**PROGRAM 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**

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

**b. Find the movie names where one or more actors acted in two or more movies. c. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).**

**d. 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.**

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

**Table Creation;**

create table actor(

aid integer(4) primary key,

aname varchar(10),

agender varchar(1));

create table director(

did integer(5) primary key,

dname varchar(10),

phno varchar(10));

create table movies(

mid integer(6) primary key,

title varchar(10),

moyear varchar(4),

lang varchar(10),

did integer(5) references director(did) on delete cascade);

create table moviecast(

aid integer(4) references actor(aid) on delete cascade,

mid integer(6) references movies(mid) on delete cascade,

role varchar(10),

primary key(aid,mid));

create table rating(

mid integer (6) references movies(mid) on delete cascade,

rev integer (2) check (rev<6),

primary key(mid,rev));

**Insertion of Values to Tables;**

insert into actor values(1001,'preetham','M');

insert into actor values(1002,'jason','M');

insert into actor values(1003,'tom','M');

insert into actor values(1004,'jenna','f');

insert into actor values(1005,'anna','f');

insert into director values(10001,'hitchcock','9889898989');

insert into director values(10002,'steven','988989845');

insert into director values(10003,'varun','9889898798');

insert into director values(10004,'nolan','9889898111');

insert into director values(10005,'richbitch','9889898000');

insert into movies values(100001,'Psycho','1960','English',10001);  insert into movies values(100002,'Don','2011','Hindi',10003);

insert into movies values(100003,'super','2019','kannada',10003);

insert into movies values(100004,'vertigo','1958','English',10001);  insert into movies values(100005,'dunkirk','2016','English',10004);  insert into movies values(100006,'jaws','1975','English',10002);

insert into moviecast values(1001,100003,'Pscho');

insert into moviecast values(1004,100005,'Lead role');

insert into moviecast values(1003,100002,'villian');

insert into moviecast values(1002,100001,'Lead role');

insert into moviecast values(1005,100004,'side actor');

insert into moviecast values(1001,100006,'lead actor');

insert into rating values(100001,3);

insert into rating values(100002,4);

insert into rating values(100003,5);

insert into rating values(100004,1);

insert into rating values(100005,3);

insert into rating values(100006,2.5);

**Queries:**

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

   select title from movies natural join director where dname=’hitchcock’;

MID TITLE MOYE LANG DID DID DNAME PHNO

100001 Psycho 1960 English 10001 10001 hitchcock 9889898989 100004 vertigo 1958 English 10001 10001 hitchcock 9889898989

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

select distinct title from movies natural join moviecast where aid in (select aid from moviecast group by aid having count(\*)>=2);

TITLE

super jaws

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

Select distinct aname from (actor join moviecast using(aid)) join movies using(mid) where moyear not between 2000 and 2015;

ANAME

preetham

**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 title movie\_name, count(rev) stars, max(rev) highest\_rating from movies natural join rating group by movies.title order by movies.title;

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

update rating set rev=5 where mid in(select mid from movies natural join director where dname='steven');

Sql> select \* from rating;

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

**SALESMAN (*Salesman\_id, Name, City, Commission*)**

**CUSTOMER (*Customer\_id, Cust\_Name, City, Grade, Salesman\_id*)**

**ORDERS (*Ord\_No, Purchase\_Amt, Ord\_Date, Customer\_id, Salesman\_id*)  Write SQL queries to**

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

**2. Find the name and numbers of all salesmen who had more than one customer. 3. List all salesmen and indicate those who have and don’t have customers in their cities (Use UNION operation.)**

**4. Create a view that finds the salesman who has the customer with the highest order of a day. 5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.**

**Table Creation;**

create table salesman(

sid integer(4) primary key,

sname varchar(10) not null,

city varchar(10) not null,

commission integer(10) not null);

create table customer(

cid integer(5) primary key,

cname varchar(10) not null,

city varchar(10) not null,

grade integer(3) not null check (grade<=10),

sid integer(4) references salesman(sid) on delete cascade);

create table orders(

ordno integer (3) primary key,

pamt integer (10) not null,

orddate date not null,

sid integer(4) references salesman(sid) on delete cascade,

cid integer(5) references customer(cid) on delete cascade);

**Insertion of Values to Tables;**

insert into salesman values(1000,'mike','bangalore',5000);

insert into salesman values(1001,'micheal','mangalore',500);

insert into salesman values(1002,'bevan','bangalore',1000);

insert into salesman values(1003,'clarke','pune',10000.50);

insert into salesman values(1004,'virat','mumbai',6500);

insert into customer values(25251,'varun','bellary',8.5,1003);

insert into customer values(25255,'pavan','shimoga',4.5,1000);

insert into customer values(25265,'sripad','bellary',7,1003);

insert into customer values(25278,'manish','jaipur',6,1004);

insert into customer values(25281,'kariyamma','chennai',2.4,1002);

insert into orders values(101,80000,'05-09-17',1003,25265);

insert into orders values(102,45000,'05-10-17',1000,25255);

insert into orders values(103,57000,'25-09-17',1003,25251);

insert into orders values(104,10000,'29-11-17',1004,25278);

insert into orders values(105,100000,'03-01-17',1002,25281);

select sid from customer group by(sid) having

count(sid)>=2;

update salesman set city='bangalore' where sname='clarke';

update customer set city='bangalore' where city='bellary';

update customer set grade=10 where cname='pavan';

update orders set pamt=1000000 where ordno=104;

**Queries:**

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

select count(cid) from customer where grade>=(select avg(grade) from customer  where city='bangalore');

Count(cid)

2

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

select sid, sname from salesman where sid in (select sid from customer group by sid having  count(sid)>=2);

Sname sid

Clarke 1003

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

select s.sid,s.sname,s.city from salesman s,customer c where s.city=c.city union select s.sid,  s.sname,'no match' from salesman s where not s.city=any(select city from customer);

Sid Sname city

1000 mike bangalore

1002 bevan bangalore

1003 clarke bangalore

1001 micheal no match

1004 virat no match

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

 create view abc as select salesman.sid,sname from salesman,orders where orders.pamt=(select max(pamt) from orders) and orders.sid=salesman.sid;

select\*from abc;

Ord\_date Sid Sname

05-sep-17 1000 mike

05-oct-17 1001 micheal

25-sep-17 1001 micheal

29-nov-1 1002 bevan

03-jan-17 1001 micheal

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

A; delete from salesman where

sid=1000; 1 row affected;

Sql> select \* from salesman;

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

**BOOK (*Book\_id, Title, Publisher\_Name, Pub\_Year*)**

**BOOK\_AUTHORS (Book\_id, Author\_*Name*)**

**PUBLISHER (*Name, Address, Phone*)**

**BOOK\_COPIES (*Book\_id, programme\_id, No-of\_Copies*)**

**BOOK\_LENDING (*Book\_id, programme\_id, Card\_No, Date\_Out, Due\_Date*)  LIBRARY\_PROGRAMME (*programme\_id, programme\_name, Address*)**

**Write SQL queries to**

**PROGRAM 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.**

**Create database tables;**

create table publisher(

pname varchar(10) primary

key, addr varchar(10),

phno integer(10) );

create table book(

bookid integer(4) primary

key, title varchar(10),

pyear integer(4),

pname varchar(10) references publisher(pname) on delete cascade);

create table book\_author(

bookid integer(4) references book(bookid) on delete cascade,

aname varchar(10),

primary key(bookid,aname) );

create table library(branchid integer(5) primary key,

bname varchar(10),

addr varchar(10) );

create table book\_copies(

bookid integer(4) references book(bookid) on delete cascade,  branchid integer(5) references library(branchid) on delete cascade,  noc integer(3),

primary key(bookid,branchid));

create table book\_lending(

bookid integer(4) references book(bookid) on delete cascade,  branchid integer(5) references library(branchid) on delete cascade,  cardno integer(6) not null,

duedate date,

date\_out date,

primary key(bookid,branchid,cardno));

**Insertion of Values to Tables:**

insert into publisher values( 'bloomsbury','london',207631);  insert into publisher values( 'alen-unwin','uk',842501);

insert into publisher values( 'marvel','NY',212576);

insert into publisher values( 'varun','Uganda',565662);

insert into book values(0007,'HP-TOOP',2007,'bloomsbury');  insert into book values(1004,'HOBBIT',1937,'alen-unwin');  insert into book values(0958,'FF:BOD',2007,'marvel');

insert into book values(3243,'13RW',2010,'varun');

insert into book\_author values(0007,'JK.ROWLING');

insert into book\_author values(1004,'JR.TOLKIEN');

insert into book\_author values(0958,'M.FARMER');

insert into book\_author values(3243,'RS.VARUN');

insert into library values(77777,'AVALAHALLI','BANGALORE');  insert into library values(24564,'GURGAUM','NEWDELHI'); insert into library values(53656,'KORMANGALA','BANGALORE');  insert into library values(53565,'AFJDAJ','BANGALORE');

insert into book\_copies values(0007,77777,5);

insert into book\_copies values(0007,24564,5);

insert into book\_copies values(0958,24564,10);

insert into book\_copies values(3243,53656,7);

insert into book\_copies values(1004,53565,1);

insert into book\_lending values(1004,53565,1,'10-10-17','20-09-17'); insert into book\_lending values(0007,77777,43,'20-10-17','21-09-17'); insert into book\_lending values(0007,24564,443,'11-01-18','23-09-17');  insert into book\_lending values(3243,53656,45,'21-02-18','24-09-17');  insert into book\_lending values(0958,24564,4565,'23-12-17','25-09-17');  insert into book\_lending values(0958,24564,1,'29-12-17','29-09-17');  insert into book\_lending values(0007,77777,1,'31-12-17','30-09-17');

**Queries:**

**1. select book.bookid,title,pname,aname,noc,branchid from**

**book,book\_copies,book\_author where book.bookid=book\_author.bookid and  book.bookid=book\_copies.bookid order by branchid;**

BOOKID TITLE PNAME ANAME NOC BRANCHID



7 HP-TOOP bloomsbury JK.ROWLING 5 24564 958 FF:BOD marvel M.FARMER 10 24564 1004 HOBBIT alen-unwin JR.TOLKIEN 1 53565 3243 13RW varun RS.VARUN 7 53656 7 HP-TOOP bloomsbury J K.ROWLING 5 77777

**2. select cardno from book\_lending**

**where date\_out between '01-01-17' and '07-01-17' group by cardno having count(cardno)>=2  );**

CARDNO BRANCHID BOOKID

1 53565 1004

1 24564 958

1 77777 7

**3.delete from book where bookid=0007;**

BOOKID BRANCHID CARDNO DUEDATE DATE\_OUT

1004 53565 1 10-OCT-17 20-SEP-17

3243 53656 45 21-FEB-18 02-OCT-17

958 24564 4565 23-DEC-17 25-SEP-17

958 24564 1 29-DEC-17 29-SEP-17

**4.** CREATE VIEW YEAR AS SELECT pyear FROM BOOK;

**select \* from book order by(pyear);**

BOOKID TITLE PYEAR PNAME

1004 HOBBIT 1937 alen-unw

7 HP-TOOP 2007 bloomsbu

958 FF:BOD 2007 marvel

3243 13RW 2010 varun

**5. create view dbs as select bookid,title from**

**book; select \* from dbs;**

BOOKID TITLE

7 HP-TOOP

1004 HOBBIT

958 FF:BOD

3243 13RW

**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**

**a10 percent raise.**

**3. Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as**

**the maximum salary, the minimum salary, and the average salary in this department**

**4. Retrieve the name of each employee who works on all the projects controlled by**

**department number 5 (use NOT EXISTS operator). For each department that has more**

**than five employees, retrieve the department number and the number of its employees**

**who are making more than Rs. 6,00,000.**

**Table Creation;**

create table department (dno varchar(20) primary key,

dname varchar(20),

mgrstartdate date);

create table employee (ssn varchar(10) primary key,

fname varchar(10),

lname varchar(10),

addr varchar(30),

sex char (1),

salary float(10,2),

superssn varchar(10) references employee (ssn),

dno varchar(10) references department (dno));

alter table department add mgrssn varchar(10) references employee(ssn);

create table dlocation (dloc varchar(20),

dno varchar(10) references department (dno),

primary key (dno, dloc));

create table project (pno integer primary key,

pname varchar(20),

ploc varchar(20),

dno varchar(10) references department (dno));

create table workson (noh integer (2),

ssn varchar(10) references employee (ssn),

pno integer(4) references project(pno),

primary key (ssn, pno));

**Insertion of values to tables ;**

insert into EMPLOYEE values ('CEC01','John','Scott','Dallas,Houston','M',850000,NULL,NULL);

insert into EMPLOYEE values ('CEC02','James','Borg','Voss,Houston','M',770000,NULL,NULL);

insert into EMPLOYEE values ('CEC03','Alicia','Zeleya','Castie, Spring','F',780000,NULL,NULL);

insert into EMPLOYEE values('CEC04','Ramesh','Narayan','Banglore','M',800000,NULL,NULL);

insert into EMPLOYEE values ('CEC05','Reena','Sharma','Mumbai','F',580000,NULL,NULL);

insert into EMPLOYEE values ('CEC06','Aditya','Roy','Jaipur','M',850000,NULL,NULL);

insert into department values(1,'AppDev','2000-06-01','CEC01');

insert into department values(2,'Accounts','2000-05-25','CEC04');

insert into department values(3,'Robotics','2015-02-05','CEC06');

insert into department values(4,'Testing','2000-05-25','CEC01');

insert into department values(5,'ES','2014-11-25','CEC06');

update EMPLOYEE set Dno=1 where SSN='CEC01';

update EMPLOYEE set Dno=3 where SSN='CEC02';

update EMPLOYEE set Dno=4 where SSN='CEC03';

update EMPLOYEE set Dno=2 where SSN='CEC04';

update EMPLOYEE set Dno=5 where SSN='CEC05';

update EMPLOYEE set Dno=3 where SSN='CEC06';

update EMPLOYEE set SuperSSN='CEC06' where SSN='CEC01';

update EMPLOYEE set SuperSSN='CEC01' where SSN='CEC03';

update EMPLOYEE set SuperSSN='CEC05' where SSN='CEC02';

update EMPLOYEE set SuperSSN='CEC06' where SSN='CEC05';

insert into dlocation values('Texas',1);

insert into dlocation values('Mumbai',2);

insert into dlocation values('New Jersey',3);

insert into dlocation values('Jaipur',3);

insert into dlocation values('Bangalore',4);

insert into dlocation values('New Jersey',5);

insert into PROJECT values (111, 'IOT', 'Bangalore', 1),

(222, 'INSAT', 'Jaipur', 5),

(333, 'UDAN', 'Jaipur', 5),

(444, 'MetalDetector', 'Texas', 3),

(555, 'WarFeildSpying', 'Texas', 3);

insert into workson values (30,'CEC01',111);

insert into workson values (15,'CEC02',222);

insert into workson values (15,'CEC02',333);

insert into workson values (12,'CEC03',444);

insert into workson values (20,'CEC05',333);

insert into workson values (10,'CEC05',555);

insert into workson values (10,'CEC06',222);

insert into workson values (20,'CEC06',444);

insert into workson values (10,'CEC01',555);

**Queries**

**1)**

(select Pno

from Project P,Department D,Employee E

where P.Dno=D.Dno and D.MgrSSN=E.SSN and E.Lname='Scott')

union

(select Pno

from workson W,employee E

where W.SSN=E.SSN and E.Lname='Scott');

2)

select e.Fname,e.Salary, 1.1\*e.Salary Inc\_Salary

from employee e,project p, workson w

where e.SSN=w.SSN and p.Pno=w.Pno and p.Pname='IOT';

3)

select Dname,sum(Salary) TOT\_SAL, MAX(Salary) MAX\_SAL,

MIN(Salary) MIN\_SAL, AVG(Salary) AVG\_SAL

from employee natural join department

where Dname='Accounts';

4)

select E.fname

from employee e

where NOT EXISTS(select P.Pno

from Project P

where P.Dno=5 and

P.Pno not in(select W.Pno

from workson W

where E.ssn=W.ssn));

**5)**

select dno,count(\*)

from employee

where

dno in (select dno

from employee

group by dno

having count(\*)>1)

and

salary>600000

group by dno;