 a
ise4b           4 b
ise4c           4 c
ise2a           2 a

8 rows selected.
```

```
SQL> Select * from subject;

SUBCODE TITLE                          SEM    CREDITS
------- -------------------- ---------- ----------
10is81  PW                              8          4
10is82  INS                             8          4
10isl88 PWL                             8          2
15is61  CN                              6          4
15is62  DBMS                            6          4
15is41  DMS                             4          4
15is42  ADE                             4          4
15che21 Chemistry                       2          4
15pcd22 PCD                             2          4

9 rows selected.
```

```
SQL> Select * from class;

USN         SSID
---------- ------
4al14is001 ise8a
4al14is002 ise8a
4al14is003 ise8b
4al14is004 ise8c
4al15is010 ise6a
4al15is011 ise6a
4al16is020 ise4a
4al16is021 ise4b
4al16is022 ise4c
4al16is023 ise4c
4al17is040 ise2a

USN         SSID
---------- ------
4al17is041 ise2a
4al17is042 ise2a
4al15is012 ise6a

14 rows selected.
```

```
SQL> Select * from iamarks;

USN        SUBCODE SSID       TEST1      TEST2      TEST3    FINALIA
---------- ------- ------ ---------- ---------- ---------- ----------
4al14is001 10is81  ise8a          15         16         18
4al14is001 10is82  ise8a          10          9          6
4al14is003 10isl88 ise8b          15          5          9
4al14is004 10is82  ise8c          20         15         17
4al15is011 15is62  ise6a          17         10         10
4al16is022 15is41  ise4c          10          9          6
4al16is023 15is42  ise4c          12         11         13
4al17is042 15pcd22 ise2a           9         14         13

8 rows selected.
```

***Queries:***

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

```
select s.*, ss.Sem, ss.Sec
from STUDENT s, SEMSEC ss, CLASS c
where s.Usn = c.Usn and ss.Ssid = c.Ssid
and ss.Sem = 4 and ss.Sec='c';
```

```
SQL> select s.*, ss.Sem, ss.Sec
  2  from STUDENT s, SEMSEC ss, CLASS c
  3  where s.Usn = c.Usn and ss.Ssid = c.Ssid
  4  and ss.Sem = 4 and ss.Sec='c';

USN        SNAME      ADDRESS    PHONE      G          SEM SE
---------- ---------- ---------- ---------- - ---------- --
4al16is023 santosh    mangaluru  8812332201 m           4 c
4al16is022 krishna    bellary    9944850121 m           4 c
```

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

```
selects s.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;
```

```
SQL> select ss.Sem, ss.Sec, s.Gender, count(s.Gender) as count
  2  from STUDENT s, SEMSEC ss, CLASS c
  3  where s.Usn = c.Usn and ss.Ssid = c.Ssid
  4  group by ss.Sem, ss.Sec, s.Gender
  5  order by Sem;

       SEM SE G      COUNT
---------- -- - ----------
         2 a  f          1
         2 a  m          2
         4 a  m          1
         4 b  f          1
         4 c  m          2
         6 a  f          1
         6 a  m          2
         8 a  f          1
         8 a  m          1
         8 b  f          1
         8 c  f          1

11 rows selected.
```

**3. Create a view of Test1 marks of student USN '1BI15CS101' in allsubjects.**

create view Test1_view as
select Test1, Subcode
from IAMARKS
where Usn = '4al14is001';

```
SQL> create view Test1_view as
  2  select Test1, Subcode
  3  from IAMARKS
  4  where Usn = '4al14is001';

View created.

SQL> select * from Test1_view;

     TEST1 SUBCODE
---------- -------
        15 10is81
        10 10is82
```

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

```
create or replace procedure AVGMARKS is
cursor c_iamarks is
select greatest(Test1,Test2) as a, greatest(Test1,Test3) as b, greatest(Test3,Test2) as c
from IAMARKS
where Finalia is null for update;

c_a number;
c_b number;
c_c number;
c_sm number;
c_av number;

begin
openc_iamarks;
loop
fetchc_iamarks into c_a, c_b, c_c;
   exit when c_iamarks%notfound;
if (c_a != c_b) then
   c_sm:=c_a+c_b;
else
   c_sm:=c_a+c_c;
end if;

c_av:=c_sm/2;

update IAMARKS set Finalia=c_av where current of c_iamarks;

end loop;
closec_iamarks;
end;
 /
```

( ***Note***: This procedure will not update the average values in IAMARKS table until it has been called explicitly. So each time when a new entry is done to IAMARKS table,procedure AVGMARKS can be called to calculate and update the averagemarks.)

```
SQL> create or replace procedure AVGMARKS is
  2  cursor c_iamarks is
  3  select      greatest(Test1,Test2) as a, greatest(Test1,Test3) as b, greatest(Test3,Test2) as c
  4  from IAMARKS
  5  where Finalia is null for update;
  6
  7  c_a number;
  8  c_b number;
  9  c_c number;
 10  c_sm number;
 11  c_av number;
 12
 13  begin
 14  open c_iamarks;
 15  loop
 16     fetch c_iamarks into c_a, c_b, c_c;
 17        exit when c_iamarks%notfound;
 18
 19  if (c_a != c_b) then
 20         c_sm:=c_a+c_b;
 21     else
 22         c_sm:=c_a+c_c;
 23     end if;
 24
 25  c_av:=c_sm/2;
 26
 27  update IAMARKS set Finalia=c_av  where current of c_iamarks;
 28
 29  end loop;
 30  close c_iamarks;
 31  end;
 32    /

Procedure created.
```

**Below SQL code is to invoke the PL/SQL stored procedure from the command line:**

```
begin
AVGMARKS;
end;
/
```

```
SQL> begin
  2  AVGMARKS;
  3  end;
  4  /

PL/SQL procedure successfully completed.

SQL> Select * from iamarks;

USN         SUBCODE SSID         TEST1      TEST2      TEST3    FINALIA
---------- ------- ------ ---------- ---------- ---------- ----------
4al14is001 10is81  ise8a          15         16         18         17
4al14is001 10is82  ise8a          10          9          6         10
4al14is003 10isl88 ise8b          15          5          9         12
4al14is004 10is82  ise8c          20         15         17         19
4al15is011 15is62  ise6a          17         10         10         14
4al16is022 15is41  ise4c          10          9          6         10
4al16is023 15is42  ise4c          12         11         13         13
4al17is042 15pcd22 ise2a           9         14         13         14

8 rows selected.
```

**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.**

```
Select s.Usn,s.Sname,s.Address,s.Phone,s.Gender,
(case
when ia.finalia between 17 and 20 then'outstanding'
when ia.finalia between 12 and 16 then 'average'
else'weak'
end) as cat
from STUDENT s, SEMSEC ss, IAMARKS ia, SUBJECT sub
where s.Usn = ia.Usn
and ss.Ssid = ia.Ssid
and sub.Subcode = ia.Subcode
and sub.Sem = 8;
```

```
SQL> select s.Usn,s.Sname,s.Address,s.Phone,s.Gender,
  2  (case
  3  when ia.finalia between 17 and 20 then 'outstanding'
  4  when ia.finalia between 12 and 16 then 'average'
  5  else 'weak'
  6                 end) as cat
  7  from STUDENT s, SEMSEC ss, IAMARKS ia, SUBJECT sub where s.Usn = ia.Usn
  8  and ss.Ssid = ia.Ssid
  9  and sub.Subcode = ia.Subcode
 10  and sub.Sem = 8;

USN        SNAME      ADDRESS    PHONE      G CAT
---------- ---------- ---------- ---------- - -----------
4al14is001 akshay     mangaluru  8877881122 m weak
4al14is001 akshay     mangaluru  8877881122 m outstanding
4al14is003 trupti     bengaluru  7712312312 f average
4al14is004 supriya    mangaluru  8877881122 f outstanding
```

## Viva Question

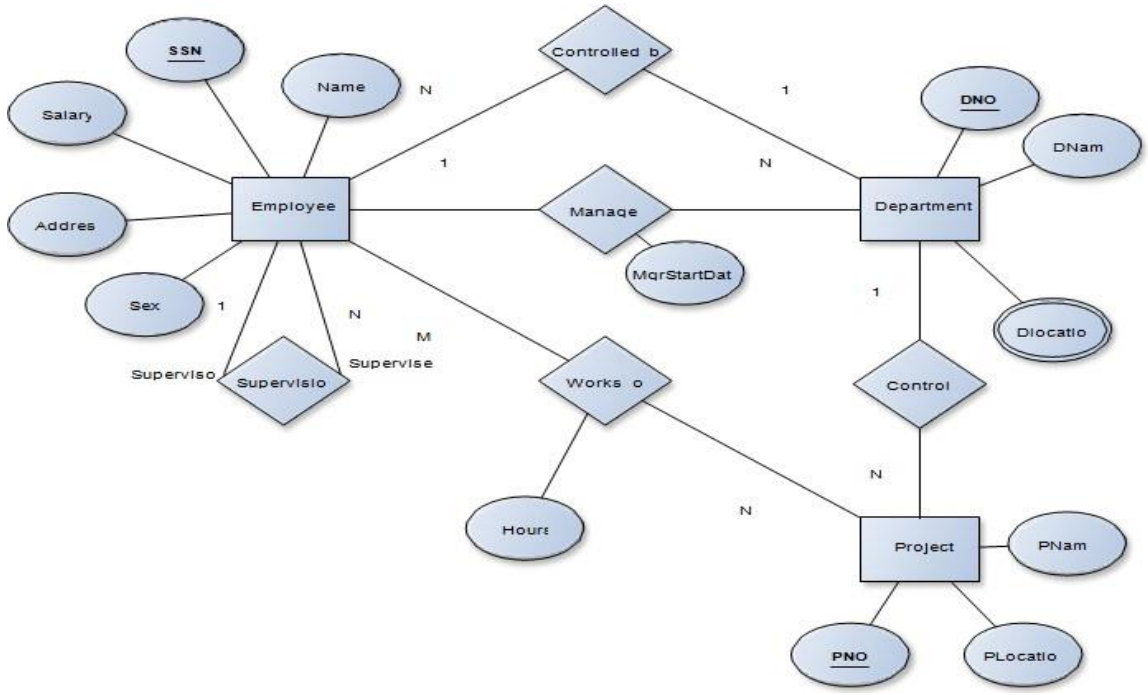
1. **Define Normalization.**
2. **Enlist the advantages of normalizing database.**
3. **What is Entity?**
4. **What is entity set?**
5. **What is Relationship?**
6. **What is Relationship Set?**
7. **What is Degree of Relationship?**
8. **Name the Degree of Relationship?**
9. **What is Data Model?**
10. **What is E-R model?**

**5. Consider the schema for Company Database:**

**EMPLOYEE (*SSN, Name, Address, Sex, Salary, Sup_Ssn, 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).**
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.**

***Entity-Relationship Diagram***



**Schema Diagram**

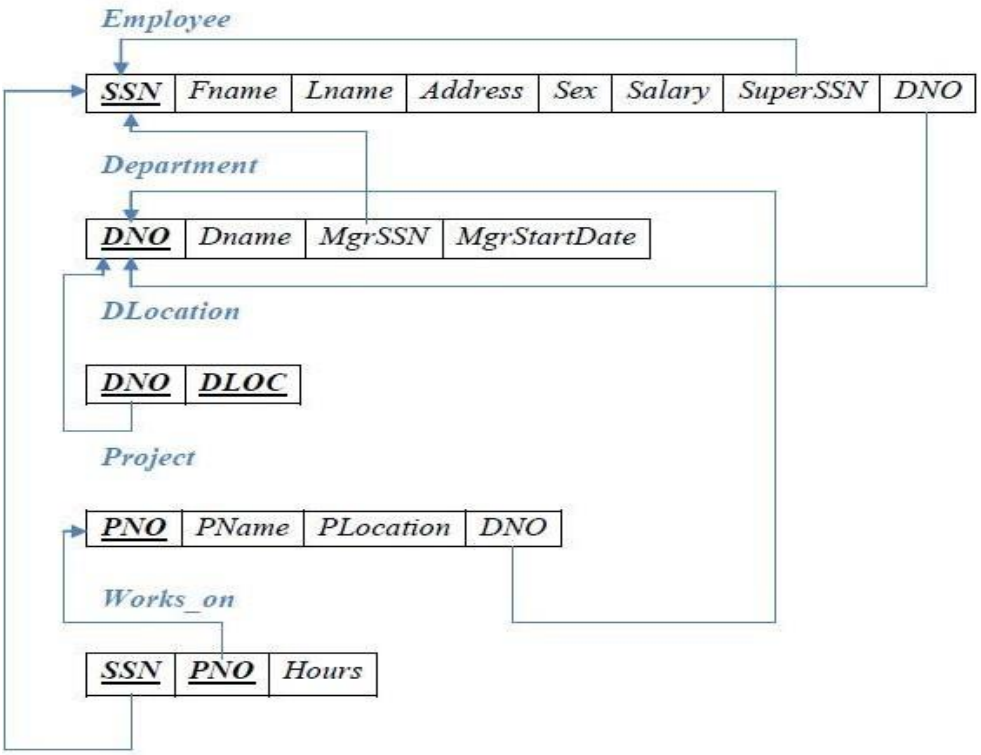



**TABLE CREATION**

**// FOR MYSQL:**

```
create table EMPLOYEE
(
Ssn varchar(8),
Fname varchar(10),
Lname varchar(10),
Address varchar(10),
Sex varchar(1),
Salary int,
Sup_Ssn varchar(8),
Dnoint,
primary key(Ssn),
foreign key(Sup_Ssn) references EMPLOYEE(Ssn)
);
```

**// FOR ORACLE:**

```
create table EMPLOYEE
(
Ssn varchar(8),
Fname varchar(10),
Lname varchar(10),
Address varchar(10),
Sex varchar(1),
Salary int,
Sup_Ssn varchar(8),
primary key(Ssn),
foreign key(Sup_Ssn) references EMPLOYEE(Ssn)
);


create table DEPARTMENT
(
Dnoint,
Dname varchar(20),
MgrSsn varchar(8),
Mgr_sdate date,
primary key(Dno),
foreign key(MgrSsn) references EMPLOYEE (Ssn)
);
```

**NOTE:** Once DEPARTMENT and EMPLOYEE tables are created we must alter EMPLOYEE table to add foreign key constraint to EMPLOYEE.Dno using sql command.

**// FOR MYSQL:**

```
alter table EMPLOYEE add constraint foreign key (Dno) references
DEPARTMENT(Dno);
```

**//ORACLE:**

```
ALTER TABLE EMPLOYEE ADD Dno INT REFERENCES DEPARTMENT(Dno);
```

```
create table DLOCATION
(
Dno int,
Dloc varchar(20),
primary key (Dno,Dloc),
foreign key(Dno) references DEPARTMENT(Dno)
);

create table PROJECT
(
Pnoint,
Pname varchar(20),
Plocation varchar(20),
Dno int,
primary key(Pno),
foreign key(Dno) references DEPARTMENT (Dno)
);

create table WORKS_ON
(
Pnoint,
Ssn varchar(8),
Hours int,
primary key(Ssn,Pno),
foreign key(Ssn) references EMPLOYEE(Ssn),
foreign key(Pno) references PROJECT(Pno)
);
```

**Table Descriptions**

```
SQL> DESC EMPLOYEE;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 SSN                                       NOT NULL VARCHAR2(8)
 FNAME                                              VARCHAR2(10)
 LNAME                                              VARCHAR2(10)
 ADDRESS                                            VARCHAR2(10)
 SEX                                                VARCHAR2(1)
 SALARY                                             NUMBER(38)
 SUP_SSN                                            VARCHAR2(8)
 DNO                                                NUMBER(38)
```

```
SQL> desc DEPARTMENT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 DNO                                       NOT NULL NUMBER(38)
 DNAME                                              VARCHAR2(20)
 MGRSSN                                             VARCHAR2(8)
 MGR_SDATE                                          DATE

SQL> desc DLOCATION;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 DNO                                       NOT NULL NUMBER(38)
 DLOC                                      NOT NULL VARCHAR2(20)

SQL> desc DLOCATION;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 DNO                                       NOT NULL NUMBER(38)
 DLOC                                      NOT NULL VARCHAR2(20)
```

```
SQL> desc PROJECT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 PNO                                       NOT NULL NUMBER(38)
 PNAME                                              VARCHAR2(20)
 PLOCATION                                          VARCHAR2(20)
 DNO                                                NUMBER(38)

SQL> desc WORKS_ON;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 PNO                                       NOT NULL NUMBER(38)
 SSN                                       NOT NULL VARCHAR2(8)
 HOURS                                              NUMBER(38)
```

**Insertion Of Values To Tables:**

***inserting values into employee table:***

```
insert into EMPLOYEE values ('alis01','john','scott','bangalore','m',2000000,NULL,NULL);
insert into EMPLOYEE values ('alis02','james','smith','kolar','m',1500000,'alis01',NULL);
insert into EMPLOYEE values ('alis03','william','baker','bangalore','m',1500000,'alis01',
NULL);
insert into EMPLOYEE values ('alis04','elson','scott','mysore','m',1500000,'alis01',NULL);
insert into EMPLOYEE values ('alis05','pavan','hegde','mangalore','m',700000,'alis02',
NULL);
insert into EMPLOYEE values ('alis06','girish','jain','mysore','m',1000000,'alis03',NULL);
insert into EMPLOYEE values ('alis07','neha','salian','bangalore','f',600500,'alis02',NULL);
insert into EMPLOYEE values ('alis08','ashika','hegde','mangalore','f',800000,'alis04',
NULL);
```

```
insert into EMPLOYEE values ('alis09','santhosh','kumar','mumbai','m',500000,'alis02',
NULL);
insert into EMPLOYEE values ('alis10','mythri','m','mysore','f',300000,'alis02',NULL);
insert into EMPLOYEE values ('alis11','nagesh','tantri','bangalore','m',900000,'alis04',
NULL);
insert into EMPLOYEE values ('alis12','vignesh','g','bangalore','m',650000,'alis02',NULL);
insert into EMPLOYEE values ('alis13','kaveri','k','mangalore','f',750000,'alis01',NULL);
```

***Inserting Values Into<u>Department</u> Table:***

**// FOR MYSQL:**

```
insert into DEPARTMENT values(1,'accounts','alis02','2001-01-01');
insert into DEPARTMENT values(2,'marketing','alis03','2016-08-11');
insert into DEPARTMENT values(3,'it','2008-03-23','alis04');
insert into DEPARTMENT values(4,'production','alis08','2012-08-10');
insert into DEPARTMENT values(5,'support','alis01','2010-03-05');
```

**//FOR ORACLE:**

```
insert into DEPARTMENT values(1,'accounts','alis02','01-jan-01');
insert into DEPARTMENT values(2,'marketing','alis03','11-aug-16');
insert into DEPARTMENT values(3,'it','alis04','23-mar-08');
insert into DEPARTMENT values(4,'production','alis08','10-aug-12');
insert into DEPARTMENT values(5,'support','alis01','05-mar-10');
```

***Update Entries Of Employee Table To Fill Missing DNO:***

```
update EMPLOYEE set Dno=5 where Ssn='alis01';
update EMPLOYEE set Dno=1 where Ssn='alis02';
update EMPLOYEE set Dno=2 where Ssn='alis03';
update EMPLOYEE set Dno=3 where Ssn='alis04';
update EMPLOYEE set Dno=1 where Ssn='alis05';
update EMPLOYEE set Dno=2 where Ssn='alis06';
update EMPLOYEE set Dno=1 where Ssn='alis07';
update EMPLOYEE set Dno=4 where Ssn='alis08';
update EMPLOYEE set Dno=1 where Ssn='alis09';
update EMPLOYEE set Dno=1 where Ssn='alis10';
update EMPLOYEE set Dno=3 where Ssn='alis11';
update EMPLOYEE set Dno=1 where Ssn='alis12';
update EMPLOYEE set Dno=5 where Ssn='alis13';
```

***Inserting Values Into DLOCATION Table:***

```
insert into DLOCATION values(1,'bangalore');
insert into DLOCATION values(2,'bangalore');
insert into DLOCATION values(3,'bangalore');
insert into DLOCATION values(1,'mangalore');
insert into DLOCATION values(3,'mangalore');
insert into DLOCATION values(4,'mysore');
insert into DLOCATION values(5,'hubli');
```

***Inserting Values Into PROJECT Table:***

```
insert into PROJECT values(100,'market_s','bangalore',1);
insert into PROJECT values(101,'stocks','bangalore',1);
insert into PROJECT values(102,'GST_b','bangalore',1);
insert into PROJECT values(103,'T_cards','bangalore',2);
insert into PROJECT values(104,'Jio_money','bangalore',2);
insert into PROJECT values(105,'iot','bangalore',3);
insert into PROJECT values(106,'Pro_xl','bangalore',4);
insert into PROJECT values(107,'project_j','bangalore',5);
insert into PROJECT values(108,'project_d','bangalore',5);
```

***Inserting Values Into WORKS_ON Table:***

```
insert into WORKS_ON(Pno,Ssn,Hours) values(100,'alis02',20);
insert into WORKS_ON(Pno,Ssn,Hours) values(100,'alis09',30);
insert into WORKS_ON(Pno,Ssn,Hours) values(101,'alis10',10);
insert into WORKS_ON(Pno,Ssn,Hours) values(101,'alis02',34);
insert into WORKS_ON(Pno,Ssn,Hours) values(102,'alis12',25);
insert into WORKS_ON(Pno,Ssn,Hours) values(102,'alis07',65);
insert into WORKS_ON(Pno,Ssn,Hours) values(103,'alis03',34);
insert into WORKS_ON(Pno,Ssn,Hours) values(104,'alis06',22);
insert into WORKS_ON(Pno,Ssn,Hours) values(105,'alis11',12);
insert into WORKS_ON(Pno,Ssn,Hours) values(107,'alis13',34);
insert into WORKS_ON(Pno,Ssn,Hours) values(107,'alis08',63);
insert into WORKS_ON(Pno,Ssn,Hours) values(107,'alis01',27);
insert into WORKS_ON(Pno,Ssn,Hours) values(108,'alis13',10);
insert into WORKS_ON(Pno,Ssn,Hours) values(108,'alis08',30);
insert into WORKS_ON(Pno,Ssn,Hours) values(108,'alis05',20);
insert into WORKS_ON(Pno,Ssn,Hours) values(105,'alis04',12);
```

```
SQL> select * from EMPLOYEE;

SSN      FNAME      LNAME      ADDRESS    S     SALARY SUP_SSN         DNO
-------- ---------- ---------- ---------- - ---------- -------- ----------
alis01   john       scott      bangalore  m    2000000                   5
alis02   james      smith      kolar      m    1500000 alis01            1
alis03   william    baker      bangalore  m    1500000 alis01            2
alis04   elson      scott      mysore     m    1500000 alis01            3
alis05   pavan      hegde      mangalore  m     700000 alis02            1
alis06   girish     jain       mysore     m    1000000 alis03            2
alis07   neha       salian     bangalore  f     600500 alis02            1
alis08   ashika     hegde      mangalore  f     800000 alis04            4
alis09   santhosh   kumar      mumbai     m     500000 alis02            1
alis10   mythri     m          mysore     f     300000 alis02            1
alis11   nagesh     tantri     bangalore  m     900000 alis04            3

SSN      FNAME      LNAME      ADDRESS    S     SALARY SUP_SSN         DNO
-------- ---------- ---------- ---------- - ---------- -------- ----------
alis12   vignesh    g          bangalore  m     650000 alis02            1
alis13   kaveri     k          mangalore  f     750000 alis01            5

13 rows selected.
```

```
SQL> select * from DEPARTMENT;

       DNO DNAME                MGRSSN   MGR_SDATE
---------- -------------------- -------- ---------
         1 accounts             alis02   01-JAN-01
         2 marketing            alis03   11-AUG-16
         3 it                   alis04   23-MAR-08
         4 production           alis08   10-AUG-12
         5 support              alis01   05-MAR-10
```

```
SQL> select * from DLOCATION;

       DNO DLOC
---------- --------------------
         1 bangalore
         1 mangalore
         2 bangalore
         3 bangalore
         3 mangalore
         4 mysore
         5 hubli

7 rows selected.
```

```
SQL> select * from PROJECT;

       PNO PNAME                PLOCATION                DNO
---------- -------------------- -------------------- ----------
       100 market_s             bangalore                 1
       101 stocks               bangalore                 1
       102 GST_b                bangalore                 1
       103 T_cards              bangalore                 2
       104 Jio_money            bangalore                 2
       105 iot                  bangalore                 3
       106 Pro_xl               bangalore                 4
       107 project_j            bangalore                 5
       108 project_d            bangalore                 5

9 rows selected.
```

```
SQL>  select * from WORKS_ON;

       PNO SSN             HOURS
---------- -------- ----------
       100 alis02           20
       100 alis09           30
       101 alis10           10
       101 alis02           34
       102 alis12           25
       102 alis07           65
       103 alis03           34
       104 alis06           22
       105 alis11           12
       107 alis13           34
       107 alis08           63

       PNO SSN             HOURS
---------- -------- ----------
       107 alis01           27
       108 alis13           10
       108 alis08           30
       108 alis05           20
       105 alis04           12

16 rows selected.
```

**QUERIES:**

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

```
(select p.Pno
from PROJECT p, DEPARTMENT d, EMPLOYEE e
where p.Dno=d.Dno and d.MgrSsn=e.Ssn
      and e.Lname='scott')
UNION
(select 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
----------
       105
       107
       108
```

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

```
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
---------- ---------- ----------
elson      scott         1650000
nagesh     tantri         990000
```

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

```
select sum(e.Salary) as total_salary, max(e.Salary) as max_salary, min(e.Salary) as
min_salary, avg(e.Salary) as average_salary
from EMPLOYEE e, DEPARTMENT d
where e.Dno=d.Dno
and
d.dname='accounts';
```

```
TOTAL_SALARY MAX_SALARY MIN_SALARY AVERAGE_SALARY
------------ ---------- ---------- --------------
     4250500    1500000     300000     708416.667
```

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

**FOR MYSQL:**

```
select e.Fname,e.Lname from EMPLOYEE e
where NOT EXISTS(select * from WORKS_ON w where w.Pno IN (select p.Pno from
PROJECT p where p.Dno=5)
and NOT EXISTS (select * from WORKS_ON o where o.Ssn=e.Ssn and o.Pno=w.Pno));
```

**FOR ORACLE:**

```
select e.fname, e.lname
from employee e
where NOT EXISTS((select Pno from project where Dno='5')
                        minus (select Pno from works_on where e.Ssn=Ssn));
```

```
FNAME      LNAME
---------- ----------
ashika     hegde
kaveri     k
```

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

```
select d.Dno, count (*)
from DEPARTMENT d, EMPLOYEE e
where d.Dno=e.Dno
ande.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(*)
---------- ----------
         1          4
```

## Viva question

1. **What is Mapping Cardinalities**
2. **What are the different types of Mapping**
3. **What is One-to-one mapping?**
4. **What is One-to-many mapping?**
5. **What is Many-to-one mapping**?
6. **What is Many-to-many mapping?**
7. **What is DDL?**
8. **What is DML?**
9. **What is DCL?**