K. L. E. SOCIETY'S

K. L. E. INSTITUTE OF TECHNOLOGY,

Gokul, Hubballi-580 030

Dept of Computer Science & Engg.

Laboratory Manual

V SEMESTER

DBMS LABORATORY WITH MINI PROJECT LABORATORY

(18CSL58)

2020-21

Prepared By	Pradeep Surasura	
Approved By	Prof. Yerriswamy T.	

HOD

List of Experiments

Part A: SQL Programming

PROBLEM 1

Consider the following schema for a Library Database:
BOOK(Book_id, Title, Publisher_Name, Pub_Year)

BOOK_AUTHORS(Book_id, Author_Name)

PUBLISHER(Name, Address, Phone)

BOOK_COPIES(Book_id, Branch_id, No-of_Copies)

BOOK_LENDING(Book_id, Branch_id, Card_No, Date_Out, Due_Date)

LIBRARY_BRANCH(Branch_id, Branch_Name, Address)

Write SQL queries to

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

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

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

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

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

SQL> create table book_authors (bookid references book(bookid) on delete set null, author_name varchar(10));

Table created.

SQL> create table publisher(name varchar(10) primary key, address varchar(20), phone number(10));

Table created.

SQL> create table book(bookid int primary key, title varchar(10),

pub_name references publisher(name) on delete set null, pub_year number(4));

Table created.

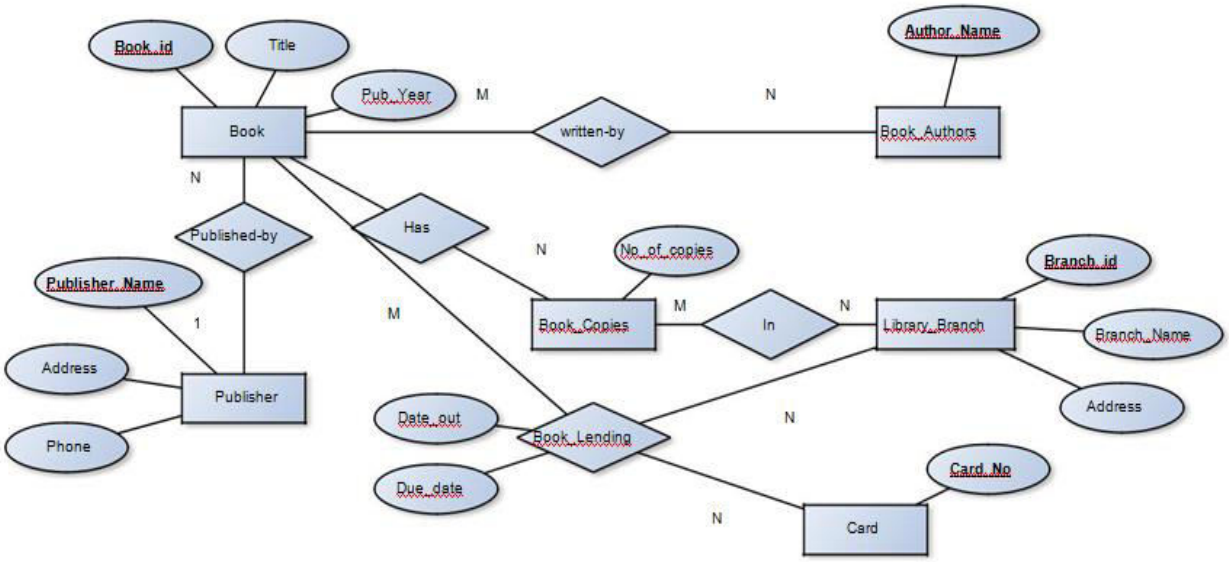
SQL> create table library_branch(programme_idint primary key, programme_name varchar(10), address varchar(20));

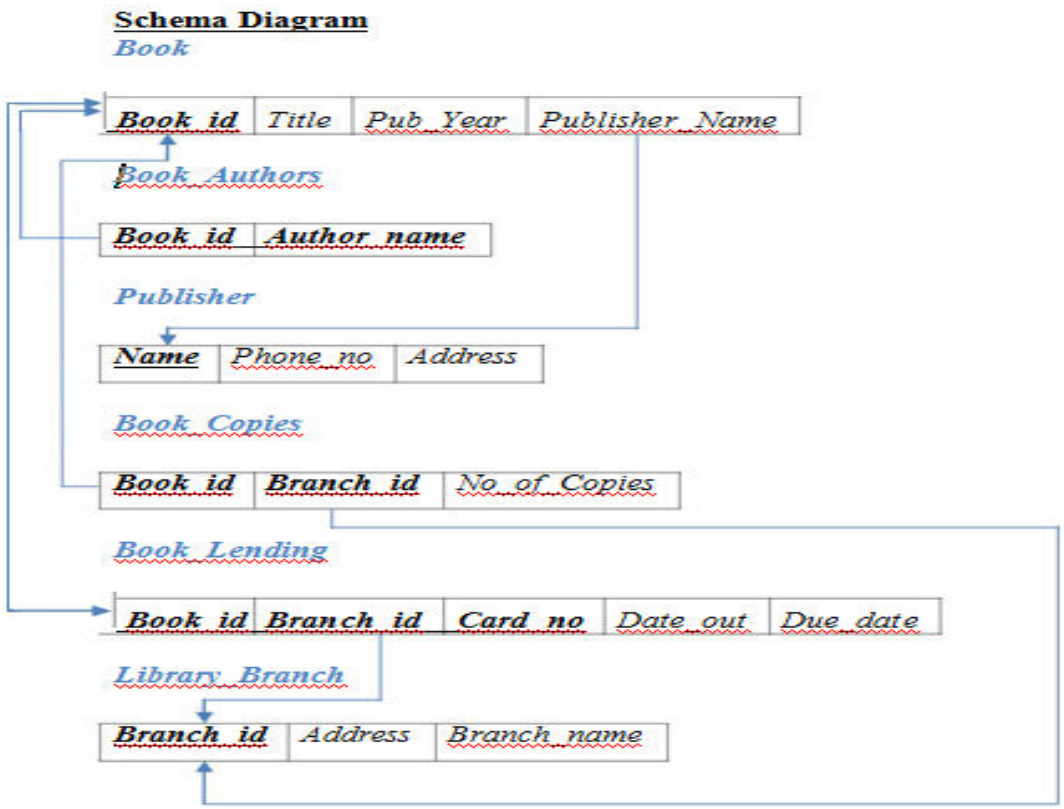
Table created.

SQL> create table book_copies(bookid references book(bookid) on delete set null, programme_idreferences library_branch(programme_id) on delete set null, no_of_copies int);

Table created.

Entity-Relationship Diagram





SQL> create table book_lending(bookid references book(bookid) on delete set null,

programme_id references library_branch(programme_id) on delete set null,

card_no int, date_out date, due_date date, primary key(bookid, programme_id, card_no));

Table created.

SQL> insert into book_authors values('&bookid','&author_name');

Enter value for bookid: 1001

Enter value for author_name: Herbert

old 1: insert into book_authors values('&bookid','&author_name')

new 1: insert into book_authors values('1001','Herbert')

1 row created.

SQL> select * from book_authors;

BOOKID AUTHOR_NAM

--------- ----------

1001 Herbert

1002 Padmareddy

1003 Reily

1004 Kottur

1005 Navathe

1006 Raghu

6 rows selected.

SQL> insert into publisher values ('&name','&address','&phone');

Enter value for name: Excellent

Enter value for address: hubli

Enter value for phone: 1234567891

old 1: insert into publisher values ('&name','&address','&phone')

new 1: insert into publisher values ('Excellent','hubli','1234567891')

1 row created.

SQL> select * from publisher;

NAME ADDRESS PHONE

---------- -------------------- ---------

Excellent hubli 1.235E+09

Pearson sydney 3.217E+09

global india 5.214E+09

springer usa 3.625E+09

ieee india 3.217E+09

SQL> insert into book values('&bookid','&title','&pub_name','&pub_year');

Enter value for bookid: 1001

Enter value for title: ds

Enter value for pub_name: global

```
Enter value for pub_year: 2015
old   1: insert into book values('&bookid','&title','&pub_name','&pub_year')
new   1: insert into book values('1001','ds','global','2015')
1 row created.
SQL> select * from book;
  BOOKID TITLE     PUB_NAME    PUB_YEAR
--------- ---------- ---------- ---------
    1001 ds        global        2015
    1002 java      ieee          2016
    1003 cpp       Pearson       2014
    1004 database  Excellent     2000
    1005 oops      springer      2011
SQL> insert into library_branch values('&branch_id','&branch_name','&address');
Enter value for branch_id: 01
Enter value for branch_name: kleit
Enter value for address: hubli
old   1: insert into library_branch values('&branch_id','&branch_name','&address')
new   1: insert into library_branch values('01','kleit','hubli')
1 row created.
SQL> select * from library_branch;
BRANCH_ID BRANCH_NAM ADDRESS
--------- ---------- --------------------
      1 kleit      hubli
      2 gokul      india
      3 main       usa
      4 sydney     australia
```

```
      5 uk        London

SQL> insert into book_copies values('&bookid','&branch_id','&no_of_copies');

Enter value for bookid: 1001

Enter value for branch_id: 01

Enter value for no_of_copies: 10

old   1: insert into book_copies values('&bookid','&branch_id','&no_of_copies')

new   1: insert into book_copies values('1001','01','10')

1 row created.

SQL> select * from book_copies;

  BOOKID BRANCH_ID NO_OF_COPIES

--------- --------- ------------

    1001         1         10

    1002         2         20

    1001         3         10

    1005         4         10

    1004         2          5

    1002         4          5


6 rows selected.

SQL> insert into  book_lending values('&bookid','&branch_id','&card_no','&date_out','&due_date');

Enter value for bookid: 1001

Enter value for branch_id: 01

Enter value for card_no: 7001

Enter value for date_out: 10-aug-2017

Enter value for due_date: 28-aug-2017

old   1: insert into  book_lending values('&bookid','&branch_id','&card_no','&date_out','&due_date')
```

```
new   1: insert into  book_lending values('1001','01','7001','10-aug-2017','28-aug-2017')

1 row created.

SQL> select * from book_lending;

  BOOKID BRANCH_ID   CARD_NO DATE_OUT  DUE_DATE

--------- --------- --------- --------- ---------

    1001         1      7001 10-AUG-17 28-AUG-17

    1001         5      7002 20-AUG-17 05-SEP-17

    1005         4      7003 05-SEP-17 20-SEP-17

    1004         3      7004 10-SEP-17 25-SEP-17

    1003         4      7005 11-SEP-17 26-SEP-17
```

Ldq1.sql

```
SQL>  select a.author_name,b.bookid,b.title, b.pub_name,bc.no_of_copies as branch_copies
  2  from book b, book_authors a, book_copies bc
  3  where a.bookid=bc.bookid and a.bookid=b.bookid and
  4  bc.branch_id in (select branch_id from library_branch
  5  group by branch_id);

AUTHOR_NAM    BOOKID TITLE      PUB_NAME   BRANCH_COPIES

---------- --------- ---------- ---------- -------------

Herbert       1001 ds        global              10

Herbert       1001 ds        global              10

Padmareddy    1002 java      ieee                 5

Padmareddy    1002 java      ieee                20

Kottur        1004 database  Excellent            5

Navathe       1005 oops      springer            10

6 rows selected.
```

Ldq2:

```
select card_no, count(*)
 from book_lending
 where date_out between '01-jan-2017' and '30-sep-2017'
 group by card_no
 having count(*) >=1;
```

```
CARD_NO  COUNT(*)
-------- ---------
   7004        1
   7001        1
   7002        1
   7003        1
   7005        1
   7006        1
```

```
SQL> delete from book where bookid=1004;
1 row deleted.
SQL> select * from book;
  BOOKID TITLE      PUB_NAME    PUB_YEAR
--------- ---------- ---------- ---------
    1001 ds        global         2015
    1002 java      ieee           2016
    1003 cpp       Pearson        2014
    1005 oops      springer       2011
SQL> update book_lending set bookid=1002 where bookid=1004;
1 row updated.
SQL> select * from  book_lending;
```

```
  BOOKID BRANCH_ID   CARD_NO DATE_OUT  DUE_DATE
--------- --------- --------- --------- ---------
    1001         1      7001 10-AUG-17 28-AUG-17
    1001         5      7002 20-AUG-17 05-SEP-17
    1005         4      7003 05-SEP-17 20-SEP-17
    1002         3      7004 10-SEP-17 25-SEP-17
    1003         4      7005 11-SEP-17 26-SEP-17
    1001         1      7006 01-FEB-17 20-FEB-17
6 rows selected.
SQL> create view Display_books as  select b.bookid,b.title, bc.no_of_copies, bc.branch_id
  2   from book b,book_copies bc
  3   where b.bookid=bc.bookid;
View created.
SQL> select * from Display_books;
  BOOKID TITLE      NO_OF_COPIES BRANCH_ID
--------- ---------- ------------ ---------
    1001 ds                 10         1
    1002 java               20         2
    1001 ds                 10         3
    1005 oops               10         4
    1004 database            5         2
    1002 java                5         4
6 rows selected.
SQL>create table book_part (bookid int, title varchar(10),
pub_name varchar(10),
pub_year int) PARTITION BY RANGE (pub_year)
```

```
(PARTITION p1 VALUES LESS THAN (1990),
 PARTITION p2 VALUES LESS THAN  (2000),
 PARTITION p3 VALUES Less THAN (2010)
 );
SQL> insert into book_part values (1005,'da','ieee',2009);
1 row created.
SQL> select * from book_part;
   BOOKID TITLE      PUB_NAME    PUB_YEAR
--------- ---------- ---------- ---------
    1001 da         ieee          1988
    1001 da         ieee          1988
    1001 da         ieee          1988
    1001 da         ieee          1988
    1001 da         ieee          1988
    1001 da         ieee          1990
    1001 da         ieee          1990
    1001 da         ieee          1990
    1001 da         ieee          1990
    1005 da         ieee          2000
    1005 da         ieee          2000
    1005 da         ieee          2000
    1005 da         ieee          2000
    1005 da         ieee          2009
    1005 da         ieee          2009
    1005 da         ieee          2009
    1001 da         ieee          2005
```

1001 da ieee 2005

1001 da ieee 2005

1001 da ieee 2005

20 rows selected.

PROBLEM-2

The following relations keep track of airline flight information:

Consider the following schema for Order Database:

SALESMAN(Salesman_id, Name, City, Commission)

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

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

Write SQL queries to

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

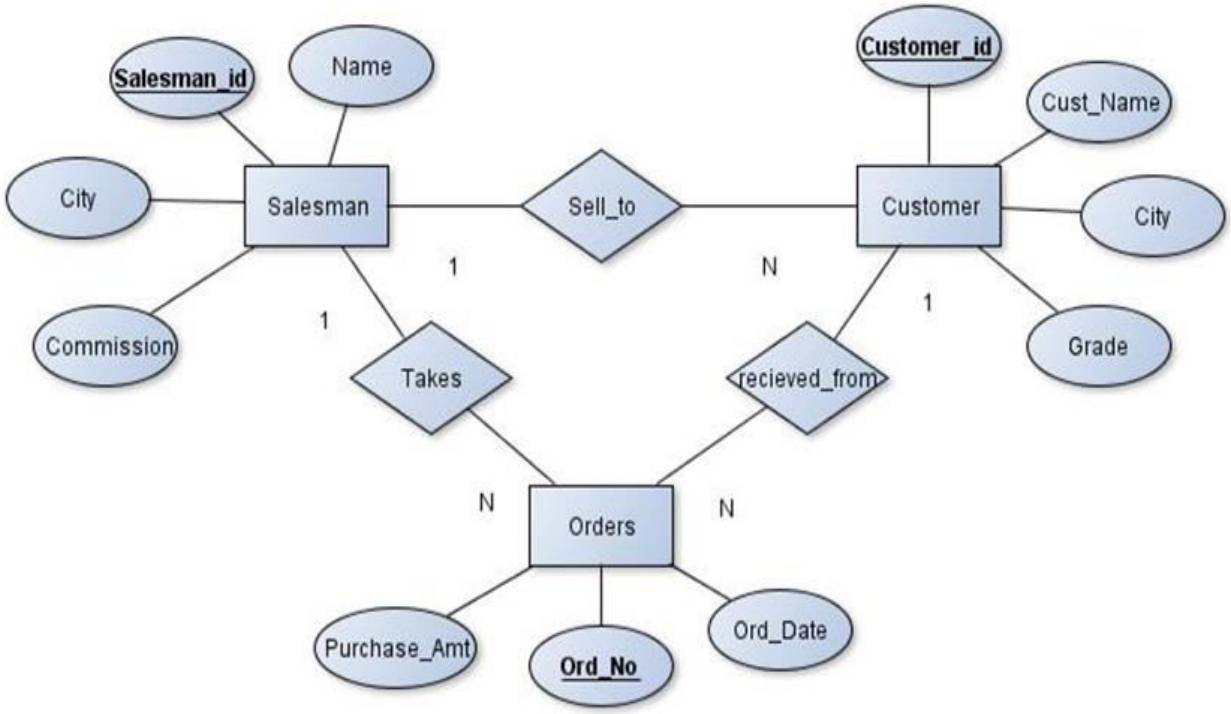
2. Find the name and numbers of all salesman who had more than one customer.

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

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

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

Entity-Relationship Diagram



Schema Diagram

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*

SQL> create table salesman(salesman_id int primary key, sname varchar(10), city varchar(10), commi

ssion int);

Table created.

SQL> create table customer(customer_id int primary key, cust_name varchar(10), city varchar(10), grades INT, sales_id references salesman(salesman_id) on delete cascade);

Table created.

SQL> create table orders(ord_no int primary key, purchase_amt int, ord_date date, cust_id references customer(customer_id) on delete cascade, sales_id references salesman(salesman_id) on delete cascade);

Table created.

SQL> insert into customer values('&CUSTOMER_ID','&CUST_NAME','&city','&grades','&SALES_ID');

Enter value for customer_id: 7001

Enter value for cust_name: arora

Enter value for city: dharwad

Enter value for grades: c

Enter value for sales_id: 4002

old 1: insert into customer values('&CUSTOMER_ID','&CUST_NAME','&city','&grades','&SALES_ID')

new 1: insert into customer values('7001','arora','dharwad','c','4002')

1 row created.

SQL> select * from salesman;

SALESMAN_ID SNAME CITY COMMISSION

---------- ---------- ---------- ----------

4001 Raghu Hubli 5

4002 vijay gadag 8

4003 sam mumbai 12

4004 henry bangalore 13

4005 moses delhi 18

```
4006 gani      hubli           8
4007 jery      belgaum         7
7 rows selected.
SQL> insert into customer values('&CUSTOMER_ID','&CUST_NAME','&city','&grades','&SALES_ID');
Enter value for customer_id: 7002
Enter value for cust_name: mishra
Enter value for city: bangalore
Enter value for grades: 5
Enter value for sales_id: 4003
old   1: insert into customer values('&CUSTOMER_ID','&CUST_NAME','&city','&grades','&SALES_ID')
new   1: insert into customer values('7002','mishra','bangalore','b','4003')
1 row created.
SQL> SELECT * FROM CUSTOMER;
CUSTOMER_ID CUST_NAME  CITY       SALES_ID    GRADES
----------- ---------- ---------- --------- ---------
      7001 arora      dharwad        4002        3
      7002 mishra     bangalore      4003        5
      7003 singh      delhi          4003        4
      7004 agarwal    chennai        4004        2
      7005 koti       hubli          4005        5
      7006 reddy      bangalore      4003        3
      7007 jevan      belgaum        4002        4
7 rows selected.
SQL> insert into orders values('&ord_no','&purchase_amt','&ord_date','&cust_id','&sales_id');
Enter value for ord_no: 1001
Enter value for purchase_amt: 800
```

```
Enter value for ord_date: 12-sep-2017
Enter value for cust_id: 7004
Enter value for sales_id: 4003
old   1: insert into orders values('&ord_no','&purchase_amt','&ord_date','&cust_id','&sales_id')
new   1: insert into orders values('1001','800','12-sep-2017','7004','4003')
1 row created.
SQL> select * from orders;
  ORD_NO PURCHASE_AMT ORD_DATE    CUST_ID  SALES_ID
--------- ------------ --------- --------- ---------
    1001         800 12-SEP-17      7004      4003
    1002         900 11-SEP-17      7002      4001
    1003        1200 09-SEP-17      7005      4005
    1004        1500 02-SEP-17      7001      4001
    1005        2000 03-SEP-17      7003      4001
    1006        2400 04-SEP-17      7002      4005
    1007        2600 05-SEP-17      7004      4001
7 rows selected.
SQL> select count(*),GRADES   from customer
  2   where grades >  ( select avg(grades)from customer
  3   where city='bangalore') group by GRADES;
 COUNT(*)    GRADES
--------- ---------
       2         5
SQL> select salesman_id,sname,count(*)
    from customer,salesman
    where sales_id=salesman_id
```

```
   group by salesman_id,sname
   having count(*)>2;
SALESMAN_ID SNAME      COUNT(*)
----------- ---------- ---------
      4002 vijay            4
      4003 sam              3
SQL>  ( select s.sname
  from salesman s, customer c
    where s.city=c.city and
   sales_id=salesman_id )
    UNION ( select s.sname
   from salesman s, customer c
    where s.city!=c.city and
    sales_id=salesman_id);
SNAME
----------
Raghu
henry
moses
sam
vijay
SQL> select * from orders;
  ORD_NO PURCHASE_AMT ORD_DATE    CUST_ID  SALES_ID
--------- ------------ --------- --------- ---------
    1001         800 12-SEP-17      7004     4003
    1002         900 11-SEP-17      7002     4001
```

```
   1003       1200 09-SEP-17     7005     4005

   1004       1500 02-SEP-17     7001     4001

   1005       2000 03-SEP-17     7003     4001

   1006       2400 04-SEP-17     7002     4005

   1007       2600 05-SEP-17     7004     4001

   1008       2500 11-SEP-17     7002     4001

   1009       1200 11-SEP-17     7005     4001

   1010       1200 11-SEP-17     7004     4001

   1011       1000 11-SEP-17     7002     4002

11 rows selected.

SQL>  select s.SALESMAN_ID,s.sname, s.city, count(*) as highest_orders

  2    from customer c, salesman s

  3    where s.SALESMAN_ID=c.SALES_ID

  4    group by s.SALESMAN_ID,s.sname, s.city

  5   having count(*) = (   select max(count(sales_id))   from orders

  6   where ord_date='&ord_date'

  7   group by sales_id);

Enter value for ord_date: 11-SEP-2017

old   6:  where ord_date='&ord_date'

new   6:  where ord_date='11-SEP-2017'

SALESMAN_ID SNAME      CITY       HIGHEST_ORDERS

----------- ---------- ---------- --------------

      4002 vijay      gadag                 4

SQL> create view highest_order as select s.SALESMAN_ID,s.sname, s.city, count(*) as highest_orders

  2  from customer c, salesman s

  3  where s.SALESMAN_ID=c.SALES_ID
```

```
 4  group by s.SALESMAN_ID,s.sname, s.city
 5  having count(*) = (   select max(count(sales_id))   from orders
 6  where ord_date='&ord_date'
 7  group by sales_id);
Enter value for ord_date: 11-SEP-2017
old   6: where ord_date='&ord_date'
new   6: where ord_date='11-SEP-2017'
View created.
SQL> select * from highest_order;
SALESMAN_ID SNAME      CITY       HIGHEST_ORDERS
----------- ---------- ---------- --------------
      4002 vijay      gadag                 4
SQL> delete from salesman where SALESMAN_ID=1000;
1 row deleted.
SQL> select * from salesman;
SALESMAN_ID SNAME      CITY       COMMISSION
----------- ---------- ---------- ----------
      4001 Raghu      hubli              5
      4002 vijay      gadag              8
      4003 sam        mumbai            12
      4004 henry      bangalore         13
      4005 moses      delhi             18
      4006 gani       hubli              8
      4007 jery       belgaum            7
7 rows selected.
SQL> select * from orders;
```

```
ORD_NO PURCHASE_AMT ORD_DATE    CUST_ID  SALES_ID
--------- ---------- --------- --------- ---------
    1001        800 12-SEP-17      7004      4003
    1002        900 11-SEP-17      7002      4001
    1003       1200 09-SEP-17      7005      4005
    1004       1500 02-SEP-17      7001      4001
    1005       2000 03-SEP-17      7003      4001
    1006       2400 04-SEP-17      7002      4005
    1007       2600 05-SEP-17      7004      4001
7 rows selected.
SQL> select * from customer;
CUSTOMER_ID CUST_NAME  CITY       SALES_ID    GRADES
----------- ---------- ---------- --------- ---------
      7001 arora      dharwad        4002        3
      7002 mishra     bangalore      4003        5
      7003 singh      delhi          4003        4
      7004 agarwal    chennai        4004        2
      7005 koti       hubli          4005        5
      7006 reddy      bangalore      4003        3
      7007 jevan      belgaum        4002        4
      7008 motilal    hubli          4002        4
      7009 kandkur    hubli          4002        4
      7010 jtk        hubli          4001        4
      7011 giri       hubli          4001        4
11 rows selected.
```

PROBLEM 3

Consider the schema for Movie Database:
ACTOR(Act_id, Act_Name, Act_Gender)

DIRECTOR(Dir_id, Dir_Name, Dir_Phone)

MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)

MOVIE_CAST(Act_id, Mov_id, Role) RATING(Mov_id, Rev_Stars)

Write SQL queries to

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

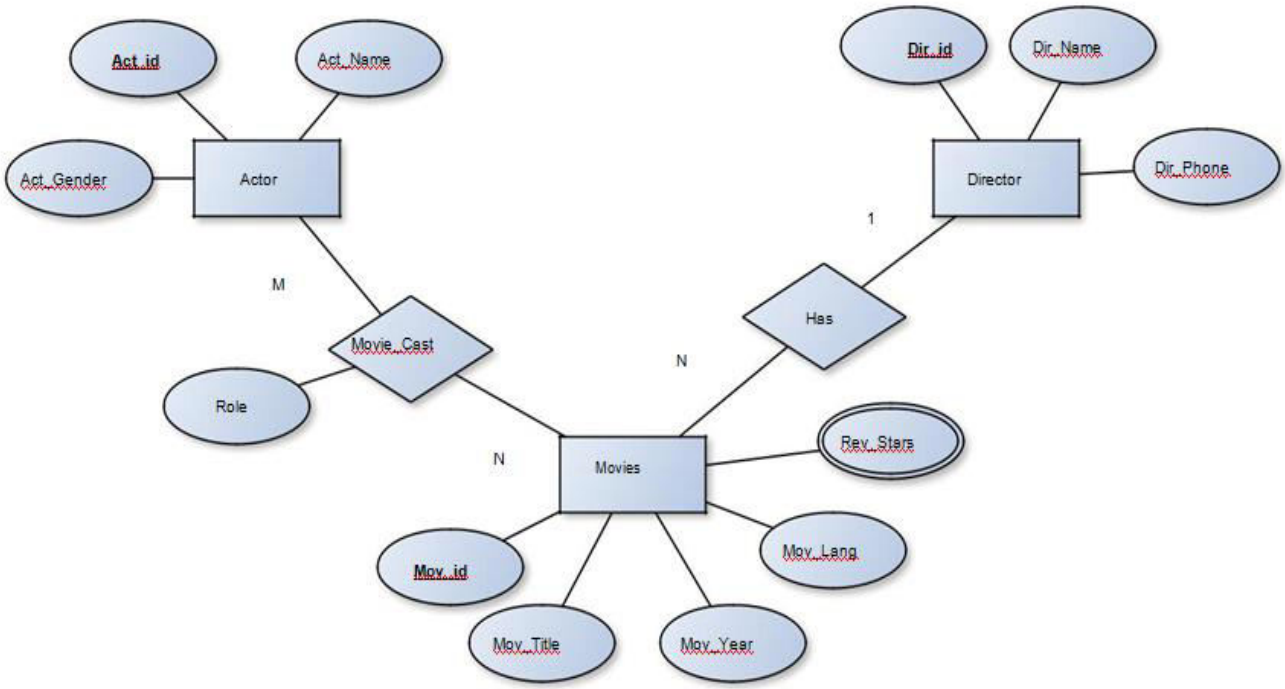
2. Find the movie names where one or more actors acted in two or more movies.

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

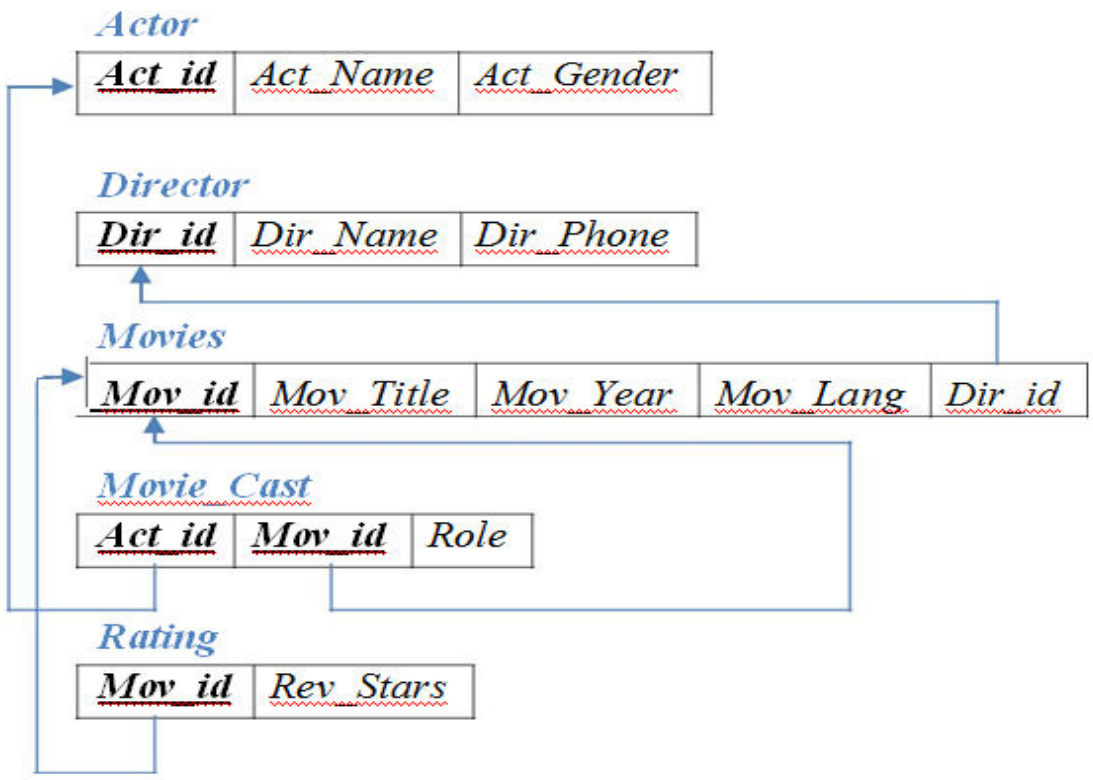
4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

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

Entity-Relationship Diagram



Schema Diagram



SQL> create table actor(act_id int primary key, act_name varchar(10) not null, act_gender varchar(7),);

Table created.

SQL> create table director (dir_id int primary key, dir_name varchar(10), dir_phone number(10));

Table created.

SQL> create table movies (mov_id int primary key, mov_title varchar(15), mov_year number(4), mov_lan

g varchar(10), dir_id references director(DIR_ID) on delete cascade);

Table created.

SQL> create table movie_cast (act_id references actor(ACT_ID) on delete cascade, mov_id references m

ovies(mov_id) on delete cascade, role varchar(10));

Table created.

SQL> create table rating(mov_id references movies(mov_id) on delete cascade, rev_stars int);

Table created.

SQL> insert into actor values('&act_id','&act_name','&act_gender');

Enter value for act_id: 1001

Enter value for act_name: johnny

Enter value for act_gender: male

old 1: insert into actor values('&act_id','&act_name','&act_gender')

new 1: insert into actor values('1001','johnny','male')

1 row created.

SQL> insert into director values('&dir_id','&dir_name','&dir_phone');

Enter value for dir_id: 5001

Enter value for dir_name: hitchcock

Enter value for dir_phone: 8529631472

old 1: insert into director values('&dir_id','&dir_name','&dir_phone')

```
new   1: insert into director values('5001','hitchcock','8529631472')

1 row created.

SQL> insert into movies values('&mov_id','&MOV_TITLE','&MOV_YEAR','&MOV_LANG','&DIR_ID');

Enter value for mov_id: 7001

Enter value for mov_title: titanic

Enter value for mov_year: 2012

Enter value for mov_lang: english

Enter value for dir_id: 5002

old   1: insert into movies values('&mov_id','&MOV_TITLE','&MOV_YEAR','&MOV_LANG','&DIR_ID')

new   1: insert into movies values('7001','titanic','2012','english','5002')

1 row created.

SQL> select * from actor;

  ACT_ID ACT_NAME   ACT_GE

--------- ---------- ------

    1001 johnny     male

    1002 salman     male

    1003 amitabh    male

    1004 barrymore  female

    1005 umaturman  female

    1006 kaira      female

    1007 zareen     female

    1008 shahid     male

    1009 ajay       male

    1010 kajol      female

10 rows selected.

SQL> select * from director;
```

```
  DIR_ID DIR_NAME   DIR_PHONE
--------- ---------- ---------
    5001 hitchcock  8.530E+09
    5002 stephen    7.419E+09
    5003 ajay       5.463E+09
    5004 disney     4.588E+09
    5005 karan      1.237E+09
    5006 mallik     7.413E+09
    5007 dev        1.457E+09
7 rows selected.
SQL> select * from movies;
  MOV_ID MOV_TITLE       MOV_YEAR MOV_LANG      DIR_ID
--------- --------------- --------- ---------- ---------
    7001 titanic            2012 english        5002
    7002 fanaa              2014 hindi          5007
    7003 hum                1995 hindi          5006
    7004 sholay             1985 hindi          5005
    7005 harryporter         2002 english        5002
    7006 deewar              2016 hindi          5005
    7007 goal               2004 english        5001
7 rows selected.
SQL> insert into movie_cast values('&ACT_ID','&MOV_ID','&ROLE');
Enter value for act_id: 1001
Enter value for mov_id: 7007
Enter value for role: hero
old   1: insert into movie_cast values('&ACT_ID','&MOV_ID','&ROLE')
```

```
new   1: insert into movie_cast values('1001','7007','hero')

1 row created.

SQL> select * from movie_cast;

  ACT_ID    MOV_ID ROLE

--------- --------- ----------

    1001      7007 hero

    1003      7004 hero

    1007      7004 heroine

    1010      7006 heroine

    1008      7006 hero

    1002      7006 hero

    1006      7006 heroine

    1002      7004 hero

    1001      7001 hero

    1004      7005 heroine

    1010      7002 heroine

11 rows selected.

SQL> insert into rating values('&mov_id','&rev_stars');

Enter value for mov_id: 7001

Enter value for rev_stars: 5

old   1: insert into rating values('&mov_id','&rev_stars')

new   1: insert into rating values('7001','5')

1 row created.

SQL> select * from  rating;

  MOV_ID REV_STARS

--------- ---------
```

```
   7001       5

   7002       4

   7003       3

   7004       5

   7005       0

   7006       4

   7007       2

7 rows selected.

SQL> select m.mov_title

  2  from movies m, director d

  3  where m.dir_id=d.dir_id and dir_name='hitchcock';

MOV_TITLE

---------------

Goal

SQL> select m.mov_title

  2  from movies m where exists (select mc.mov_id, count(*)

  3  from movie_cast mc

  4  where m.mov_id=mc.mov_id and exists (select act_id, count(*)

  5  from movie_cast

  6  group by act_id

  7  having count(*) >1 )

  8  group by mov_id

  9  having count(*)>2);

MOV_TITLE

---------------

sholay
```

```
deewar

SQL> (select a.act_name from actor a, movie_cast mc,movies m
  2  where a.act_id=mc.act_id and m.mov_id=mc.mov_id and mov_year<2000)
  3  intersect
  4  (select a.act_name from actor a, movie_cast mc,movies m
  5  where a.act_id=mc.act_id and m.mov_id=mc.mov_id and mov_year>2015);

ACT_NAME
----------
Salman

SQL> select mov_title,REV_STARS as highest_rating
  2  from movies m,rating r
  3  where m.mov_id=r.mov_id and rev_stars in
  4  (select max(REV_STARS) from rating)
  5  group by  mov_title,REV_STARS;

MOV_TITLE       HIGHEST_RATING
--------------- --------------
titanic                  5
sholay                   5

SQL> select mov_title,rev_stars
  2  from movies m,rating r
  3  where m.mov_id=r.mov_id and rev_stars>1
  4  order by mov_title;

MOV_TITLE       REV_STARS
--------------- ---------
deewar                4
fanaa                 4
```

```
goal                2

hum                 3

sholay              5

titanic             5

6 rows selected.

SQL> (select mov_title,REV_STARS as highest_rating

  2  from movies m,rating r

  3  where m.mov_id=r.mov_id and rev_stars in

  4  (select max(REV_STARS) from rating)

  5  group by  mov_title,REV_STARS) union (select mov_title,rev_stars

  6  from movies m,rating r

  7  where m.mov_id=r.mov_id and rev_stars>1);

MOV_TITLE       HIGHEST_RATING

--------------- --------------

deewar                  4

fanaa                   4

goal                    2

hum                     3

sholay                  5

titanic                 5

6 rows selected.

SQL> update rating set rev_stars=5 where mov_id in( select mov_id from movies natural join
director

  2  where dir_name='stephen');

2 rows updated.

SQL> select * from rating;

   MOV_ID REV_STARS
```

--------- ---------

7001	5
7002	4
7003	3
7004	5
7005	5
7006	4
7007	2

7 rows selected.

PROGRAM -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)

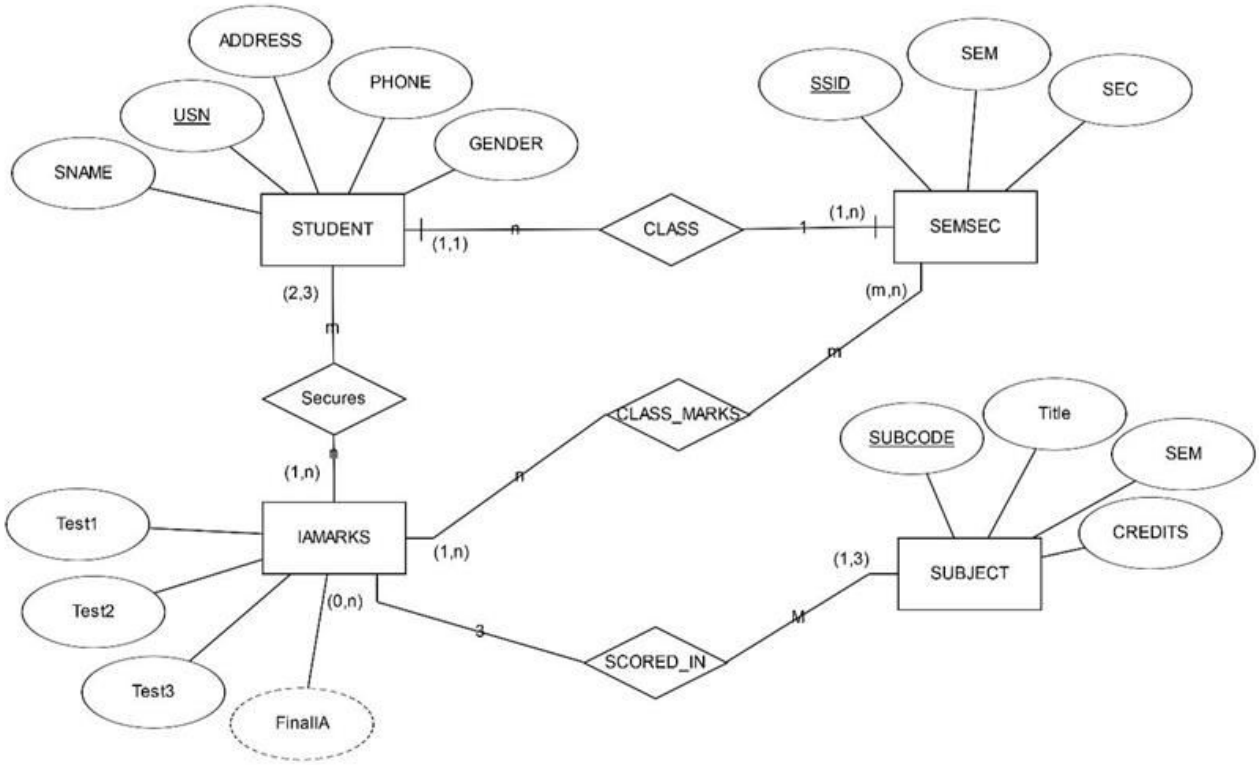
IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)

Write SQL queries to

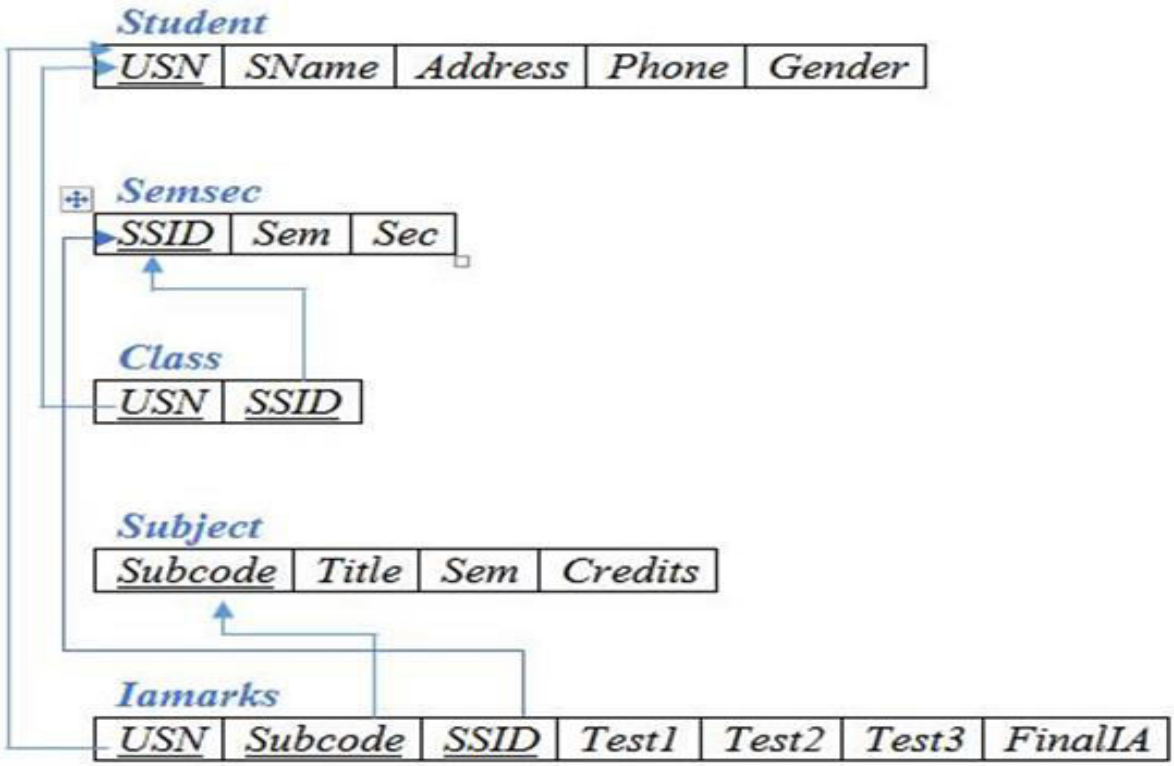
1. List all the student details studying in fourth semester 'C' section.
2. Compute the total number of male and female students in each semester and in each section.
3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.
4. Calculate the FinalIA (average of best two test marks) 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



SQL> create table student (usn int primary key, sname varchar(10) not null, address varchar(10), pho
ne number(10), gender varchar(7));

Table created.

SQL> create table semsec (ssid int primary key, sem int, sec varchar(2))
2 partition by range (sem)
3 (partition p1 values less than (4),
4 partition p2 values less than (8));

Table created.

SQL> create table classes(usn references student (usn) on delete cascade, ssid references semsec (ss
id) on delete cascade);

Table created.

SQL> create table subject(sub_code varchar(8) primary key, title varchar(10), sem int, credits int)
partition by range (sem)
2 (partition p1 values less than (4),
3 partition p2 values less than (8));

Table created.

SQL> create table iamarks (usn references student (usn) on delete cascade, subcode references
subjec
t(sub_code) on delete cascade, ssid references semsec (ssid) on delete cascade, test1 int,test2 int,
test3 int, finalia int);

Table created.

SQL> insert into student values('&usn','&sname','&address','&phone','&gender');

Enter value for usn: 1001

Enter value for sname: vikram

Enter value for address: hubli

Enter value for phone: 1236547890

```
Enter value for gender: male
old   1: insert into student values('&usn','&sname','&address','&phone','&gender')
new   1: insert into student values('1001','vikram','hubli','1236547890','male')
1 row created.
SQL> select * from  student;
     USN SNAME      ADDRESS        PHONE GENDER
--------- ---------- ---------- --------- -------
    1001 vikram     hubli      1.237E+09 male
    1002 ajay       pune       1.473E+09 male
    1003 anjali     bangalore  7.895E+09 female
    1004 baagi      belgavi    1.237E+09 female
    1005 sam        hubli      7.419E+09 male
    1006 anjan      hubli      4.561E+09 male
6 rows selected.
SQL> insert into semsec values('&ssid','&sem','&sec')
  2  ;
Enter value for ssid: 111
Enter value for sem: 3
Enter value for sec: A
old   1: insert into semsec values('&ssid','&sem','&sec')
new   1: insert into semsec values('111','3','A')
1 row created.
SQL> select * from semsec;
    SSID       SEM SE
--------- --------- --
     111         3 A
```

```
    112        5 C

    113        7 B

    114        4 A

    115        5 B

    116        5 A

    117        5 C

7 rows selected.

SQL> insert into classes values('&usn','&ssid');

Enter value for usn: 1001

Enter value for ssid: 115

old   1: insert into classes values('&usn','&ssid')

new   1: insert into classes values('1001','115')

1 row created.

SQL> select * from classes;

     USN      SSID

---------- ---------

    1001       115

    1002       115

    1003       116

    1004       117

    1005       111

    1006       115

6 rows selected.

SQL> insert into subject values ('&sub_code','&title','&sem','&credits');

Enter value for sub_code: cs0011

Enter value for title: java
```

Enter value for sem: 5

Enter value for credits: 4

old 1: insert into subject values ('&sub_code','&title','&sem','&credits')

new 1: insert into subject values ('cs0011','java','5','4')

1 row created.

SUB_CODE TITLE SEM CREDITS

-------- ---------- --------- ---------

cs0011 java 5 4

ec0031 cpp 5 4

cs0054 database 5 4

cs0071 ecs 7 4

is0054 ds 5 4

ee0055 eee 5 4

cs0065 cg 6 4

7 rows selected.

SQL> insert into iamarks values('&usn','&subcode','&ssid','&test1','&test2','&test3','&finalia');

Enter value for usn: 1001

Enter value for subcode: cs0054

Enter value for ssid: 115

Enter value for test1: 20

Enter value for test2: 20

Enter value for test3: 10

Enter value for finalia: 20

old 1: insert into iamarks values('&usn','&subcode','&ssid','&test1','&test2','&test3','&finalia')

new 1: insert into iamarks values('1001','cs0054','115','20','20','10','20')

1 row created.

```
SQL> select * from iamarks;

     USN SUBCODE      SSID     TEST1     TEST2     TEST3   FINALIA

--------- -------- --------- --------- --------- --------- ---------

    1001 cs0054        115       20        20        10        20

    1002 cs0071        114       16        16        12        16

    1006 ee0055        112       10        18        19        19

    1003 ec0031        115       20        19        19        20

    1005 cs0054        115       20        19        20        20

    1005 cs0071        116       19        20        20        20

    1004 cs0054        115       10        10        10        10

7 rows selected.

SQL>  select s.usn,s.sname

  2   from student s,semsec sc,classes c

  3   where sc.ssid=c.ssid and s.usn=c.usn and sc.sem=5 and sc.sec='C';

     USN SNAME

--------- ----------

    1004 baagi

SQL>  select sc.ssid,sem,sec,gender,count(*)

  2  from semsec sc,classes c, student s

  3  where sc.ssid=c.ssid and s.usn=c.usn

  4  group by sc.ssid,sem,sec,gender;

    SSID      SEM SE GENDER   COUNT(*)

--------- --------- -- ------- ---------

     116        5 A  female         1

     111        3 A  male           1

     117        5 C  female         1
```

```
    115        5 B  male          3

SQL> create view stud_iamarks as  select usn,sname,subcode, test1
  2   from student natural join iamarks
  3   where usn='&usn';
Enter value for usn: 1005
old   3:  where usn='&usn'
new   3:  where usn='1005'
View created.
SQL> select * from stud_iamarks;
     USN SNAME      SUBCODE      TEST1
---------- ---------- -------- ---------
    1005 sam       cs0054         20
    1005 sam       cs0071         21
SQL> alter table iamarks drop column finalia;
Table altered.
SQL> alter table iamarks add finalia int;
Table altered.
SQL> select * from iamarks;
     USN SUBCODE      SSID     TEST1     TEST2     TEST3   FINALIA
---------- -------- --------- --------- --------- --------- ---------
    1001 cs0054        115       20        20        10
    1002 cs0071        114       16        16        12
    1006 ee0055        112       10        18        19
    1003 ec0031        115       20        19        19
    1005 cs0054        115       20        19        20
    1005 cs0071        116       19        20        20
```

```
1004 cs0054        115        10        10        10

7 rows selected.

SQL>

SQL> update iamarks set finalia= round((test1+test2+test3-least(test1,test2,test3))/2);


SQL> CREATE OR REPLACE PROCEDURE AVGMARKS
 2  IS
 3  CURSOR C_IAMARKS IS
 4  SELECT GREATEST(TEST1,TEST2) AS A, GREATEST(TEST1,TEST3) AS B, GREATEST(TEST3,TEST2) AS
C
 5  FROM IAMARKS
 6  WHERE FINALIA IS NULL
 7  FOR UPDATE;
 8  C_A NUMBER;
 9  C_B NUMBER;
10  C_C NUMBER;
11  C_SM NUMBER;
12  C_AV NUMBER;
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  --DBMS_OUTPUT.PUT_LINE(C_A || ' ' || C_B || ' ' || C_C);
19  IF (C_A >= C_B AND C_B >= C_C) THEN
20  C_SM:=C_A+C_B;
21  ELSIF (C_A >= C_B AND C_C >= C_B) THEN
```

```
 22  C_SM:=C_A+C_C;
 23  ELSIF (C_B >= C_A AND C_A >= C_C) THEN
 24  C_SM:=C_B+C_A;
 25  ELSIF (C_B >= C_A AND C_C >= C_A) THEN
 26  C_SM:=C_B+C_C;
 27  ELSIF(C_C >= C_A AND C_A >= C_B) THEN
 28  C_SM:=C_C+C_A;
 29  ELSIF(C_C >= C_A AND C_B >= C_A)THEN
 30  C_SM:=C_C+C_B;
 31  END IF;
 32  C_AV:=C_SM/2;
 33  --DBMS_OUTPUT.PUT_LINE('SUM = '||C_SM);
 34  --DBMS_OUTPUT.PUT_LINE('AVERAGE = '||C_AV);
 35  UPDATE IAMARKS SET FINALIA = C_AV WHERE CURRENT OF C_IAMARKS;
 36  END LOOP;
 37  CLOSE C_IAMARKS;
 38  END;
 39  /
Procedure created.
SQL> begin
  2  avgmarks;
  3  end;
  4  /
PL/SQL procedure successfully completed.
SQL> select * from iamarks;
    USN SUBCODE      SSID     TEST1     TEST2     TEST3   FINALIA
```

```
--------- -------- --------- --------- --------- --------- ---------
    1001 cs0054       115       20        20        10        20
    1002 cs0071       114       16        16        12        16
    1006 ee0055       112       10        18        19        19
    1003 ec0031       115       20        19        19        20
    1005 cs0054       115       20        19        20        20
    1005 cs0071       116       19        20        20        20
    1004 cs0054       115       10        10        10        10
7 rows selected.
SQL> select
  2  CASE
  3  WHEN finalia <= 20 and finalia >=17 then 'Outstanding'
  4  WHEN finalia <= 16 and finalia>= 12 then 'Average'
  5  else 'Weak'
  6  end as finalia
  7  from iamarks;
FINALIA
-----------
Outstanding
Outstanding
Outstanding
Outstanding
Outstanding
Outstanding
Weak
7 rows selected.
```

PROJECT – 5

Consider the schema for Company Database:
EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)

DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)

DLOCATION(DNo,DLoc)

PROJECT(PNo, PName, PLocation, DNo)

WORKS_ON(SSN, PNo, Hours)

Write SQL queries to

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

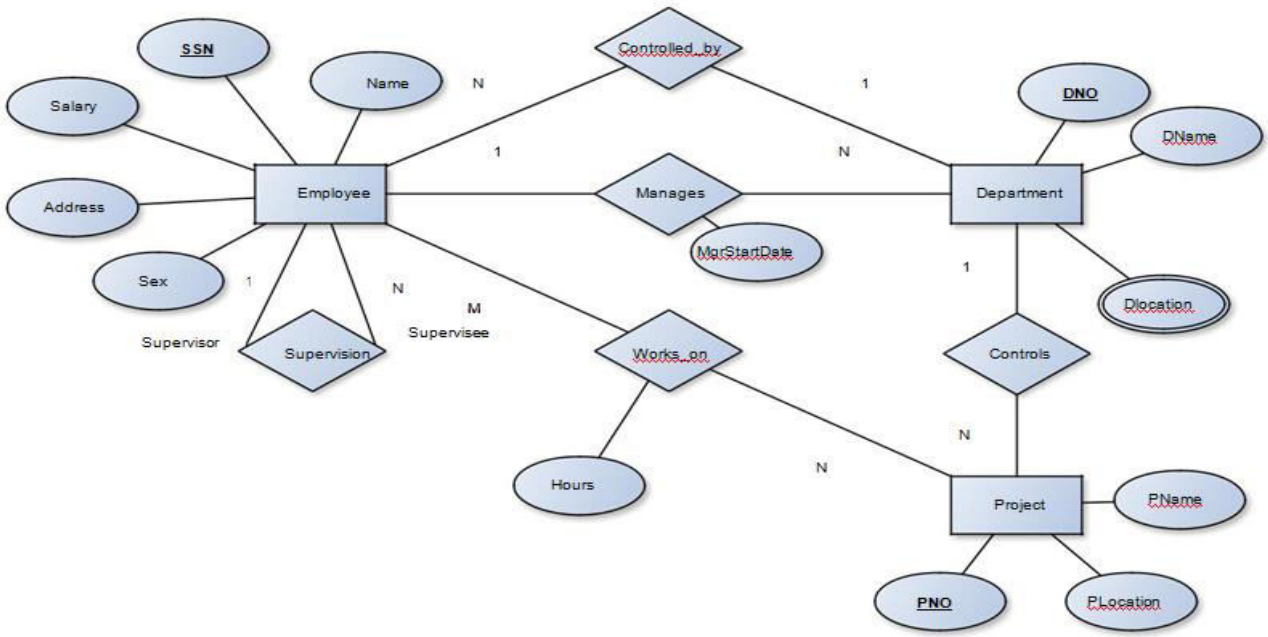
2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.

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

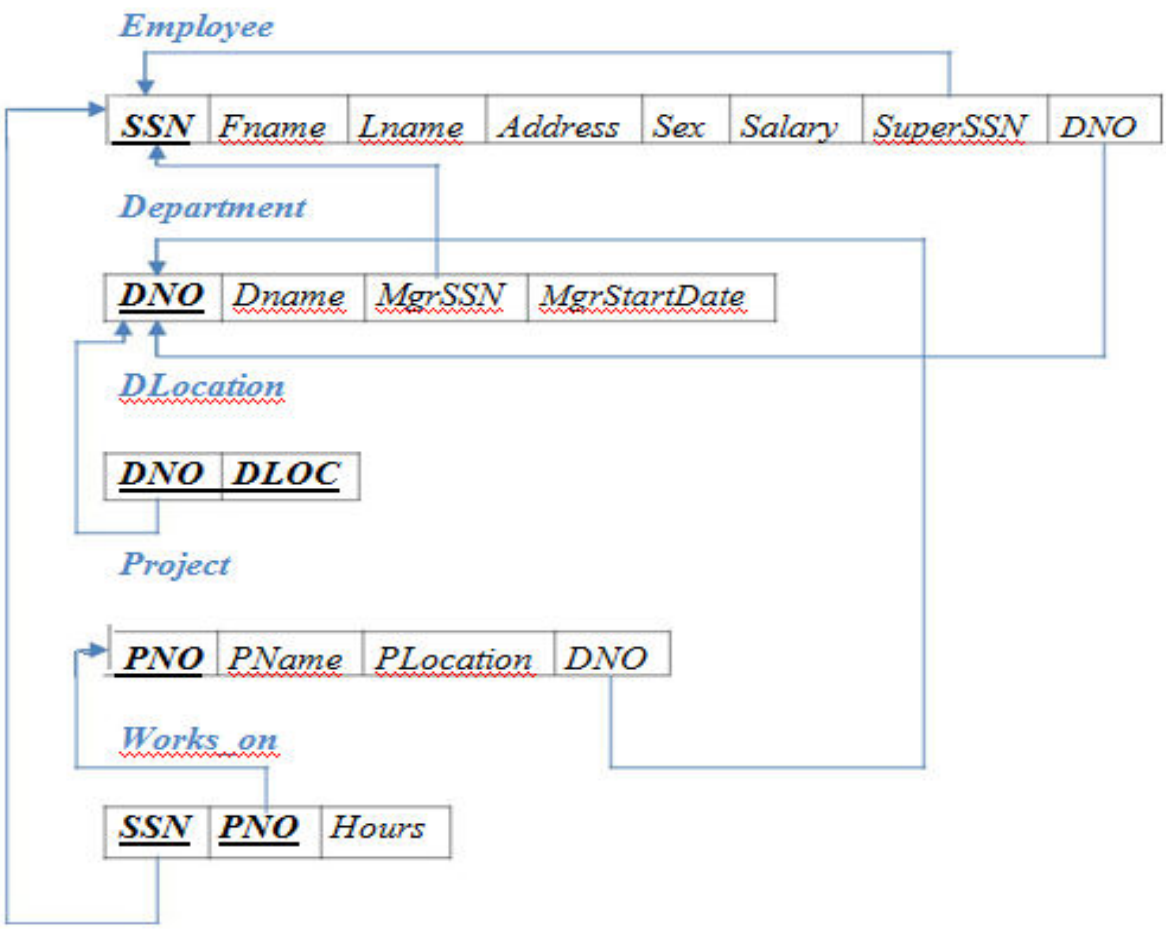
4. Retrieve the name of each employee who works on all the projects controlledby department number 5 (use NOT EXISTS operator).

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

Entity-Relationship Diagram



Schema Diagram




```
SQL>  CREATE TABLE department
 2   ( dno    NUMBER(4)
 3                 CONSTRAINT department_pk PRIMARY KEY,
 4    dname  VARCHAR2(30)
 5                 CONSTRAINT department_name_unique UNIQUE,
 6    mgrssn      NUMBER(6),
 7    mgrstartdate       date
 8   );
Table created.
SQL> CREATE TABLE employee
 2  ( ssn     NUMBER(6)
```

```
 3              CONSTRAINT employees_pk PRIMARY KEY,
 4  ename       VARCHAR2(20)
 5              CONSTRAINT emp_first_name_not_null NOT NULL,
 6  address     VARCHAR2(15),
 7  sex varchar(7),
 8  salary          NUMBER(6),
 9  superssn        CONSTRAINT emp_mgr_to_empno_fk REFERENCES employee,
10   dno   CONSTRAINT emp_to_dept_fk REFERENCES department);
```

Table created.

```
SQL>create table dlocation(dno CONSTRAINT dept_to_dloc_fk REFERENCES department,
dloc varchar(10));
```

Table created.

```
SQL> create table project (pno int CONSTRAINT project_pk primary key, pname varchar(10) not null,
pl
oc varchar(10), dno CONSTRAINT proj_to_dept_fk REFERENCES department);
```

Table created.

```
SQL> create table workson (ssn CONSTRAINT emp_workson_fk REFERENCES employee, pno
CONSTRAINT workson_proj_fk REFERENCES project, hours int);
```

Table created.

```
SQL> insert into department values('&dno','&dname','&mgrssn','&mgrstartdate');
Enter value for dno: 1
Enter value for dname: research
Enter value for mgrssn: 1234
Enter value for mgrstartdate: 10-aug-2014
old   1: insert into department values('&dno','&dname','&mgrssn','&mgrstartdate')
new   1: insert into department values('1','research','1234','10-aug-2014')
1 row created.
```

```
SQL> select * from department;

    DNO DNAME                          MGRSSN MGRSTARTD
--------- ------------------------------ --------- ---------
        1 research                         1234 10-AUG-14
        2 headquarters                     2345 20-MAY-15
        3 rd                               3456 20-AUG-14
        4 maintenance                      4567 22-JAN-13
        5 accounting                       5678 11-FEB-11
        6 deploy                           6789 12-DEC-11
6 rows selected.
SQL> insert into employee values('&ssn','&ename','&address','&sex','&salary','&superssn','&dno');
Enter value for ssn: 1234
Enter value for ename: john
Enter value for address: hubli
Enter value for sex: male
Enter value for salary: 15000
Enter value for superssn: 1234
Enter value for dno: 2
old   1: insert into employee values('&ssn','&ename','&address','&sex','&salary','&superssn','&dno')
new   1: insert into employee values('1234','john','hubli','male','15000','1234','2')
1 row created.
SQL> select * from employee;

    SSN ENAME           ADDRESS        SEX       SALARY  SUPERSSN      DNO
--------- -------------------- --------------- ------- --------- --------- ---------
    1234 john           hubli          male      15000     1234        2
    2345 scott          delhi          male      12000     1234        1
```

```
3456 henry          hubli        male      11000     2345      3
4567 jay            mumbai       male      14000     1234      4
5678 dom            sydney       male      12500     2345      4
6789 daya           bangalore    female    12500     2345      3
1452 jaya           belgaum      female    11400     1234      5
3654 gani           hubli        male      13500     1452      1
7485 kaykay         darwad       male      12500     1452      3
9 rows selected.
SQL> insert into dlocation values('&dno','&dloc');
Enter value for dno: 1
Enter value for dloc: hubli
old   1: insert into dlocation values('&dno','&dloc')
new   1: insert into dlocation values('1','hubli')
1 row created.
SQL> select * from dlocation;
    DNO DLOC
--------- ----------
      1 hubli
      1 darwad
      2 delhi
      3 mumbai
      4 kolkatta
      5 chennai
      5 bangalore
      3 delhi
8 rows selected.
```

```
SQL> insert into project values ('&pno','&pname','&ploc','&dno');
Enter value for pno: 1
Enter value for pname: chocolate
Enter value for ploc: hubli
Enter value for dno: 1
old   1: insert into project values ('&pno','&pname','&ploc','&dno')
new   1: insert into project values ('1','chocolate','hubli','1')
1 row created.
SQL> select * from project;
    PNO PNAME      PLOC           DNO
--------- ---------- ---------- ---------
       1 chocolate  hubli            1
       2 biscuits   darwad           2
       3 rolls      delhi            1
       4 pizza      mumbai           4
       5 burger     chennai          3
       6 kfc        kolkatta         5
       7 mfc        banglore         3
       8 inox       banglore         4
8 rows selected.
SQL> insert into workson values('&ssn','&pno','&hours');
Enter value for ssn: 1234
Enter value for pno: 1
Enter value for hours: 42
old   1: insert into workson values('&ssn','&pno','&hours')
new   1: insert into workson values('1234','1','42')
```

```
1 row created.

SQL> select * from workson;

     SSN       PNO     HOURS

--------- --------- ---------

    1234         1        42

    2345         2        40

    3456         3        42

    4567         4        40

    6789         2        40

    1234         1        30

    1234         2        20

    7485         5        40

    4567         4        40

    5678         5        40

    6789         3        40

    1452         3        40

12 rows selected.

SQL> select pq.pno, pq.pname from project pq where pno in(

  2  ( select p.pno

  3  from project p, employee e, workson w

  4  where w.pno=p.pno and e.ssn=w.ssn and e.ename='scott' )

  5  union

  6  ( select p.pno

  7  from project p, employee e,department d

  8  where e.ssn=d.mgrssn and p.dno=d.dno and ename='scott' )

  9  );
```

```
    PNO PNAME
--------- ----------
      2 biscuits
SQL> SELECT SALARY*1.1
  2  FROM EMPLOYEE e, PROJECT p, WORKSON w
  3  WHERE PNAME='rolls' and e.ssn=w.ssn and p.pno=w.pno;
SALARY*1.1
----------
    12540
    13750
    12100
SQL>  select max(salary),min(salary),avg(salary), sum(salary)
  2   from employee e, department d
  3   where e.dno=d.dno and DNAME='accounting';
MAX(SALARY) MIN(SALARY) AVG(SALARY) SUM(SALARY)
----------- ----------- ----------- -----------
    11400       11400       11400       11400
SQL> select e.ename
  2  from employee e
  3  where exists (select *
  4  from workson w,project p
  5  where p.pno=w.pno and e.ssn=w.ssn and p.dno=2);
ENAME
--------------------
john
scott
```

```
daya
SQL> select e.ename
  2  from employee e
  3  where not exists (select *
  4  from workson w,project p
  5  where p.pno=w.pno and e.ssn=w.ssn and p.dno=2);
ENAME
--------------------
jaya
henry
gani
jay
dom
kaykay
6 rows selected.
SQL> select d.dno, count(*)
  2  from employee e, department d
  3  where e.salary > 12000 and e.dno=d.dno
  4  group by d.dno
  5  having count(*)>=2
  6  ;
     DNO  COUNT(*)
--------- ---------
       4         2
       3         2
```