VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Belgaum, Karnataka-590 014

Laboratory Manual

Database Management System Laboratory with Mini Project

(17CSL58)

Compiled by

1. Prof. Latharani T R.
2. Prof. Ancy Thomas
3. Prof. Geetha N
4. Prof. Lakshmi S R

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

(ACCREDITED BY NBA)

ACHARYA INSTITUTE OF TECHNOLOGY

Soldevanahalli, Bengaluru-560107

2019-2020

Table of contents

	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) 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.	
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 '1BI17CS101' 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.	

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 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.	

- **VISION OF THE INSTITUTE**

Acharya Institute of Technology, committed to the cause of value-based education in all disciplines, envisions itself as fountainhead of innovative human enterprise, with inspirational initiatives for Academic Excellence

- **MISSION OF THE INSTITUTE**

Acharya Institute of Technology, strives to provide excellent academic ambiance to the students for achieving global standards of technical education, foster intellectual and personal development, meaningful research and ethical service to sustainable societal needs.

- **VISION OF THE DEPARTMENT**

Envisions to be recognized for quality education and research in the field of Computing, leading to creation of globally competent engineers, who are innovative and adaptable to the changing demands of industry and society.

- **MISSION OF THE DEPARTMENT**

  - ✓ Act as a nurturing ground for young computing aspirants to attain the excellence by imparting quality education.
  - ✓ Collaborate with industries and provide exposure to latest tools/ technologies.
  - ✓ Create an environment conducive for research and continuous learning

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

Students shall

- Have a successful career in academia, R&D organizations, IT industry or pursue higher studies in specialized field of Computer Science & Engineering and allied disciplines.
- Be competent, creative and a valued professional in the chosen field
- Engage in life-long learning, professional development and adapt to the working environment quickly
- Become effective collaborators and exhibit high level of professionalism by leading or participating in addressing technical, business, environmental and societal challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO	**statement**
	Students shall
PSO-1:	Apply the knowledge of hardware, system software, algorithms, networking and data bases.
PSO-2:	Design, analyze and develop efficient and secure algorithms using appropriate data structures, databases for processing of data.
PSO-3	Be capable of developing stand alone, embedded and web-based solutions having easy to operate interface using software engineering practices and contemporary computer programming languages.

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

EXPERIMENT 1:

Consider the following schema for a Library Database:

BOOK (Book_id, Title, Publisher_Name, Pub_Year)

BOOK_AUTHORS (Book_id, Author_Name)

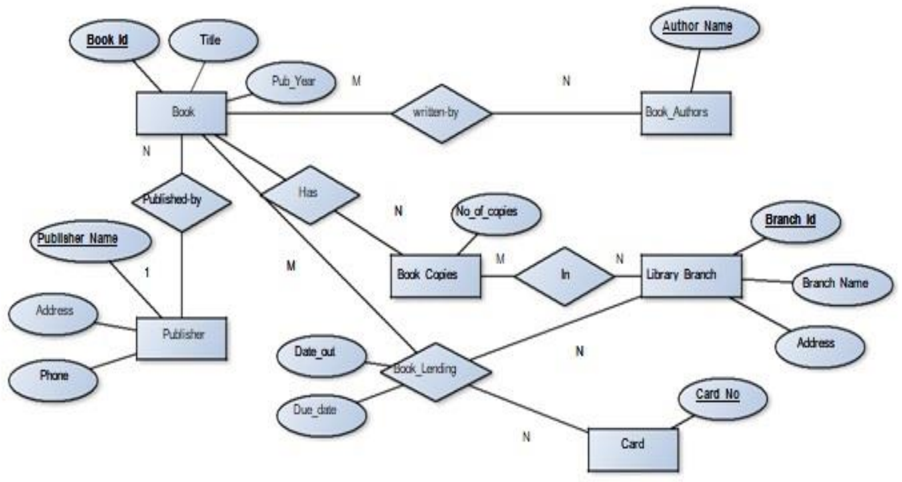
PUBLISHER (Name, Address, Phone)

BOOK_COPIES (Book_id, Branch_id, No-of_Copies)

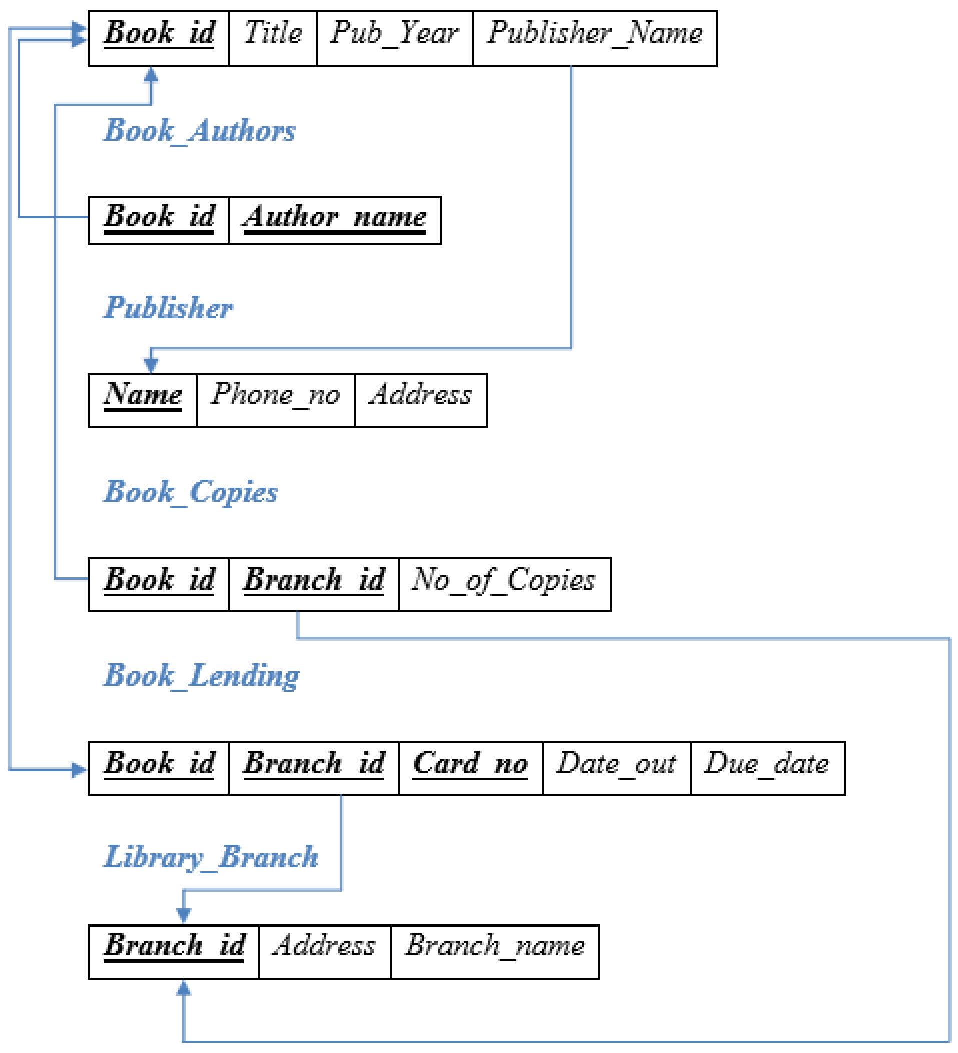
BOOK_LENDING (Book_id, Branch_id, Card_No, Date_Out, Due_Date)

LIBRARY_BRANCH (Branch_id, Branch_Name, Address)

Entity-Relationship Diagram



Schema Diagram



---

create table publisher
(
Name varchar(12),
Address varchar(12),
Phone bigint,
primary key(name)
);

```
desc publisher;

 Name                                   Null?              Type
 --------------------------------------- -------- ----------------------------
 NAME                                    NOT NULL          VARCHAR2(12)
 ADDRESS                                                   VARCHAR2(12)
 PHONE                                                     BIGINT


insert into publisher values('mcgrawhill','Bangalore',9480506312);
insert into publisher values('pearson','Newdelhi',9785642365);
insert into publisher values('random house','Hydrabad',8796452368);
insert into publisher values('sapna','Chenai',8947589632);
insert into publisher values('oxford','Bangalore',9785642315);

select *from publisher;

NAME                ADDRESS          PHONE
------------        ------------         ----------
mcgrawhill          Bangalore     9480506312
pearson             Newdelhi      9785642365
random house        Hydrabad      8796452368
sapna               Chenai        8947589632
oxford              Bangalore     9785642315
```

create table book
(
book_id int,
title varchar(15),
publisher_name varchar(10),
pub_year int,
primary key(book_id),
foreign key (publisher_name) references publisher(name) on delete cascade
);

```
desc book;
 Name                          Null?              Type
 ----------------             --------           -----------------
 BOOK_ID                          NOT NULL       INT
 TITLE                                            VARCHAR(15)
 PUBLISHER_NAME                                  VARCHAR(10)
 PUB_YEAR                                        INT
```

```
insert into book values(1,'DBMS','mcgrawhill',2017);
insert into book values(2,'ADBMS','mcgrawhill',2016);
insert into book values(3,'CN','pearson',2016);
insert into book values(4,'CG','oxford',2015);
insert into book values(5,'OS','pearson',2016);

select *from book;

BOOK_ TITLE          PUBLISHER_   PUB_YEAR
----- --------------- ---------- ----------
1    DBMS              mcgrawhill            2017
2    ADBMS            mcgrawhill            2016
3    CN                pearson               2016
4    CG                oxford                2015
5    OS                 pearson              2016
```

create table book_authors
(book_id int,
author_name varchar(15),
primary key(book_id,author_name),
foreign key(book_id) references book(book_id) on delete cascade
);

```
SQL>desc book_authors;
 Name                          Null?               Type
 ---------------------         --------            -----------------
 BOOK_ID                        NOT NULL                     INT
 AUTHOR_NAME                    NOT NULL           VARCHAR2(15)
```

```
insert into book_authors values(1,'navathe');
insert into book_authors values(2,'navathe');
insert into book_authors values(3,'tenenboum');
insert into book_authors values(4,'edward');
insert into book_authors values(5,'galvin');
```

```
select *from book_authors;

BOOK_ AUTHOR_NAME
----- ---------------
1     navathe
2     navathe
3     tenenboum
4     edward
5     galvin
```

create table library_branch
(
branch_id int,
branch_name varchar(10),
address varchar(15),
primary key(branch_id)
);

```
SQL>desc library_branch;
 Name                     Null?              Type
 -----------------        --------           -------------

 BRANCH_ID                NOT NULL           INT
 BRANCH_NAME                                 VARCHAR(10)
 ADDRESS                                     VARCHAR(15)
```

```
insert into library_branch values(10,'RR nagar','Bangalore');
insert into library_branch values(11,'Manipal','Bangalore');
insert into library_branch values(12,'RNSIT','Bangalore');
insert into library_branch values(13,'Rajajnagar','Bangalore');
insert into library_branch values(14,'Nitte','Mangalore');
```

```
select *from library_branch;

BRANC BRANCH_NAM ADDRESS
----- ---------- ---------------
10    RR nagar   Bangalore
11    Manipal    Bangalore
12    RNSIT      Bangalore
13    Rajajnagar Bangalore
14    Nitte      Mangalore
```

```
create table book_copies
(book_id int,
branch_id int,
no_of_copies int,
primary key(book_id, branch_id),
foreign key(book_id) references book(book_id) on delete cascade,
foreign key(branch_id) references library_branch(branch_id) on delete
cascade);
```

```
desc book_copies;
 Name                          Null?                         Type
 ----------------              --------------                --------------
 BOOK_ID                           NOT NULL                  INT
 BRANCH_ID                        NOT NULL                    INT
 NO_OF_COPIES                                                INT
```

```
insert into book_copies values(1,10,10);
insert into book_copies values(1,11,5);
insert into book_copies values(2,12,2);
insert into book_copies values(2,13,5);
insert into book_copies values(3,14,7);
insert into book_copies values(5,10,1);
insert into book_copies values(4,11,3);
```

```
select *from book_copies;

BOOK_ BRANC NO_OF_COPIES
----- ----- ------------
1     10          10
1     11           5
2     12           2
2     13           5
3     14           7
5     10           1
4     11           3
```

```
create table book_lending
(book_id int,
branch_id int,
 card_no int,
date_out date,
due_date date,
primary key(book_id,branch_id,card_no),
foreign key(book_id) references book(book_id),
foreign key(branch_id) references library_branch(branch_id) on delete
cascade
);
```

```
desc book_lending;
 Name                          Null?          Type
 ---------------               --------       -------------
 BOOK_ID                        NOT NULL            INT
 BRANCH_ID                      NOT NULL            INT
 CARD_NO                        NOT NULL            INT
 DATE_OUT                                           DATE
 DUE_DATE                                           DATE
```

```
insert into book_lending values(1,10,101,'2017-01-01','2017-06-01');
insert into book_lending values(3,14,101,'2017-01-11','2017-03-11');
insert into book_lending values(2,13,101,'2017-02-17','2017-04-21');
insert into book_lending values(4,11,101,'2017-03-15','2017-07-15');
insert into book_lending values(1,11,104,'2017-04-12','2017-05-12');
```

```
select *from book_lending;

BOOK_ BRANC CARD_ DATE_OUT  DUE_DATE
----- ----- ----- --------- ---------
1     10    101   2017-01-01 2017-06-01

3     14    101   2017-01-11 2017-03-11

2     13    101   2017-02-17 2017-04-21

4     11    101   2017-03-15 2017-07-15

1     11    104   2017-04-12 2017-05-12
```

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,c.branch_id
```

```
from book b,book_authors a,book_copies c
where b.book_id=a.book_id and
b.book_id=c.book_id;
```

```
BOOK_ID    TITLE        PUBLISHER_NAME  AUTHOR_NAME    NO_OF_COPIES
BRANCH_ID
----- --------------- ---------- --------------- ------------ -----
1    DBMS         mcgrawhill        navathe             10          10
1    DBMS         mcgrawhill        navathe              5          11
2    ADBMS          mcgrawhill      navathe              2          12
2    ADBMS          mcgrawhill      navathe              5          13
3    CN                 pearson     tenenboum            7          14
5    OS                  pearson    galvin               1          10
4    CG                 oxford      edward               3          11
```

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

```
select card_no
from book_lending
where date_out between '2017-01-01' and '2017-06-30'
group by card_no
having count(*)>3;
```

```
CARD_NO
-----
101
```

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=3;
```

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

```
create table bookpartition
(
book_id int,
pub_year int,
publisher_name varchar(10))
partition by range(pub_year)
(partition p0 values less than(2000),
partition p1 values less than(2005),
partition p2 values less than(2010));
```

```
insert into bookpartition values('1',1998, 'vikas');
insert into bookpartition values('3',1999, 'subash');
insert into bookpartition values('2',2002, 'pearson');
insert into bookpartition values('5',2005, 'vikas');
insert into bookpartition values('7',2008, 'subash');
insert into bookpartition values('8',2007, 'pearson');
```

```
select * from bookpartition;
select * from bookpartition partition(p0);
select * from bookpartition partition(p1);
select * from bookpartition partition(p2);
```

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

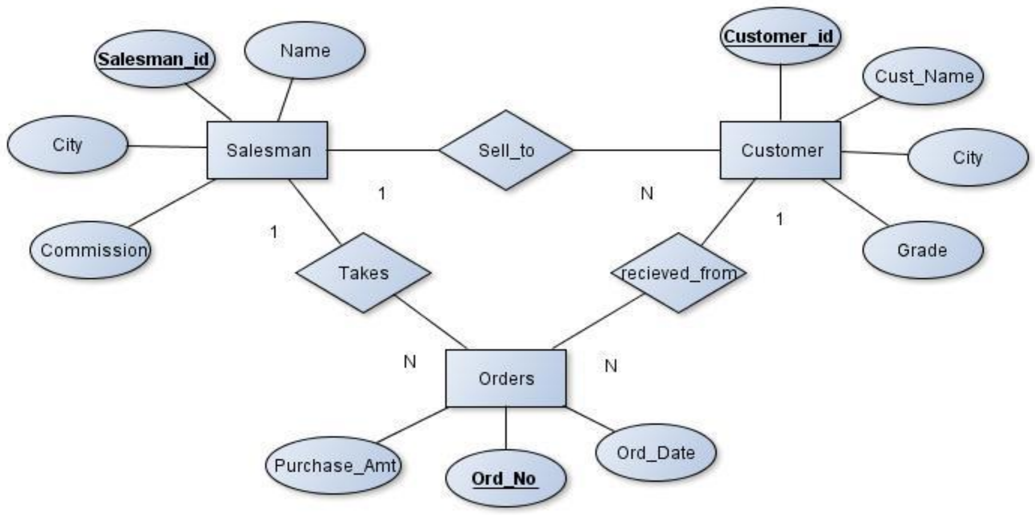
```
create view book_available as
select b.book_id,b.title,c.no_of_copies
from book_copies c,book b,library_branch l
where b.book_id=c.book_id and c.branch_id=l.branch_id;
```

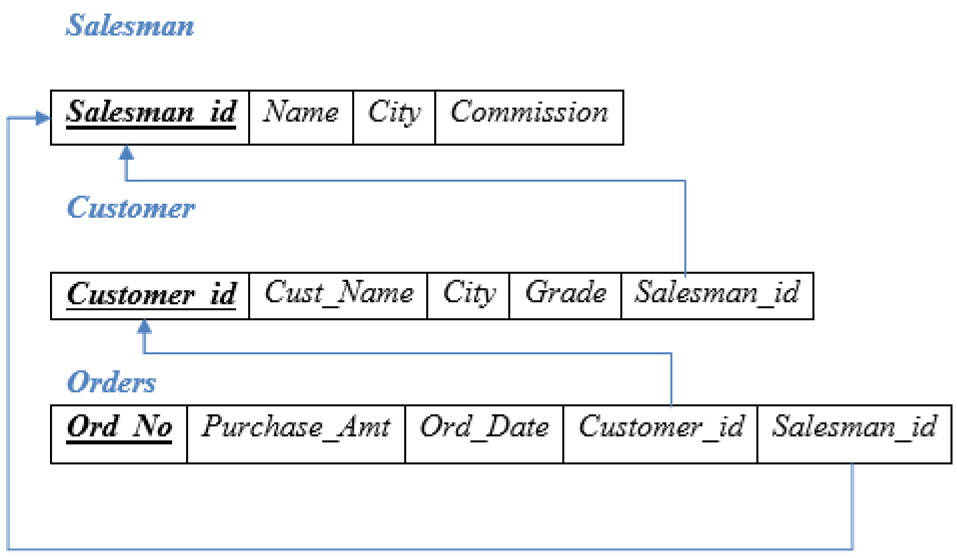
EXPERIMENT 2

Consider the following schema for Order Database:

SALESMAN(Sid, SName, City, Commission)
CUSTOMER(Cid, CName, City, Grade, Sid)
ORDERS(Orderno, Purchase_Amt, Ord_Date, Cid, Sid)

<u>Entity- Relationship Diagram</u>





Salesman

<u>Salesman_id</u>	*Name*	*City*	*Commission*

Customer

<u>Customer_id</u>	*Cust_Name*	*City*	*Grade*	*Salesman_id*

Orders

<u>Ord_No</u>	*Purchase_Amt*	*Ord_Date*	*Customer_id*	*Salesman_id*

create table salesman

```
(Sid int,
Sname varchar(15),
City varchar(15),
Commission int,
primary key(sid)
);
```

```
desc salesman;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                       NOT NULL INT
 SNAME                                              VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 COMMISSION                                         NUMBER(38)
insert into salesman values(1000,'arun','mangalore',15);
insert into salesman values(2000,'harsha','bangalore',25);
insert into salesman values(3000,'sagar','mysore',20);
insert into salesman values(4000,'priya','mangalore',30);
insert into salesman values(5000,'divya','delhi',15);
```

```
SQL> select *from salesman;

SID   SNAME           CITY            COMMISSION
----- --------------- --------------- ----------
1000  arunmangalore           15
2000  harshabangalore           25
3000  sagarmysore           20
4000  priyamangalore           30
5000  divyadelhi           15
```

```
create table customer
(cid int,
cname varchar(15),
city varchar(15),
grade int,
sid varchar(5),
primary key(cid),
foreign key(sid) references salesman(sid) on delete cascade
);
```

```
desc customer;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CID                                       NOT NULL INT
 CNAME                                              VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 GRADE                                              NUMBER(38)
 SID                                                VARCHAR2(5)
```

```
insert into customer values(10,'ravi','bangalore',100,1000);
insert into customer values(12,'shwetha','mangalore',300,1000);
insert into customer values(14,'shalini','chenai',500,2000);
insert into customer values(16,'sushmitha','bangalore',700,2000);
insert into customer values(18,'pramya','bangalore',500,3000);
```

```
select *from customer;

CID   CNAME           CITY            GRADE SID
----- --------------- --------------- ---------- -----
10    ravibangalore            100 1000
12    shwethamangalore            300 1000
14    shalinichenai              500 2000
16    sushmithabangalore            700 2000
18    pramyabangalore            500 3000
```

create table orders
(Orderno int,
purchaseamt int,
orddate date,
cid int,
sid int,
primary key(orderno),
foreign key(cid) references customer(cid) on delete cascade,
foreign key(sid) references salesman(sid) on delete cascade
);

```
desc orders;
 Name                                      Null?                    Type
 ----------------------------------------- -------- ----------------------------
```

```
ORDERNO              NOT NULL    INT
PURCHASEAMT                      INT
ORDDATE                          DATE
CID                              INT
SID                              INT
```

```
insert into orders values(60,5000, '2017-01-05',10,1000);
insert into orders values(62,3000,'2017-01-17',10,2000);
insert into orders values(64,4000,'2017-02-17',12,2000);
insert into orders values(66,450,'2017-03-17',14,3000);
insert into orders values(68,2000,'2017-02-17',16,1000);
```

```
select *from orders;
```

```
ORDER PURCHASE_AMT ORD_DATE  CID   SID
----- ------------ --------- ----- -----
60           5000 2017-01-05  10    1000
62           3000  2017-01-17 10    2000
64           4000 2017-02-17 12    2000
66            450 2017-03-17  14    3000
68           2000 2017-02-17 16    1000
```

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

```
select  grade,count(*) as count
from customer where grade >
(select avg(grade)
from customer
where city ='bangalore');
```

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

```
select s.sid,s.sname,count(cid)
from salesman s, customer c
where s.sid = c.sid
group by (s.sname) having count(cid)>1;
```

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

```
select sname, 'exists' as samecity
```

```
from salesman s
where city in
(select city
from customer
where s.sid = sid)
union
select sname, 'not exists' as samecity
from salesman s where
city not in
(select city
from customer
where s.sid = sid);
```

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

```
create view highestorder as
select s.sid,s.sname,o.purchaseamt,o.orddate
from salesman s,orders o
where s.sid = o.sid and
o.purchaseamt =
(select max(purchaseamt)
From orders c
Where c.orddate =o. orddate);

select * from highest_order;
```

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

```
delete from salesman where sid=1000;
```

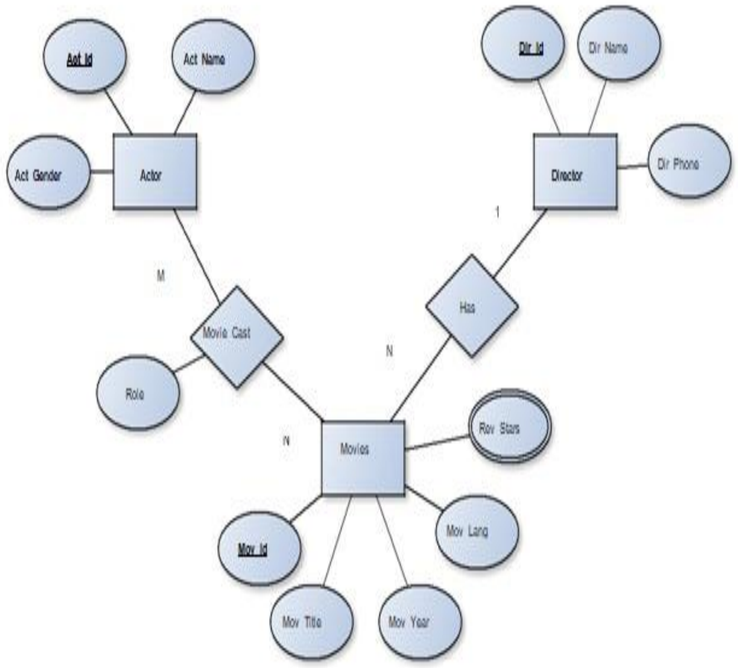
EXPERIMENT 3

Consider the schema for Movie Database:

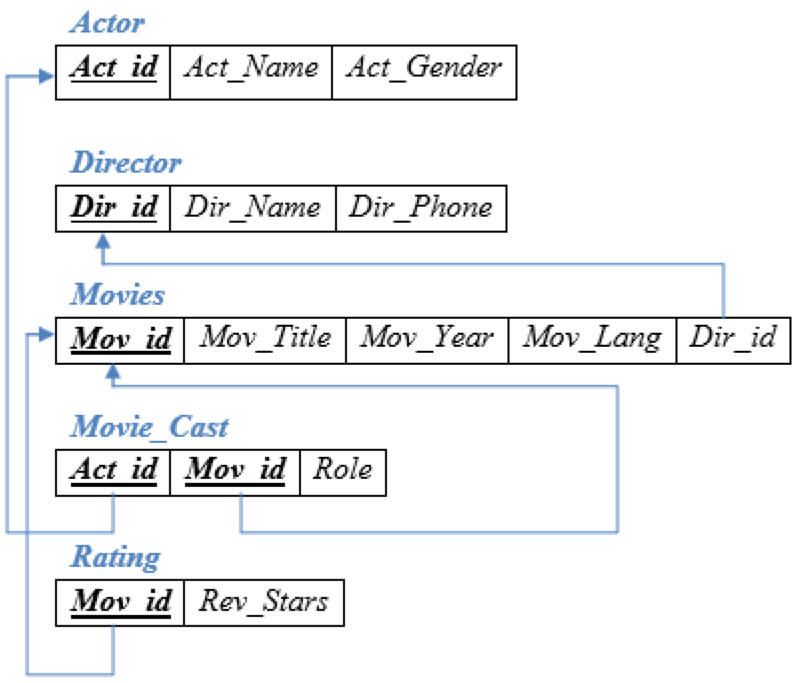
ACTOR(Act_id, Act_Name, Act_Gender)
DIRECTOR(Dir_id, Dir_Name, Dir_Phone)

MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)
MOVIE_CAST(Act_id, Mov_id, Role)
RATING(Mov_id, Rev_Stars)

Entity-Relationship Diagram



Schema Diagram



```
create table actor
(
act_id int,
act_name varchar(10),
act_gender char(1),
primary key(act_id)
);
```

```
desc actor;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                     NOT NULL INT
 ACT_NAME                                            VARCHAR(10)
 ACT_GENDER                                          CHAR(1)
```

```
insert into actor values(301,'Anushka','F');
insert into actor values(302,'Prabhas','M');
insert into actor values(303,'Punith','M');
insert into actor values(304,'Jermy','M');
insert into actor values(305,'yash','M');

select *from actor;
```

```
   ACT_ID ACT_NAME   ACT_G
---------- ---------- -----
       301 Anushka    F
       302 Prabhas    M
       303 Punith     M
       304 Jermy      M
       305 yash       M
```

create table director
(
dir_id int,
dir_name varchar(20),
dir_phone bigint,
primary key(dir_id)
);

```
desc director;
 Name                                      Null?                              Type
 ----------------------------------------- -------- ----------------------------

 DIR_ID                                     NOT NULL                          INT
 DIR_NAME                                                                     VARCHAR(20)
 DIR_PHONE                                                                    BIGINT
```

```
insert into director values(70,'Hitchcock',9487563255);
insert into director values(71,'Rajamouli',8948562346);
insert into director values(72,'steven spielberg',9823654125);
insert into director values(73,'Faran',9786542356);
insert into director values(74,'Martin',8974563214);
```

```
select *from director;

   DIR_ID DIR_NAME             DIR_PHONE
---------- -------------------- ----------
        70 Hitchcock           9487563255
        71 Rajamouli           8948562346
        72 stevenspielberg     9823654125
        73 Faran               9786542356
        74 Martin              8974563214
```

create table movies

```
(
mov_id int,
mov_title varchar(10),
mov_year int,
mov_lang varchar(10),
dir_id int,
primary key(mov_id),
foreign key(dir_id) references director(dir_id)
);
```

```
desc movies;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------

 MOV_ID                                    NOT NULL NUMBER(38)
 MOV_TITLE                                          VARCHAR2(10)
 MOV_YEAR                                           NUMBER(38)
 MOV_LANG                                           VARCHAR2(10)
 DIR_ID                                             NUMBER(38)
```

```
insert into movies values(1001,'Bahubali-2',2017,'telugu',70);
insert into movies values(1002,'Singham',2008,'hindi',70);
insert into movies values(1003,'Golmaal-2',2011,'telugu',70);
insert into movies values(1004,'Allthebest',2015,'hindi',70);
insert into movies values(1005,'Akash',2009,'kannada',70);
```

```
select *from movies;

    MOV_ID MOV_TITLE    MOV_YEAR MOV_LANG       DIR_ID
---------- ---------- ---------- ---------- ----------
      1001 Bahubali-2       2017 telugu             70
      1002 Singham          2008 hindi              71
      1003 Golmaal-2        2011 telugu             72
      1005 Akash            2009 kannada            74
      1004 Allthebest       2015 hindi              73
```

```
create table moviecast
(
act_id int,
mov_id int,
role varchar(10),
primary key(act_id,mov_id),
foreign key(act_id) references actor(act_id),
foreign key(mov_id) references movies(mov_id)
);
```

```
descmoviecast;
 Name                                      Null?                                  Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                    NOT NULL                               INT
 MOV_ID                                     NOT NULL                              INT
 ROLE                                                                             VARCHAR(10)
```

```
insert into moviecast values(301,1001,'Heroine');
insert into moviecast values(302,1002,'Hero');
insert into moviecast values(303,1003,'Guest');
insert into moviecast values(304,1004,'Hero');
insert into moviecast values(305,1005,'Heroine');
```

```
select *from moviecast;

    ACT_ID     MOV_ID ROLE
---------- ---------- ----------
       301       1001 Heroine
       302       1002 Hero
       303       1003 Guest
       304       1004 Hero
       305       1005 Heroine
```

```
create table rating
(
mov_id int,
rev_stars int,
primary key(mov_id),
foreign key(mov_id) references movies(mov_id)
);
```

```
desc rating;
 Name                                      Null?                                  Type
 ---------------------------------------- -------- ----------------------------
 MOV_ID                                   NOT NULL INT
 REV_STARS                                         INT
```

```
insert into rating values(1001,4);
insert into rating values(1002,2);
insert into rating values(1003,5);
insert into rating values(1004,3);
insert into rating values(1005,4);
```

```
select *from rating;
```

```
    MOV_ID  REV_STARS
---------- ----------
      1001          4
      1002          2
      1003          5
      1004          3
      1005          4
```

1. List the titles of all movies directed by ‘Hitchcock’.

```
select mov_title
from movies m, director d
where m.dir_id = d.dir_id and d.dir_name ='Hitchcock';
```

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

```
select distinct mov_title from movies m, moviecast mc
where m.mov_id = mc.mov_id and
(select count(mov_id)
from moviecast
where act_id=mc.act_id )>=2;
```

(Or)

```
SELECT MOV_TITLE
FROM MOVIES M, MOVIECAST MV
WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN (SELECT ACT_ID
```

FROM MOVIECAST GROUP BY
ACT_ID HAVING COUNT (ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT (*)>1;

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 MOVIECAST C
ON A.ACT_ID=C.ACT_ID
JOIN MOVIES M
ON C.MOV_ID=M.MOV_ID
WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

OR

```
SELECT A.ACT_NAME, A.ACT_NAME, C.MOV_TITLE, C.MOV_YEAR
FROM ACTOR A, MOVIECAST B, MOVIES C WHERE
A.ACT_ID=B.ACT_ID
AND B.MOV_ID=C.MOV_ID
AND C.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

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 m, rating r
where m.mov_id = r.mov_id  group by (m.mov_id)
having max(rev_stars)>=1 order by m.mov_title;
```

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

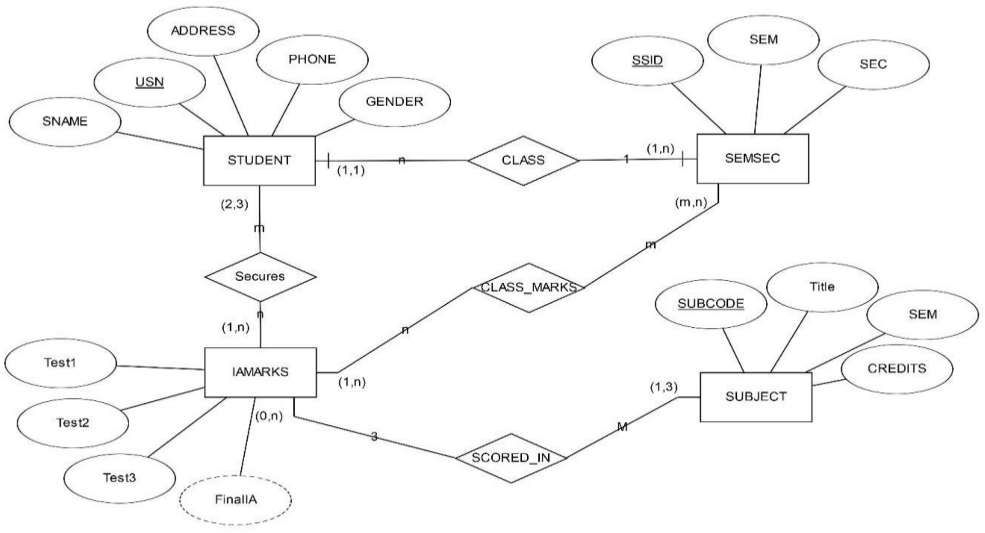
```
update rating set rev_stars=5
where mov_id in
(select m.mov_id
from movies m,director d
where m.dir_id = d.dir_id and d.dir_name='stevenspielberg');
```

EXPERIMENT 4

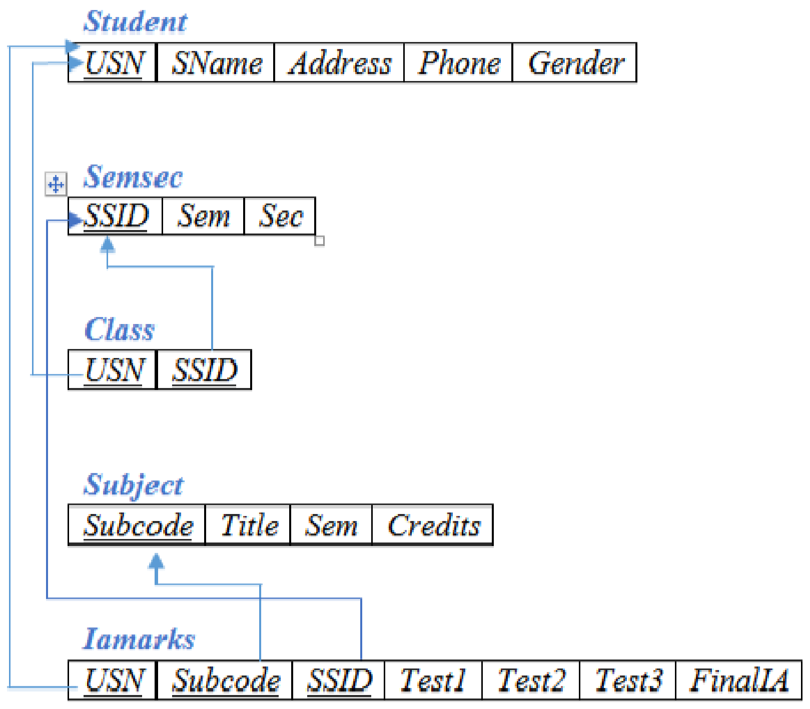
Consider the schema for College Database:

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

Entity - Relationship Diagram



Schema Diagram



```
create table student
(Usn varchar(10),
Sname varchar(10),
Address varchar(10),
Phone bigint,
Gender char(1),
primary key(usn)
);
```

```
desc student;
 Name                                    Null?    Type
 ----------------------------------------- -------- ----------------------------
 USN                                     NOT NULL VARCHAR(10)
 SNAME                                            VARCHAR(10)
 ADDRESS                                          VARCHAR(10)
 PHONE                                            BIGINT
 GENDER                                           CHAR(1)
```

```
insert into student values('CS101','Arun','ujire','9481235681','Male');
insert into student values('CS102','Pramya','Mangalore','8945689532','Female');
insert into student values('CS103','Ravi','Bangalore','9568742361','Male');
insert into student values('CS104','Vani','Puttur','8945623145','Female');
insert into student values('CS105','Akshatha','Bantwal','9845632147','Female');
insert into student values('CS106','Ranjan','Karwar','9485632158','Male');

select *from student;

USN       SNAME          ADDRESS            PHONE GENDER
---------- --------------- --------------- ---------- ------
CS101     Arunujire          9481235681 Male
CS102     Pramya         Mangalore       8945689532 Female
CS103     Ravi           Bangalore       9568742361 Male
CS104     Vani           Puttur          8945623145 Female
CS105     AkshathaBantwal        9845632147 Female
CS106     Ranjan         Karwar          9485632158 Male
```

create table semsec
(
Ssid varchar(5),
Sem int,
Sec varchar(5),
primary key(ssid)
);

```
descsemsec;
 Name                                   Null?                          Type
 ----------------------------------------- -------- ----------------------------
 SSID                                   NOT NULL                       VARCHAR(5)
 SEM                                                                   INT
 SEC                                                                   CHAR(1)

insert into semsec values('CS4A','4','A');
insert into semsec values('CS4B','4','B');
insert into semsec values('CS4C','4','C');
insert into semsec values('CS8A','8','A');
insert into semsec values('CS8B','8','B');
insert into semsec values('CS8C','8','C');

select *from semsec;

SSID       SEM SEC
----- ---------- -----
CS4A          4 A
CS4B          4 B
CS4C          4 C
```

```
CS8A          8 A
CS8B          8 B
CS8C          8 C
```

create table class
(Usn varchar(10),
Ssid varchar(5),
primary key(usn),
foreign key(usn) references student(usn) on delete cascade,
foreign key(ssid) references semsec(ssid) on delete cascade
);

```
desc class;
 Name                                      Null?                              Type
 ----------------------------------------- -------- ----------------------------
 USN                                       NOT NULL                           VARCHAR(10)
 SSID                                                                         VARCHAR(5)
```

```
insert into class values('CS101','CS8A');
insert into class values('CS102','CS8A');
insert into class values('CS103','CS8B');
insert into class values('CS104','CS8C');
insert into class values('CS105','CS4C');
insert into class values('CS106','CS4C');
```

```
select *from class;

USN        SSID
---------- -----
CS101      CS8A
CS102      CS8A
CS103      CS8B
CS104      CS8C
CS105      CS4C
CS106      CS4C
```

```
create table subject
(Subcode varchar(8),
Title varchar(15),
Sem int,
Credits int,
primary key(subcode)
);
```

```
desc subject;
 Name                                      Null?                                    Type
 ----------------------------------------- -------- ----------------------------
 SUBCODE                                   NOT NULL                                 VARCHAR(8)
 TITLE                                                                              VARCHAR(15)
 SEM                                                                                 INT
 CREDITS                                                                            INT
```

```
insert into subject values('15CS41','ACA','4','5 ');
insert into subject values('15CS42 ','GTE','4','4');
insert into subject values('15CS43','C++','4','3');
insert into subject values('10CS81','JAVA','8','4');
insert into subject values('10CS82','WEB','8','4');
insert into subject values('10CS83','IOT','8','5');
```

```
select * from subject;

SUBCODE  TITLE                SEM    CREDITS
-------- --------------- ---------- ----------
15CS41   ACA                    4          5
15CS42   GTE                    4          4
15CS43   C++                    4          3
10CS81   JAVA                   8          4
10CS82   WEB                    8          4
10CS83   IOT                    8          5
```

```
create table IAmarks
(
Usn varchar(10),
Subcode varchar(8),
Ssid varchar(5),
```

test1 int,
test2 int,
test3 int,
finalIA int,
primary key(usn,subcode,ssid),
foreign key(usn) references student(usn) on delete cascade,
foreign key(subcode) references subject(subcode),
foreign key(ssid) references semsec(ssid)
);

```
descIAmarks;
 Name                                    Null?     Type
 --------------------------------------- -------- ----------------------------

 USN                                    NOT NULL            VARCHAR(10)
 SUBCODE                                NOT NULL            VARCHAR(8)
 SSID                                   NOT NULL            VARCHAR(5)
 TEST1                                                      INT
 TEST2                                                      INT
 TEST3                                                      INT
 FINALIA                                                    INT
```

```
insert into IAmarks values('CS101','10CS81','CS8A','19','20','18','');

insert into IAmarks values('CS102','10CS81','CS8A','16','15','12','');

insert into IAmarks values('CS103','10CS82','CS8B','09','08','12','');

insert into IAmarks values('CS104','10CS83','CS8C','03','05','08','');

insert into IAmarks values('CS105','15CS41','CS4C','10','14','16','');

insert into IAmarks values('CS106','15CS42','CS4C','13','15','20','');
```

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”;
```

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

section.

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

3.Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects.

```
CREATE VIEW
STU_TEST1_MARKS_VIEW AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1AY13CS091';
```

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

```
Create view averagefinder as (select usn,subcode,greatest(test1,test2,test3) as
highest, case
When test1<=greatest(test1,test2,test3) and test1>least(test1,test2,test3) then
test1
When test2<=greatest(test1,test2,test3) and test2>least(test1,test2,test3) then
test2
Else test3
End as secondhighest from iamarks);

select * from averagefinder;

create view average as (select subcode,(highest+secondhighest)/2 as finalia
from averagefinder);

select * from average;

update iamarks
join average
on iamarks.subcode=average.subcode
set iamarks.finalia=average.finalia;
```

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.

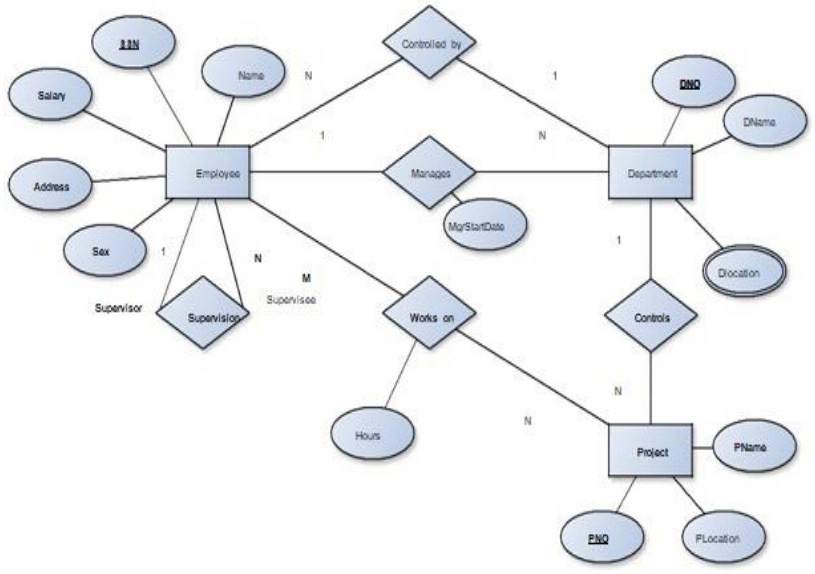
```
SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER,
(CASE
WHEN IA.FINALIA BETWEEN 17 AND 20 THEN
'OUTSTANDING' WHEN IA.FINALIA BETWEEN 12 AND 16
THEN 'AVERAGE' ELSE 'WEAK'
END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB
```

EXPERIMENT 5

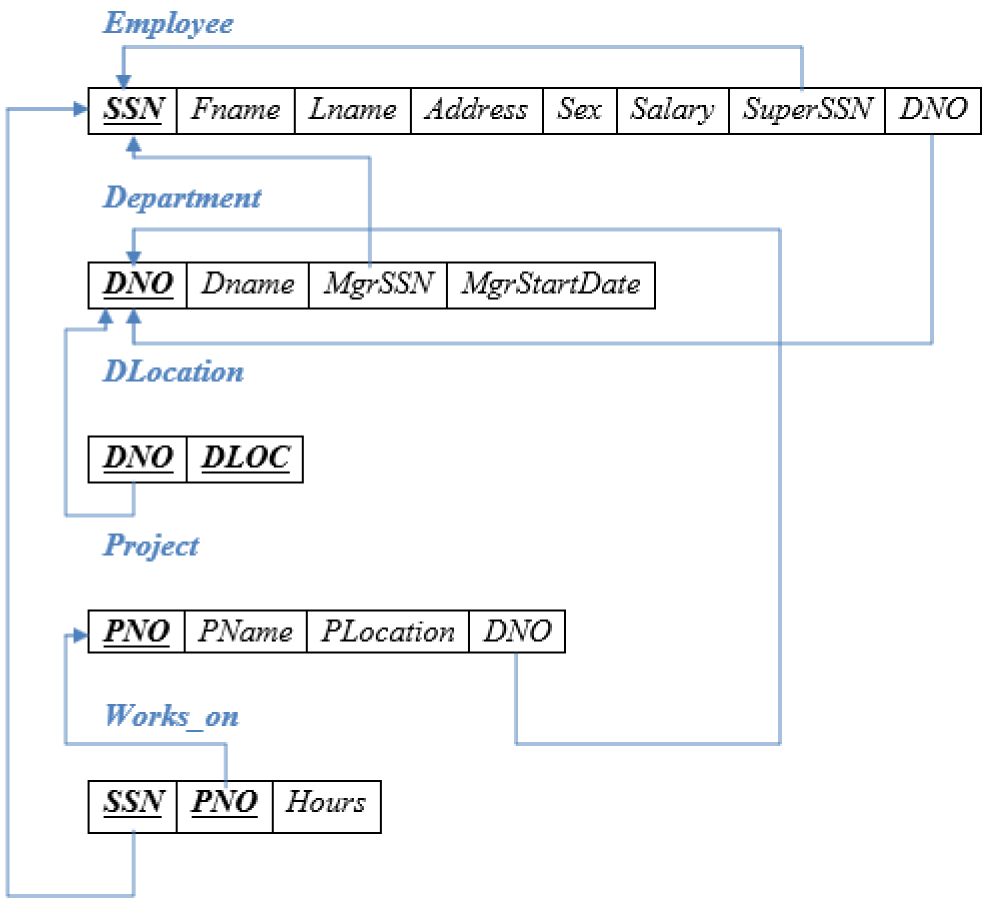
Consider the schema for Company Database:

EMPLOYEE (*SSN*, *Name, Address, Sex, Salary, SuperSSN, DNo*)
DEPARTMENT (*DNo*, *DName, MgrSSN, MgrStartDate*)
DLOCATION (*DNo,DLoc*)
PROJECT (*PNo*, *PName, PLocation, DNo*)
WORKS_ON (*SSN, PNo*, *Hours*)

Entity- Relationship Diagram



Schema Diagram



CREATE TABLE DEPARTMENT

(DNO INT PRIMARY KEY,

DNAME VARCHAR(10),

MGRSTARTDATE DATE);

CREATE TABLE EMPLOYEE

(SSN VARCHAR (20) PRIMARY KEY,

FNAME VARCHAR(10),
LNAME VARCHAR(10),

ADDRESS VARCHAR(10),

SEX CHAR (1),

```
SALARY INTEGER,

SUPERSSN VARCHAR(20) REFERENCES EMPLOYEE (SSN),

DNO INT REFERENCES DEPARTMENT(DNO));
```

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

```
ALTER TABLE DEPARTMENT
ADD MGRSSN varchar(10) REFERENCES EMPLOYEE (SSN);

CREATE TABLE DLOCATION
(DLOC VARCHAR(10),
DNO INT REFERENCES DEPARTMENT (DNO),
PRIMARY KEY (DNO, DLOC));

CREATE TABLE PROJECT
(PNO INTEGER PRIMARY KEY,
PNAME VARCHAR (10),
PLOCATION VARCHAR(10),
DNO INT REFERENCES DEPARTMENT (DNO));

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

insert into Employee values( ‘ 11’,’scott ’,’Bangalore ’,’M ’,’600000 ’,’14 ’,’1’);
insert intoEmployee values( ‘12 ’,’john ’,’Mangalore ’,’M ’,’500000’,’14 ’,’5 ’);
insert into Employee values( ‘13 ’,’James ’,’Hassan ’,’M ’,’400000 ’,’14 ’,’5 ’);
insert into Employee values( ‘ 14’,’ kavitha’,’Puttur ’,’F ’,’700000 ’,’17 ’,’5 ’,);
insert into Employee values( ‘ 15’,’ Kavya’,’Ujire ’,’ F’,’800000 ’,’ 17’,’5 ’,);
insert into Employee values( ‘ 16’,’ Veena’,’Pune ’,’ F’,’900000 ’,’ 17’,’5 ’,);
insert into Employee values( ‘ 17’,’Nagesh ’,’Mysore ’,’M ’,’ 700000’,’17 ’,’5 ’,);


insert into Department values( ‘1’,’Datamining’,’11’,’16-may-17’);
insert into Department values( ‘2’,’Administration’,’12’,’15-may-17’);
insert into Department values( ‘3’,’Networking’,’13’,’05-may-17’);
insert into Department values( ‘4’,’Testing’,’14’,’12-jun-18’);
insert into Department values( ‘5’,’accounts’,’15’,’15-jun-18’);
```

insert into DLocation values(‘1’,’Venoor’);
insert into DLocation values(‘2’,’Karkala’);

insert into DLocation values(‘3’,’Puttur’);
insert into DLocation values(‘4’,’Kerala’);
insert into DLocation values(‘5’,’Pune’);

insert into Project values(‘100’,’IOT’,’Mumbai’,’1’);
insert into Project values(‘200’,’Bigdata’,’Pune’,’1’);
insert into Project values(‘300’,’Database’,’Shirsi’,’5’);
insert into Project values(‘400’,’Cloudcomputing’,’UK’,’5’);
insert into Project values(‘500’,’Android’,’DK’,’5’);

insert into Works_on values(‘11’,’100’,’30’);
insert into Works_on values(‘11’,’300’,’10’);
insert into Works_on values(‘12’,’300’,’50’);
insert into Works_on values(‘12’,’400’,’50’);
insert into Works_on values(‘12’,’500’,’50’);
insert into Works_on values(‘13’,’200’,’50’);

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 DISTINCT P.PNO

FROM PROJECT P, DEPARTMENT D, EMPLOYEE E

WHERE E.DNO=D.DNO

AND D.MGRSSN=E.SSN

AND E.LNAME=”SCOTT”)

UNION

(SELECT DISTINCT P1.PNO

FROM PROJECT P1, WORKS_ON W, EMPLOYEE E1

WHERE P1.PNO=W.PNO

AND E1.SSN=W.SSN

AND E1.LNAME=”SCOTT”);

2. Show the resulting salaries if every employee working on the ‘IoT’ project is given a 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”;

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), MAX(E.SALARY), MIN(E.SALARY),

AVG(E.SALARY)

FROM EMPLOYEE E, DEPARTMENT D

WHERE E.DNO=D.DNO

AND D.DNAME=”ACCOUNTS”;

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

SELECT E.FNAME, E.LNAME

FROM EMPLOYEE E

WHERE NOT EXISTS (SELECT PNUMBER FROM PROJECT WHERE DNO = “5”

AND PNUMBER NOT IN (SELECT W.PNUMBER FROM WORKS_ON W WHERE

E.SSN = W.SSN));

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
AND E.SALARY>600000
AND D.DNO IN (SELECT E1.DNO
FROM EMPLOYEE E1
GROUP BY E1.DNO
HAVING COUNT(*)>5)
GROUP BY D.DNO;
```

VIVA VOCE QUESTIONS

1. What is database?
2. What is DBMS?
3. What is a Database system?
4. What are the disadvantages in File Processing System?

Answer: Data redundancy & inconsistency.
Difficult in accessing data.
Data isolation.
Data integrity.
Concurrent access is not possible.
Security Problems.

5. What is extension and intension?

Answer: Extension -It is the number of tuples present in a table at any instance. This is time dependent.

Intension – It is a constant value that gives the name, structure of table and the constraints laid on it.

6. Explain the difference between a database administrator and a data administrator.
7. Describe the three levels of data abstraction?
8. Define the "integrity rules"

Answer: There are two Integrity rules.

Entity Integrity: States that, Primary key cannot have NULL value

Referential Integrity: States that, Foreign Key can be either a NULL value or should be Primary Key value of other relation.

9. What is Data Independence?
10. What is a view? How it is related to data independence?
11. What is an Entity?
12. What is an Entity type?
13. What is an Entity set?
14. What is Weak Entity set?
15. What is an attribute?
16. What are a Relation Schema and a Relation?
17. What is degree of a Relation?
18. What is Relationship?
19. What is Relationship set?
20. What is Relationship type?

21. What is degree of Relationship type?
22. What is Data Storage – Definition Language?
23. What is DML (Data Manipulation Language)?
24. What is VDL (View Definition Language)?
25. What is DML Compiler?
26. What is DDL Interpreter?
27. What is Data Model?

Answer: A collection of conceptual tools for describing data, data relationships data semantics and constraints.

28. What is E-R model?

Answer: This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

29. What is Object Oriented model?

Answer: This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

30. What is primary key?
31. What is foreign key?
32. What is unique key?
33. What is candidate key?
34. What is composite primary key?
35. What is super key?
36. What is secondary key (alternate key)?
37. What is database trigger?
38. What is a view? How it is related to data independence?

Answer: A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base table. In other words, there is no stored file that direct represents the view instead a definition of view is stored in data dictionary.
Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence.

39. What is aggregation?

Answer: Selecting the data in group of records is called aggregation. There are five aggregate system functions they are viz. Sum, Min, Max, Avg, Count.

40. What is decomposition?

Answer: Selecting all data without any grouping and aggregate functions is called Decomposition. The data is selected, as it is present in the table.

41. What is indexing and what are the different kinds of indexing?

Answer: Indexing is a technique for determining how quickly specific data can be found. **Types:**

1. Binary search style indexing
2. B-Tree indexing
3. Inverted list indexing
4. Memory resident table
5. Table indexing

42. What is normalization?

Answer: It is a process of analysing the given relation schemas based on their Functional Dependencies (FDs) and primary key to achieve the properties (1).Minimizing redundancy, (2). Minimizing insertion, deletion and update anomalies.

43. What is Functional Dependency?
44. When is a functional dependency F said to be minimal?
45. What is multivalued dependency?
46. What is Lossless join property?

Answer: It guarantees that the spurious tuples generation does not occur with respect to relation schemas after decomposition.

47. What is 1 NF (Normal Form)?

Answer: The domain of attribute must include only atomic (simple, indivisible) values.

48. What is Fully Functional dependency?

Answer: It is based on concept of full functional dependency. A functional dependency X Y is full functional dependency if removal of any attribute A from X means that the dependency does not hold any more.

49. What is 2 NF?

Answer: A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

50. What is 3 NF?

Answer: A relation schema R is in 3NF if it is in 2NF and for every FD X A either of the following is true.

1. X is a Super-key of R.
2. A is a prime attribute of R.

In other words, if every non prime attribute is non-transitively dependent on primary key.

51. What is BCNF (Boyce-Codd Normal Form)?

Answer: A relation schema R is in BCNF if it is in 3NF and satisfies an additional constraint that for every FD X A, X must be a candidate key.

52. What is 4NF?

Answer: A relation schema R is said to be in 4NF if for every Multivalued dependency X Y that holds over R, one of following is true.
1.) X is subset or equal to (or) XY = R.
2.) X is a super key.

53. What is 5NF?

Answer: A Relation schema R is said to be 5NF if for every join dependency {R1, R2, ..., Rn} that holds R, one the following is true

1.)Ri = R for some i.
2.) The join dependency is implied by the set of FD, over R in which the left side is key of R.

54. What is DKNF (Domain Key Normal Form)?

Answer: A relation is said to be in DKNF if all constraints and dependencies that should hold on the the constraint can be enforced by simply enforcing the domain constraint and key constraint on the relation.

55. What are the Inference Rules for Functional and Multivalued Dependency?
56. What is a query?

Answer: A query with respect to DBMS relates to user commands that are used to interact with a data base. The query language can be classified into data definition language and data manipulation language.

57. What is the use of CASCADE CONSTRAINTS?

Answer: When this clause is used with the DROP command, a parent table can be dropped even when a child table exists.

58. Difference between VARCHAR and VARCHAR2?

Answer: Varchar means fixed length character data (size) i.e., min size-1 and max-2000

Varchar2 means variable length character data i.e., min-1 to max-4000

59. Which system table contains information on constraints on all the tables created?

Answer: USER_CONSTRAINTS

60. What is the main difference between the IN and EXISTS clause in sub queries?

Answer: The main difference between the IN and EXISTS predicate in sub query is the way in which the query gets executed.

IN -- The inner query is executed first and the list of values obtained as its result is used by the outer query. The inner query is executed for only once.

EXISTS -- The first row from the outer query is selected, then the inner query is executed and, the outer query output uses this result for checking. This process of inner query execution repeats as many no .of times as there are outer query rows. That is, if there are ten rows that can result from outer query, the inner query is executed that many no. of times.

61. What is transaction in DBMS?

Answer: A transaction is a logical unit of database processing that includes one or more database access operations – these can include insertion, deletion, modification or retrieval operations.

62. What are the different types of failures?

- A complete failure (system crash)
- A transaction or system error
- Local errors or exception conditions detected by the transaction
- Concurrency control enforcement
- Disk failure

- Physical problems and catastrophes

63. What are ACID properties

- Atomicity
- Consistency preservation
- Isolation
- Durability or permanency

64. What is schedule?

Answer: When transactions are executing concurrently in an interleaved fashion, then the order of execution of operations from the various transaction is known as **schedule or history.**

65. What is timestamp?

Answer: Timestamp is a unique identifier created by the DBMS to identify a transaction.

66. What is the use of Spool command?

Answer: Spool will record all your statements in a text file which will be created in the path specified by you in path directory. Until and unless you switch off the spool it won't record your statements but a file will be created in that path.

spool C:\temp.txt (temp file will be created in C drive)
select * from tablename1 (Output will be generated)
select * from tablename2 (Output will be generated)
etc......
spool off (All records after your file was created will be recorded with errors/messages/outputs/results etc.)