B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

DBMS Lab Report 2021

Submitted by

MOHAMMED ABDUL HAMID
(1BM19CS202)

Under the Guidance of
Vineetha
Assistant Professor, BMSCE

PROGRAM 1: INSURANCE DATABASE

Consider the Insurance database given below. The data types are specified.

PERSON (driver_id: String, name: String, address: String)
CAR (reg_num: String, model: String, year: int)
ACCIDENT (report_num: int, accident_date: date, location: String)
OWNS (driver_id: String, reg_num: String)

PARTICIPATED (driver_id: String,reg_num: String, report_num: int, damage_amount: int)

i) Create the above tables by properly specifying the primary keys and the foreign key

ii) Enter at least five tuples for each relation.

iii) Demonstrate how you

a.Update the damage amount to 25000 for the car with a specific reg-num(example 'K

A053408') for which the accident report number was 12.
b.Add a new accident to the database.
iv)Find the total number of people who owned cars that involved in accidents in 2008. v)Find the number of accidents in which cars belonging to a specific model (example)were involved.

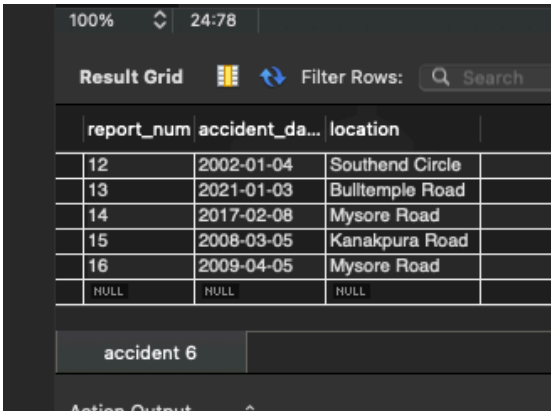
```
create database INSURANCE;

create table INSURANCE.person(

driver_id varchar(10),

name varchar(20),

address varchar(30),

primary key(driver_id)

);
```

```
create table INSURANCE.car(
reg_num varchar(10),
model varchar(10),
year int,
primary key(reg_num)
);
create table INSURANCE.accident(
report_num int,
accident_date date,
location varchar(20),
primary key(report_num)
);
create table INSURANCE.owns(
driver_id varchar(10),
reg_num varchar(10),
primary key(driver_id,reg_num),
foreign key(driver_id) references person(driver_id),
foreign key(reg_num) references car(reg_num)
);
create table INSURANCE.participated(
driver_id varchar(10),
reg_num varchar(10),
report_num int,
damage_amount int,
primary key(driver_id,reg_num,report_num),
foreign key(driver_id) references person(driver_id),
foreign key(reg_num) references car(reg_num),
```

```
foreign key(report_num) references accident(report_num)
);
use INSURANCE;
insert into person values('A01','Richard','Srinivas Nagar');
insert into person values('A02','Pradeep','Rajajinagar');
insert into person values('A03','Smith','Ashoknagar');
insert into person values('A04','Venu','N.R.Colony');
insert into person values('A05','John','Hanumanth Nagar');
select * from person;

insert into car values('KA052250','Indica', 1990);
insert into car values('KA031181','Lancer', 1957);
insert into car values('KA095477','Toyota', 1998);
insert into car values('KA053408','Honda', 2008);
insert into car values('KA041702','Audi', 2005);
select * from car;

insert into accident values(11,'2001-01-03','Mysore Road');
insert into accident values(12,'2002-01-04','Southend Circle');
insert into accident values(13,'2021-01-03','Bulltemple Road');
insert into accident values(14,'2017-02-08','Mysore Road');
insert into accident values(15,'2008-03-05 ','Kanakpura Road');
select * from accident;

insert into owns values('A01','KA052250');
insert into owns values('A02','KA053408');
insert into owns values('A03','KA031181');
```

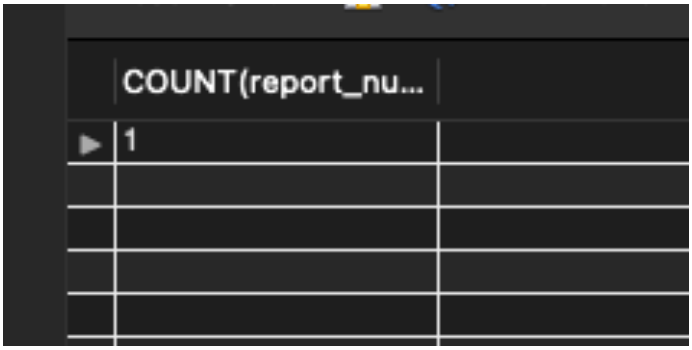
```
insert into owns values('A04','KA095477');

insert into owns values('A05','KA041702');

select * from owns;


insert into participated values('A01','KA052250',11,10000);

insert into participated values('A02','KA053408',12,50000);

insert into participated values('A03','KA095477',13,25000);

insert into participated values('A04','KA031181',14,3000);

insert into participated values('A05','KA041702',15,5000);

select * from participated;

use INSURANCE;

UPDATE participated

SET damage_amount=25000

WHERE report_num=12;


insert into accident values('16','2009-04-05','Mysore Road');

select * from accident;

use INSURANCE;

SELECT COUNT(DISTINCT driver_id) FROM accident, participated

WHERE accident.report_num = participated.report_num

AND accident_date LIKE '2008%'


use INSURANCE;

SELECT COUNT(report_num) FROM car, participated

WHERE car.reg_num = participated.reg_num

AND model='Lancer';
```

OUPUT -



Query3

Query4



Query5

PROGRAM 2: BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.

Branch (branch-name: String, branch-city: String, assets: real)
BankAccount(accno: int, branch-name: String, balance: real)
BankCustomer (customer-name: String, customer-street: String, customer-city: String)
Depositer(customer-name: String, accno: int)
Loan (loan-number: int, branch-name: String, amount: real)

i. Create the above tables by properly specifying the primary keys and the foreign keys.
ii. Enter at least five tuples for each relation.
iii. Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).

iv. Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi).
v. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

```
create database bank;

use bank;


create table branch (
        branch_name varchar(25),
   branch_city varchar(15),
   assets int,
   primary key (branch_name)
);


create table bank_account (
        accno int,
   branch_name varchar(25),
   balance int,
   primary key (accno),
```

```
    foreign key (branch_name) references branch(branch_name)
);

create table bank_customer (
        customer_name varchar(10),
    customer_street varchar(25),
    customer_city varchar(15),
    primary key (customer_name)
);

create table depositer (
        customer_name varchar(10),
        accno int,
    primary key(customer_name, accno),
    foreign key (customer_name) references bank_customer(customer_name),
    foreign key (accno) references bank_account(accno)
);

create table loan (
        loan_number int,
    branch_name varchar(25),
    amount int,
    primary key (loan_number),
    foreign key (branch_name) references branch(branch_name)
);

insert into branch values('SBI_Chamrajpet', 'Bangalore', 50000);
insert into branch values('SBI_ResidencyRoad', 'Bangalore', 10000);
insert into branch values('SBI_ShivajiRoad', 'Bombay', 20000);
```

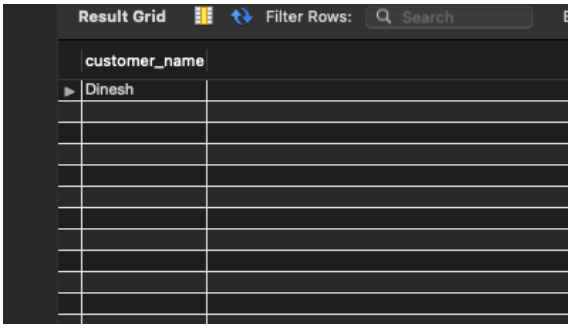
```
insert into branch values('SBI_ParliamentRoad', 'Delhi', 10000);
insert into branch values('SBI_Jantarmantar', 'Delhi', 20000);
commit;

insert into bank_account values(1, 'SBI_Chamrajpet', 2000);
insert into bank_account values(2, 'SBI_ResidencyRoad', 5000);
insert into bank_account values(3, 'SBI_ShivajiRoad', 6000);
insert into bank_account values(4, 'SBI_ParliamentRoad', 9000);
insert into bank_account values(5, 'SBI_Jantarmantar', 8000);
insert into bank_account values(6, 'SBI_ShivajiRoad', 4000);
insert into bank_account values(8, 'SBI_ResidencyRoad', 4000);
insert into bank_account values(9, 'SBI_ParliamentRoad', 3000);
insert into bank_account values(10, 'SBI_ResidencyRoad', 5000);
insert into bank_account values(11, 'SBI_Jantarmantar', 2000);
commit;

insert into bank_customer values ('Avinash', 'Bull_Temple_Road', 'Bangalore');
insert into bank_customer values ('Dinesh', 'Bannergatta_Road', 'Bangalore');
insert into bank_customer values ('Mohan', 'National_College_Road', 'Bangalore');
insert into bank_customer values ('Nikhil', 'Akbar_Road', 'Delhi');
insert into bank_customer values ('Ravi', 'Prithviraj_Road', 'Delhi');
commit;

insert into depositer values('Avinash', 1);
insert into depositer values('Dinesh', 2);
insert into depositer values('Nikhil', 4);
insert into depositer values('Ravi', 5);
insert into depositer values('Avinash', 8);
insert into depositer values('Nikhil', 9);
```

```
insert into depositer values('Dinesh', 10);

insert into depositer values('Nikhil', 11);

commit;


insert into loan values(1, 'SBI_Chamrajpet', 1000);

insert into loan values(2, 'SBI_ResidencyRoad', 2000);

insert into loan values(3, 'SBI_ShivajiRoad', 3000);

insert into loan values(4, 'SBI_ParliamentRoad', 4000);

insert into loan values(5, 'SBI_Jantarmantar', 5000);

commit;



select * from branch;

select * from bank_account;

select * from bank_customer;

select * from depositer;

select * from loan;


-- Query 3

select distinct c.customer_name from bank_customer c,bank_account b where exists(select
d.customer_name,count(d.customer_name) from depositer d,bank_account ba where ba.accno = d.accno and

c.customer_name = d.customer_name and ba.branch_name = 'SBI_ResidencyRoad' group by
d.customer_name having count(d.customer_name)>=2);


-- Query 4

select d.customer_name from depositer d,branch b,bank_account a

where b.branch_name=a.branch_name

AND a.accno=d.accno

and branch_city='Delhi'

group by d.customer_name
```

```
HAVING COUNT(distinct b.branch_name)=(
        SELECT COUNT(branch_name)
        FROM branch
        WHERE branch_city='Delhi');
```

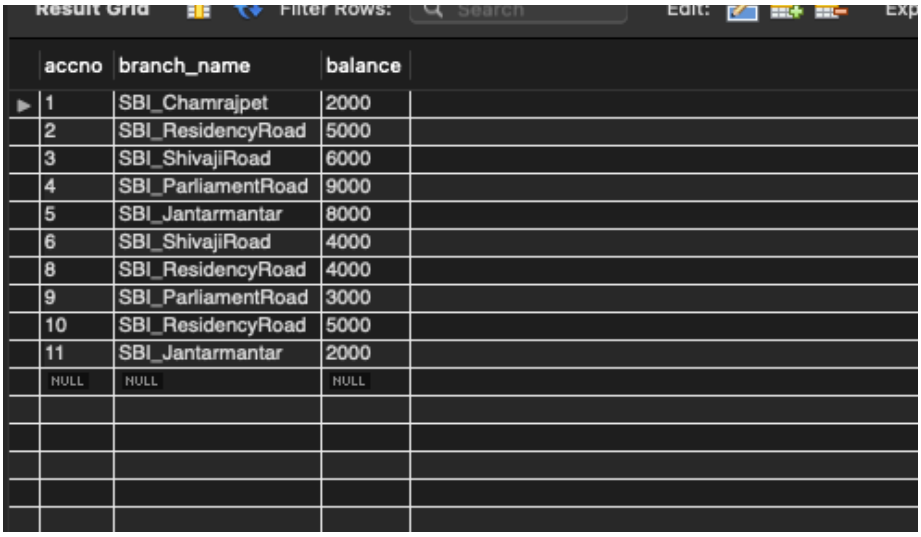
-- Query 5

```
delete from bank_account where branch_name in (select branch_name from branch where branch_city =
'Bombay');

select * from bank_account;
```



Query3

Query4



Query5

PROGRAM 3: SUPPLIER DATABASE
Consider the following schema:
SUPPLIERS(sid: integer, sname: string, address: string) PARTS(pid: integer, pname: string, color: string) CATALOG(sid: integer, pid: integer, cost: real)

The Catalog relation lists the prices charged for parts by Suppliers. Write the following queries in SQL:

- i) Find the pnames of parts for which there is some supplier.
- ii) Find the snames of suppliers who supply every part.
- iii) Find the snames of suppliers who supply every red part.
- iv) Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.
- v) Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).

vi) For each part, find the sname of the supplier who charges the most for that part.

```
create database supplier;
use supplier;

create table suppliers(
        sid int primary key,
    sname varchar(30),
    address varchar(30)
);
create table parts(
        pid int primary key,
    pname varchar(30),
```

```
    color varchar(30)
);

create table catalog (
            sid int ,
    pid int ,
    cost real,
    constraint c_sid foreign key(sid) references suppliers(sid) ,
    constraint c_pid foreign key(pid) references parts(pid)
)

insert into suppliers values(1,'Acme Widget','kolkata') ;
insert into suppliers values(2,'Tata','bengaluru') ;
insert into suppliers values(3,'Reebok','delhi') ;
insert into suppliers values(4,'Nike','delhi') ;
insert into suppliers values(5,'Reliance','delhi') ;

insert into parts values(1,'paint','red') ;
insert into parts values(2,'steel','black') ;
insert into parts values(3,'spray','red') ;
insert into parts values(4,'sheet','green');
insert into parts values(5,'tiles','blue');
delete from parts where pid=5;

insert into catalog values(1,1,100);
insert into catalog values(1,2,200);
insert into catalog values(1,3,200);
```

```
insert into catalog values(1,4,100);

insert into catalog values(2,1,300);

insert into catalog values(2,2,100);

insert into catalog values(3,2,90);

insert into catalog values(3,3,110);

insert into catalog values(3,4,110);

insert into catalog values(4,1,100);

insert into catalog values(4,3,120);

insert into catalog values(4,4,130);



select * from catalog;

select * from parts;


--      i. Find the pnames of parts for which there is some supplier.

insert into parts values(5,'tiles','blue');

select p.pname from parts p where p.pid in (select pid from catalog c group by c.pid having
count(c.sid)>0);

insert into catalog values(1,5,140);

select p.pname from parts p where p.pid in (select pid from catalog c group by c.pid having
count(c.sid)>0);

delete from catalog where pid=5;

delete from parts where pid=5;


-- ii. Find the snames of suppliers who supply every part.

select s.sname from suppliers s where s.sid in (select c.sid from catalog c group by c.sid having
count(distinct (c.pid))=(select count(p.pid) from parts p));
```

-- iii. Find the snames of suppliers who supply every red part.

```
select s.sname from suppliers s where s.sid in (select ca.sid  from catalog ca,parts p where ca.pid=p.pid and p.color='red' group by ca.sid having count(ca.pid)=(select count(*) from parts p where p.color='red'));
```

-- iv. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.

```
select ca.pid from catalog ca  where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget') having (select count(c.pid) from catalog c where c.pid=ca.pid)=1;
```

-- v. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over

-- all the suppliers who supply that part).

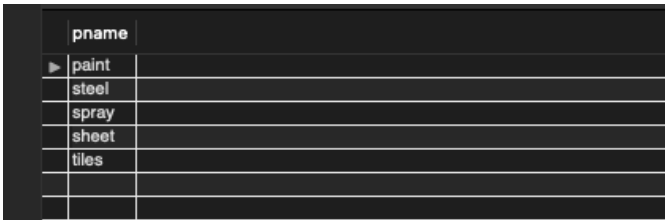
```
select distinct c.sid,c.pid from catalog c where c.cost > (select avg(ca.cost) from catalog ca where ca.pid=c.pid);
```

-- vi. For each part, find the sname of the supplier who charges the most for that part.

```
select s.sname from suppliers s where s.sid in (select c.sid from catalog c where c.cost=(select max(cost) from catalog ca where ca.pid=c.pid));
```

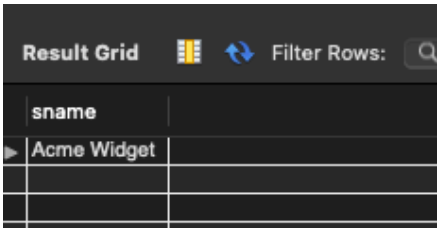
-- vii. select supplier who sell only red parts

```
select s.sname from suppliers s where s.sid in(select c.sid from catalog c where c.sid not in (select distinct(ca.sid) from catalog ca,parts p where ca.pid=p.pid and p.color!='red'));

insert into catalog values(5,1,140);

select s.sname from suppliers s where s.sid in(select c.sid from catalog c where c.sid not in (select distinct(ca.sid) from catalog ca,parts p where ca.pid=p.pid and p.color!='red'));

delete from catalog where sid=5;
```



pname
paint
steel
spray
sheet
tiles

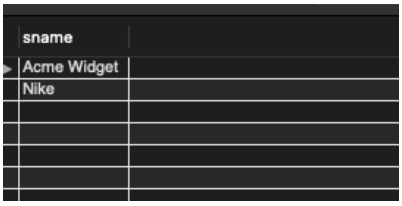
Query1



Result Grid Filter Rows:

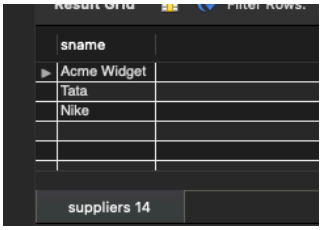
sname
Acme Widget

Query2



sname
Acme Widget
Nike

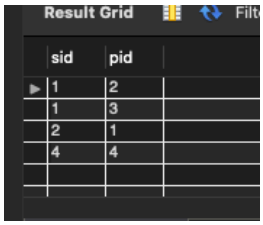
Query3



sname
Acme Widget
Tata
Nike

suppliers 14

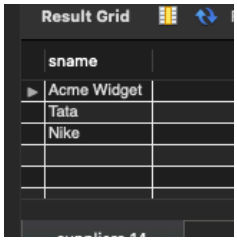
Query4



Result Grid Filt

sid	pid
1	2
1	3
2	1
4	4

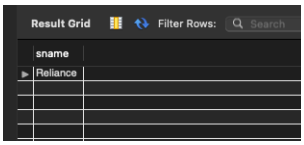
Query5



Result Grid

sname
Acme Widget
Tata
Nike

Query6



Result Grid Filter Rows: Search

sname
Reliance

Query7

PROGRAM 4: STUDENT FACULTY DATABASE

Consider the following database for student enrollment for course :
STUDENT(snum: integer, sname: string, major: string, lvl: string, age: integer) CLASS(cname: string, meets at: time, room: string, fid: integer)
ENROLLED(snum: integer, cname: string)
FACULTY(fid: integer, fname: string, deptid: integer)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level(lvl) is a two character code with 4 different values (example: Junior: JR etc)
Write the following queries in SQL. No duplicates should be printed in any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

vi. Find the names of students who are not enrolled in any class.

vii. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

```
CREATE DATABASE student_faculty;
USE student_faculty;
CREATE TABLE student(
        snum INT,
        sname VARCHAR(10),
        major VARCHAR(2),
        lvl VARCHAR(2),
        age INT, primary key(snum));

CREATE TABLE faculty(
        fid INT,fname VARCHAR(20),
        deptid INT,
    PRIMARY KEY(fid));

CREATE TABLE class(
        cname VARCHAR(20),
        metts_at TIMESTAMP,
        room VARCHAR(10),
    fid INT,
        PRIMARY KEY(cname),
        FOREIGN KEY(fid) REFERENCES faculty(fid));
```

```
CREATE TABLE enrolled(
        snum INT,
        cname VARCHAR(20),
        PRIMARY KEY(snum,cname),
        FOREIGN KEY(snum) REFERENCES student(snum),
        FOREIGN KEY(cname) REFERENCES class(cname));

INSERT INTO STUDENT VALUES(1, 'jhon', 'CS', 'Sr', 19);
INSERT INTO STUDENT VALUES(2, 'Smith', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(3 , 'Jacob', 'CV', 'Sr', 20);
INSERT INTO STUDENT VALUES(4, 'Tom ', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(5, 'Rahul', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(6, 'Rita', 'CS', 'Sr', 21);

INSERT INTO FACULTY VALUES(11, 'Harish', 1000);
INSERT INTO FACULTY VALUES(12, 'MV', 1000);
INSERT INTO FACULTY VALUES(13 , 'Mira', 1001);
INSERT INTO FACULTY VALUES(14, 'Shiva', 1002);
INSERT INTO FACULTY VALUES(15, 'Nupur', 1000);
```

```
insert into class values('class1', '12/11/15 10:15:16', 'R1', 14);
insert into class values('class10', '12/11/15 10:15:16', 'R128', 14);
insert into class values('class2', '12/11/15 10:15:20', 'R2', 12);
insert into class values('class3', '12/11/15 10:15:25', 'R3', 11);
insert into class values('class4', '12/11/15 20:15:20', 'R4', 14);
insert into class values('class5', '12/11/15 20:15:20', 'R3', 15);
insert into class values('class6', '12/11/15 13:20:20', 'R2', 14);
insert into class values('class7', '12/11/15 10:10:10', 'R3', 14);

insert into enrolled values(1, 'class1');
insert into enrolled values(2, 'class1');
insert into enrolled values(3, 'class3');
insert into enrolled values(4, 'class3');
insert into enrolled values(5, 'class4');
insert into enrolled values(1, 'class5');
insert into enrolled values(2, 'class5');
insert into enrolled values(3, 'class5');
insert into enrolled values(4, 'class5');
insert into enrolled values(5, 'class5');
```

```
-- Query 1

SELECT DISTINCT S.Sname

FROM Student S, Class C, Enrolled E, Faculty F

WHERE S.snum = E.snum AND E.cname = C.cname AND C.fid = F.fid
AND

F.fname = 'Harish' AND S.lvl = 'Jr';



-- Query 2

SELECT DISTINCT cname

        FROM class

        WHERE room='R128'

        OR

        cname IN (SELECT e.cname FROM enrolled e GROUP BY e.cname
HAVING COUNT(*)>=5);




-- Query 3

SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum IN (SELECT E1.snum

                        FROM Enrolled E1, Enrolled E2, Class C1, Class C2
```

```
                    WHERE E1.snum = E2.snum AND E1.cname <>
E2.cname

                    AND E1.cname = C1.cname

                    AND E2.cname = C2.cname AND C1.metts_at =
C2.metts_at);


-- Query 4

SELECT f.fname,f.fid

                    FROM faculty f

               WHERE f.fid in ( SELECT fid FROM class

                    GROUP BY fid HAVING COUNT(*)=(SELECT
COUNT(DISTINCT room) FROM class) );


-- Query 5

SELECT DISTINCT F.fname

FROM Faculty F

WHERE 5 > (SELECT COUNT(E.snum)

FROM Class C, Enrolled E

WHERE C.cname = E.cname

AND C.fid = F.fid);


-- Query 6

SELECT DISTINCT S.sname
```

```
FROM Student S
WHERE S.snum NOT IN (SELECT E.snum
FROM Enrolled E );

-- Query 7
SELECT S.age, S.lvl
FROM STUDENT S
GROUP BY S.age, S.lvl
HAVING S.lvl IN(SELECT S1.lvl
        FROM STUDENT S1
        WHERE S1.age=S.age
        GROUP BY S1.age, S1.lvl
        HAVING COUNT(*)  >= ALL (SELECT COUNT(*)
                FROM STUDENT S2
                WHERE S1.age=S2.age
                GROUP BY S2.lvl, S2.age))
ORDER BY S.age;
```

Result Grid Filter Rows:

Sname
Tom

Query1

Result Grid Filter Rows

cname
class10
class5
NULL

Query2

sname
Rahul

Query3

Result Grid Filter Rows:

fname	fid
Shiva	14
NULL	NULL

Query4

Result Grid Filter Rows:

fname
Harish
MV
Mira
Shiva

Query5

Result Grid Filter Rows: Search

sname
Rita

Query6

age	lvl
19	Sr
20	Jr
21	Sr

Query7

PROGRAM 5: AIRLINE FLIGHT DATABASE

Consider the following database that keeps track of airline flight information:
FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
AIRCRAFT(aid: integer, aname: string, cruisingrange: integer)
CERTIFIED(eid: integer, aid: integer)
EMPLOYEES(eid: integer, ename: string, salary: integer)
Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft, and only pilots are certified to fly.
Write each of the following queries in SQL.

i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

ii. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.

iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

iv. For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

v. Find the names of pilots certified for some Boeing aircraft.

vi. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

vii. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

```
create database flightdb;
use flightdb;

create table flights(
        flno int,
    fromplace varchar(15),
    toplace varchar(15),
    distance int,
    departs datetime,
    arrives datetime,
    price int,
    primary key (flno)
);
desc flights;
create table aircraft(
        aid int,
    aname varchar(15),
    cruisingrange int,
    primary key (aid)
);
desc aircraft;
create table employees (
        eid int,
```

```
    ename varchar(15),
    salary int,
    primary key (eid)
);
desc employees;
create table certified (
        eid int,
    aid int,
    foreign key (eid) references employees(eid),
    foreign key (aid) references aircraft(aid)
);
desc certified;
insert into flights values(101, 'Bangalore', 'Delhi', 2500, '2005-05-13 07:15:31',
'2005-05-13 18:15:31', 5000);
insert into flights values(102, 'Bangalore', 'Lucknow', 3000, '2013-05-05 07:15:31',
'2013-05-05 11:15:31', 6000);
insert into flights values(103, 'Lucknow', 'Delhi', 500, '2013-05-05 12:15:31',
'2013-05-05 17:15:31', 3000);
insert into flights values(107, 'Bangalore', 'Frankfurt', 8000, '2013-05-05 07:15:31',
'2013-05-05 22:15:31', 60000);
insert into flights values(104, 'Bangalore', 'Frankfurt', 8500, '2013-05-05 07:15:31',
'2013-05-05 23:15:31', 75000);
insert into flights values(105, 'Kolkata', 'Delhi', 3400, '2013-05-05 07:15:31',
'2013-05-05 09:15:31', 7000);
insert into flights values(106, 'Bangalore', 'Kolkata', 1000, '2013-05-05 01:15:30',
'2013-05-05 09:20:30', 10000);
```

```
insert into flights values(108, 'Lucknow', 'Kolkata', 1000, '2013-05-05 11:30:30',
'2013-05-05 15:20:30', 10000);

commit;

select * from flights;

insert into aircraft values(101, '747', 3000);
insert into aircraft values(102, 'Boeing', 900);
insert into aircraft values(103, '647', 800);
insert into aircraft values(104, 'Dreamliner', 10000);
insert into aircraft values(105, 'Boeing', 3500);
insert into aircraft values(106, '707', 1500);
insert into aircraft values(107, 'Dream', 120000);
insert into aircraft values(108, '707', 760);
insert into aircraft values(109, '747', 1000);
commit;

select * from aircraft;

insert into employees values(701, 'A', 50000);
insert into employees values(702, 'B', 100000);
insert into employees values(703, 'C', 150000);
```

```
insert into employees values(704, 'D', 90000);
insert into employees values(705, 'E', 40000);
insert into employees values(706, 'F', 60000);
insert into employees values(707, 'G', 90000);
commit;

select * from employees;

insert into certified values(701, 101);
insert into certified values(701, 102);
insert into certified values(701, 106);
insert into certified values(701, 105);

insert into certified values(702, 104);
insert into certified values(703, 104);
insert into certified values(704, 104);

insert into certified values(702, 107);
insert into certified values(703, 107);
insert into certified values(704, 107);

insert into certified values(702, 101);
insert into certified values(702, 108);
```

```
insert into certified values(701, 109);

commit;

select * from certified;


-- Query 1

select distinct a.aname from aircraft a where a.aid in (

        select c.aid from certified c, employees e where

    c.eid = e.eid and not exists(

                select * from employees e1 where e1.eid=e.eid and e1.salary<80000

    )

);


-- Query 2

select max(a.cruisingrange), c.eid from certified c, aircraft a where c.aid = a.aid
group by c.eid having count(c.eid)>3;


-- Query 3

select ename from employees where salary <(

select min(price) from flights where fromplace='Bangalore' and toplace='Frankfurt');


-- Query 4


select avg(e.salary), c.aid from certified c, employees e where c.aid in(
```

```
select aid from aircraft where cruisingrange>1000) and e.eid = c.eid group by c.aid;


-- Query 5

select ename from employees where eid in(

select eid from certified where aid in(

select aid from aircraft where aname = 'Boeing'));


-- Query 6


select aname from aircraft where cruisingrange > any (select distance from flights
where fromplace='Bangalore' and toplace='Delhi');




-- Query 7

SELECT F.flno, F.departs

FROM flights F

WHERE F.flno IN ( ( SELECT F0.flno

 FROM flights F0

 WHERE F0.fromplace = 'Bangalore' AND F0.toplace = 'Kolkata'

 AND extract(hour from F0.arrives) < 18 )

 UNION

( SELECT F0.flno

 FROM flights F0, flights F1
```

```
 WHERE F0.fromplace = 'Bangalore' AND F0.toplace <> 'Kolkata'
 AND F0.toplace = F1.fromplace AND F1.toplace = 'Kolkata'
 AND F1.departs > F0.arrives
 AND extract(hour from F1.arrives) < 18)
 UNION
( SELECT F0.flno
 FROM flights F0, flights F1, flights F2
 WHERE F0.fromplace = 'Bangalore'
 AND F0.toplace = F1.fromplace
 AND F1.toplace = F2.fromplace
 AND F2.toplace = 'Kolkata'
 AND F0.toplace <> 'Kolkata'
 AND F1.toplace <> 'Kolkata'
 AND F1.departs > F0.arrives
 AND F2.departs > F1.arrives
 AND extract(hour from F2.arrives) < 18));
```

Result Grid

aname
747
Dreamliner
Dream
707

Query1

max(a.cruisingran...	eid
3500	701
120000	702

Query2

ename
A
E

Query3

Result Grid Filter R

avg(e.salary)	aid
75000.0000	101
113333.3333	104
50000.0000	105
50000.0000	106
113333.3333	107

Query4

Result Grid

ename
A

Query5

aname
747
Dreamliner
Boeing
Dream

Query6

Result Grid Filter Rc

flno	departs
102	2013-05-05 07:15:31
106	2013-05-05 01:15:30

Query7

Program 6 : Order Database

Consider the following schema for Order Database:

SALESMAN (*Salesman_id, Name, City, Commission*) CUSTOMER (*Customer_id, Cust_Name, City, Grade, Salesman_id*)

DATABASE MANAGEMENT SYSTEM LABORATORY MANUAL

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.

```
create database ORDER_H;

use ORDER_H;



CREATE TABLE SALESMAN (SALESMAN_ID INT, S_NAME VARCHAR (20),

CITY varchar (20), COMMISSION INT, PRIMARY KEY (SALESMAN_ID));



DESC SALESMAN;
```

```
CREATE TABLE CUSTOMER1 (CUSTOMER_ID INT, CUST_NAME VARCHAR (20),
CITY VARCHAR (20),

GRADE INT,PRIMARY KEY (CUSTOMER_ID), SALESMAN_ID INT REFERENCES
SALESMAN (SALESMAN_ID) ON DELETE SET NULL);


DESC CUSTOMER1;


CREATE TABLE ORDERS (ORD_NO INT, PURCHASE_AMT INT, ORD_DATE DATE,

PRIMARY KEY (ORD_NO),

CUSTOMER_ID INT REFERENCES CUSTOMER1 (CUSTOMER_ID) ON DELETE
CASCADE, SALESMAN_ID INT REFERENCES SALESMAN (SALESMAN_ID) ON
DELETE CASCADE);


INSERT INTO SALESMAN VALUES (1000,  "JOHN","BANGALORE",25);

INSERT INTO SALESMAN VALUES (2000,  "RAVI","BANGALORE",20);

INSERT INTO SALESMAN VALUES (3000,  "KUMAR","MYSORE",15);

INSERT INTO SALESMAN VALUES (4000,  "SMITH","DELHI",30);

INSERT INTO SALESMAN VALUES (5000,  "HARSHA","HYDRABAD",15);


SELECT * FROM SALESMAN;




INSERT INTO CUSTOMER1 VALUES (10,  "PREETHI","BANGALORE", 100, 1000);

INSERT INTO CUSTOMER1 VALUES (11,  "VIVEK","MANGALORE", 300, 1000);

INSERT INTO CUSTOMER1 VALUES (12,  "BHASKAR","CHENNAI", 400, 2000);

INSERT INTO CUSTOMER1 VALUES (13,  "CHETHAN","BANGALORE", 200, 2000);

INSERT INTO CUSTOMER1 VALUES (14,  "MAMATHA","BANGALORE", 400, 3000);
```

```
SELECT * FROM CUSTOMER1;

INSERT INTO ORDERS VALUES (50, 5000, '04-05-17', 10, 1000);
INSERT INTO ORDERS VALUES (51, 450,  '20-01-17', 10, 2000);
INSERT INTO ORDERS VALUES (52, 1000,  '24-02-17', 13, 2000);
INSERT INTO ORDERS VALUES (53, 3500,  '13-04-17', 14, 3000);
INSERT INTO ORDERS VALUES (54, 550,  '09-03-17', 12, 2000);
SELECT * FROM ORDERS;


SELECT GRADE, COUNT(DISTINCT CUSTOMER_ID) FROM CUSTOMER1
GROUP BY GRADE
HAVING GRADE > (SELECT AVG(GRADE)
FROM CUSTOMER1
WHERE CITY='BANGALORE');

SELECT SALESMAN_ID, S_NAME FROM SALESMAN A
WHERE 1 < (SELECT COUNT(*) FROM CUSTOMER1
WHERE SALESMAN_ID=A.SALESMAN_ID);

SELECT SALESMAN.SALESMAN_ID, S_NAME, CUST_NAME, COMMISSION
FROM SALESMAN, CUSTOMER1
WHERE SALESMAN.CITY = CUSTOMER1.CITY
UNION
SELECT SALESMAN_ID, S_NAME, 'NO MATCH', COMMISSION FROM SALESMAN
WHERE NOT CITY = ANY
```

```
(SELECT CITY

FROM CUSTOMER1)

ORDER BY 2 DESC;


CREATE VIEW ELITSALESMAN AS SELECT B.ORD_DATE, A.SALESMAN_ID,
A.S_NAME

FROM SALESMAN A, ORDERS B

WHERE A.SALESMAN_ID = B.SALESMAN_ID

AND B.PURCHASE_AMT=(SELECT MAX(PURCHASE_AMT) FROM ORDERS C

WHERE C.ORD_DATE = B.ORD_DATE);



DELETE FROM SALESMAN

WHERE SALESMAN_ID=1000;

SELECT * FROM SALESMAN;
```

Result Grid Filter Rows: Search

GRADE	COUNT(DISTINCT CUSTOMER_ID)
300	1
400	2

Query1

SALESMAN_ID	S_NAME
2000	RAVI
NULL	NULL

Query2

SALESMAN_ID	S_NAME	CUST_NAME	COMMISSION
4000	SMITH	NO MATCH	30
2000	RAVI	PREETHI	20
2000	RAVI	CHETHAN	20
2000	RAVI	MAMATHA	20
3000	KUMAR	NO MATCH	15
5000	HARSHA	NO MATCH	15

Query3

Result Grid Filter Rows: Search

SALESMAN_ID	S_NAME	CUST_NAME	COMMISSION
4000	SMITH	NO MATCH	30
2000	RAVI	PREETHI	20
2000	RAVI	CHETHAN	20
2000	RAVI	MAMATHA	20
3000	KUMAR	NO MATCH	15
5000	HARSHA	NO MATCH	15

Query4

Result Grid Filter Rows: Search

SALESMAN_ID	S_NAME	CITY	COMMISSION
2000	RAVI	BANGALORE	20
3000	KUMAR	MYSORE	15
4000	SMITH	DELHI	30
5000	HARSHA	HYDRABAD	15
NULL	NULL	NULL	NULL

Query5

Program 7 : Book 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.**

```
create database Lab7;

use Lab7;



create table publisher (

   name varchar (20) primary key,

   phone integer,
```

```
    address varchar (20)
);
desc publisher;

create table book (
    book_id integer primary key,
    title varchar (20),
    pub_year varchar (20),
    publisher_name varchar (20),
    foreign key (publisher_name) references publisher (name) on delete cascade
);
desc book;

create table book_authors (
    author_name varchar (20),
    book_id integer,
    foreign key (book_id) references book (book_id) on delete cascade,
    primary key (book_id, author_name)
);
desc book_authors;
```

```
create table library_branch (
   branch_id integer primary key,
   branch_name varchar (50),
   address varchar (50)
);
desc library_branch;

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

create table card (
   card_no integer primary key
);
desc card;
```

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

insert into publisher values ('mcgraw-hill', 99890, 'bangalore');
insert into publisher values ('pearson', 98890, 'newdelhi');
insert into publisher values ('random house', 74556, 'hyderabad');
insert into publisher values ('hachette livre', 897086, 'chenai');
insert into publisher values ('grupo planeta', 77561, 'bangalore');
select * from publisher;
```

```
insert into book values (1,'dbms','01-2017', 'mcgraw-hill');

insert into book values (2,'adbms','06-2016', 'mcgraw-hill');

insert into book values (3,'cn','09-2016', 'pearson');

insert into book values (4,'cg','09-2015', 'grupo planeta');

insert into book values (5,'os','05-2016', 'pearson');

select * from book;


insert into book_authors values ('navathe', 1);

insert into book_authors values ('navathe', 2);

insert into book_authors values ('tanenbaum', 3);

insert into book_authors values ('edward angel', 4);

insert into book_authors values ('galvin', 5);

select * from book_authors;


insert into library_branch values (10,'rr nagar','bangalore');

insert into library_branch values (11,'rnsit','bangalore');

insert into library_branch values (12,'rajaji nagar', 'bangalore');

insert into library_branch values (13,'nitte','mangalore');

insert into library_branch values (14,'manipal','udupi');

select * from library_branch;
```

```
insert into book_copies values (10, 1, 10);

insert into book_copies values (5, 1, 11);

insert into book_copies values (2, 2, 12);

insert into book_copies values (5, 2, 13);

insert into book_copies values (7, 3, 14);

insert into book_copies values (1, 5, 10);

insert into book_copies values (3, 4, 11);

select * from book_copies;


insert into card values (100);

insert into card values (101);

insert into card values (102);

insert into card values (103);

insert into card values (104);

select * from card;


insert into book_lending values ('01-01-17','01-06-17', 1, 10, 101);

insert into book_lending values ('11-01-17','11-03-17', 3, 14, 101);

insert into book_lending values ('21-02-17','21-04-17', 2, 13, 101);

insert into book_lending values ('15-03-17','15-07-17', 4, 11, 101);

insert into book_lending values ('12-08-17','12-08-17', 1, 11, 104);
```

```
select * from book_lending;

select b.book_id, b.title, b.pub_year, b.publisher_name, bc.no_of_copies,
        ba.author_name, lb.branch_name from book b, book_authors ba,

library_branch lb, book_copies bc where b.book_id = ba.book_id and b.book_id =
        bc.book_id and lb.branch_id = bc.branch_id;

select card_no from book_lending where year(date_out) >17 and month(date_out)<7
        group by card_no having count(card_no) >2 ;

delete from book where book_id = 3;

select * from book;

select * from book_authors;

select * from book_copies;

select * from book_lending;
```

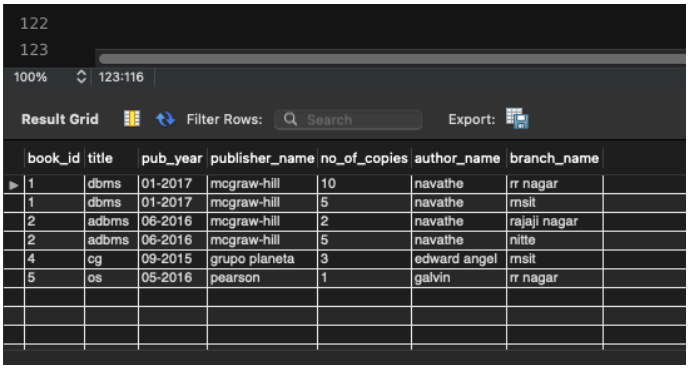
create view q4_view as select pub_year from book;

select * from q4_view;

create view q5_view as select b.book_id, b.title, bc.no_of_copies from book b,

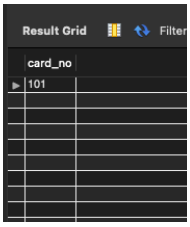
book_copies bc where b.book_id = bc.book_id;

select * from q5_view;

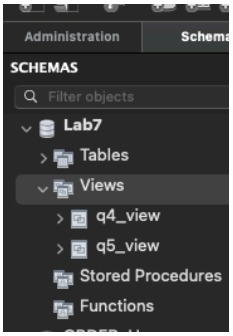


book_id	title	pub_year	publisher_name	no_of_copies	author_name	branch_name
1	dbms	01-2017	mcgraw-hill	10	navathe	rr nagar
1	dbms	01-2017	mcgraw-hill	5	navathe	rnsit
2	adbms	06-2016	mcgraw-hill	2	navathe	rajaji nagar
2	adbms	06-2016	mcgraw-hill	5	navathe	nitte
4	cg	09-2015	grupo planeta	3	edward angel	rnsit
5	os	05-2016	pearson	1	galvin	rr nagar

Query1



Query2



Query
3,4,5

Program 8:

Consider the following database of student enrollment in courses & books adopted for each course. STUDENT (regno: string, name: string, major: string, bdate:date)
COURSE (course #:int, cname:string, dept:string)
ENROLL (regno:string, course#:int, sem:int, marks:int)

BOOK _ ADOPTION (course# :int, sem:int, book-ISBN:int)

TEXT (book-ISBN:int, book-title:string, publisher:string, author:string)

 i. Create the above tables by properly specifying the primary keys and the foreign keys.
ii. Enter at least five tuples for each relation.
iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.
v. List any department that has all its adopted books published by a specific publisher.
vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the results.

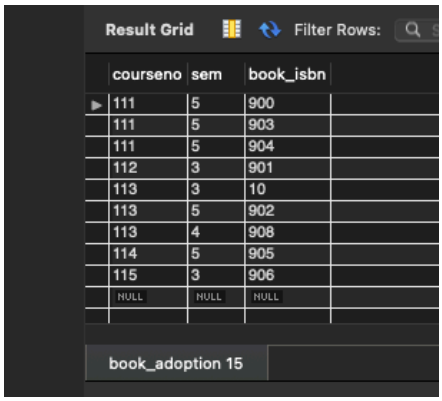
```
create database student_enroll;
use student_enroll;

create table student(
regno varchar(15),
name varchar(20),
major varchar(20),
bdate date,
primary key(regno));
desc student;

create table course(
courseno int,
cname varchar(20),
dept varchar(20),
primary key(courseno));
desc course;

create table enroll(
regno varchar(15),
courseno int,
sem int,
marks int,
primary key(regno,courseno),
```

```
foreign key(regno) references student(regno),
foreign key(courseno) references course(courseno));
desc enroll;
create table textbook(
book_isbn int,
book_title varchar(20),
publisher varchar(20),
author varchar(20),
primary key(book_isbn));
desc textbook;

create table book_adoption(
courseno int,
sem int,
book_isbn int,
primary key(courseno,book_isbn),
foreign key(courseno) references course(courseno),
foreign key(book_isbn) references textbook(book_isbn));
desc book_adoption;

insert into student values('1BM11CS001','A','Sr','19931230');
insert into student values('1BM11CS002','B','Sr','19930924');
insert into student values('1BM11CS003','C','Sr','19931127');
insert into student values('1BM11CS004','D','Sr','19930413');
insert into student values('1BM11CS005','E','Jr','19940824');
commit;
```

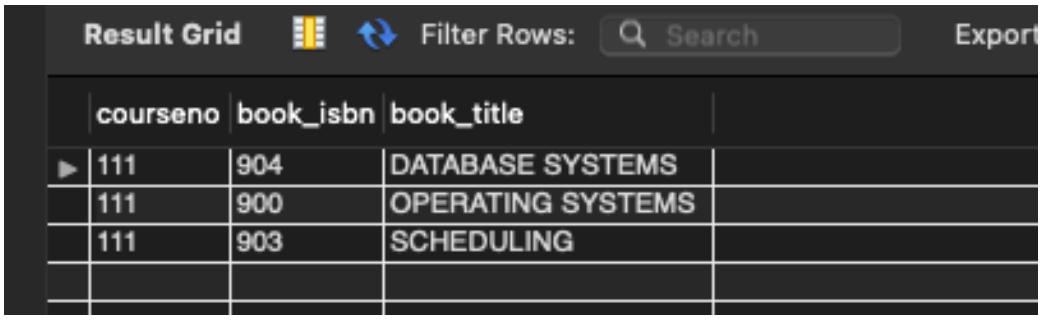
```
select * from student;

insert into course values(111,'OS','CSE');
insert into course values(112,'EC','ECE');
insert into course values(113,'SS','ISE');
insert into course values(114,'DBMS','CSE');
insert into course values(115,'SIGNALS','ECE');
commit;
select * from course;

insert into textbook values(10,'DATABASE SYSTEMS','PEARSON','SCHIELD');
insert into textbook values(900,'OPERATING SYSTEMS','PEARSON','LELAND');
insert into textbook values(901,'CIRCUITS','HALL INDIA','BOB');
insert into textbook values(902,'SYSTEM SOFTWARE','PETERSON','JACOB');
insert into textbook values(903,'SCHEDULING','PEARSON','PATIL');
insert into textbook values(904,'DATABASE SYSTEMS','PEARSON','JACOB');
insert into textbook values(905,'DATABASE MANAGER','PEARSON','BOB');
insert into textbook values(906,'SIGNALS','HALL INDIA','SUMIT');
commit;
select * from textbook;

insert into enroll values('1BM11CS001',115,3,100);
insert into enroll values('1BM11CS002',114,5,100);
insert into enroll values('1BM11CS003',113,5,100);
insert into enroll values('1BM11CS004',111,5,100);
insert into enroll values('1BM11CS005',112,3,100);
```

```
commit;

select * from enroll;


insert into book_adoption values(111,5,900);

insert into book_adoption values(111,5,903);

insert into book_adoption values(111,5,904);

insert into book_adoption values(112,3,901);

insert into book_adoption values(113,3,10);

insert into book_adoption values(114,5,905);

insert into book_adoption values(113,5,902);

insert into book_adoption values(115,3,906);

commit;

select * from book_adoption;

#Query1

insert into textbook values(908,'UNIX CONCEPTS','TATA MCGRAW
HILL','SUMITABHA DAS');

insert into book_adoption values(113,4,908);

select * from textbook;

select * from book_adoption;


#Query2

select c.courseno,t.book_isbn,t.book_title

from course c,book_adoption ba,textbook t

where c.courseno=ba.courseno

and ba.book_isbn=t.book_isbn

and c.dept='CSE'
```

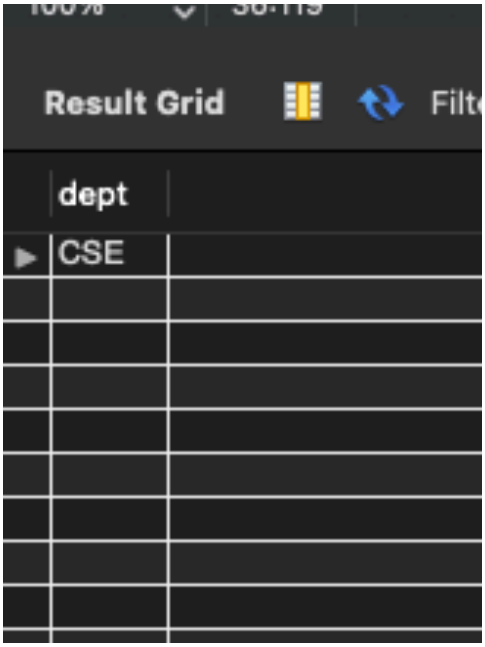
```
and 2<(select COUNT(book_isbn)
from book_adoption b
where c.courseno=b.courseno)
order by t.book_title;

#Query3
select distinct c.dept
from course c
where c.dept in(select c.dept
    from course c,book_adoption b,textbook t
    where c.courseno=b.courseno
    and t.book_isbn=b.book_isbn
    and t.publisher='PEARSON')
    and c.dept not in(select c.dept
    from course c,book_adoption b,textbook t
    where c.courseno=b.courseno
    and t.book_isbn=b.book_isbn
    and t.publisher != 'PEARSON');
```



Query1



Query2



Query3

Program 9: Movie database 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.
create database movie;

use movie;

```
create table actor(

act_id int,

act_name varchar(20),

act_gender char(1),

primary key(act_id));

desc actor;
```

```
create table director(
dir_id int,
dir_name varchar(20),
dir_phone int(10),
primary key(dir_id));
desc director;

alter table director
modify column dir_phone bigint;
desc director;

create table movies(
mov_id int,
mov_title varchar(25),
mov_year int,
mov_lang varchar(12),
dir_id int,
primary key(mov_id),
foreign key(dir_id) references director(dir_id));
desc movies;

create table movie_cast(
```

```
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));
desc movie_cast;

create table rating(
mov_id int,
rev_stars varchar(25),
primary key(mov_id),
foreign key(mov_id) references movies(mov_id));
desc rating;

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');
commit;
select * from actor;

insert into director values(60,'RAJAMOULI', 8751611001);
```

```
insert into director values(61,'HITCHCOCK', 7766138911);
insert into director values(62,'FARAN', 9986776531);
insert into director values(63,'STEVEN SPIELBERG', 8989776530);
commit;
select * from director;

insert into movies values(1001,'BAHUBALI-2', 2017, 'TELAGU', 60);
insert into movies values(1002,'BAHUBALI-1', 2015, 'TELAGU', 60);
insert into movies values(1003,'AKASH', 2008, 'KANNADA', 61);
insert into movies values(1004,'WAR HORSE', 2011, 'ENGLISH', 63);
commit;
select * from movies;

insert into movie_cast values(301, 1002, 'HEROINE');
insert into movie_cast values(301, 1001, 'HEROINE');
insert into movie_cast values(303, 1003, 'HERO');
insert into movie_cast values(303, 1002, 'GUEST');
insert into movie_cast values(304, 1004, 'HERO');
commit;
select * from movie_cast;

insert into rating values(1001, 4);
insert into rating values(1002, 2);
```

```
insert into rating values(1003, 5);

insert into rating values(1004, 4);

commit;

select * from rating;


#Query1

-- List the titles of all movies directed by 'Hitchcock'


select mov_title from movies m where dir_id=(select dir_id from director
where dir_name='Hitchcock');


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




#Query2

-- find the movie names where one or more actor acted in two or more
movies


select m.mov_title

from movies m, movie_cast mc

where m.mov_id=mc.mov_id

and mc.act_id in( select act_id from movie_cast group by act_id having
count(act_id)>1)

group by mov_title
```

```
having count(*)>1;


select m.mov_title from movies m,movie_cast mc where
m.mov_id=mc.mov_id

and mc.act_id in(select act_id from movie_cast group by act_id having
count(act_id)>1)

group by mov_title having count(*)>1;
```

#Query3

-- list all actors who acted in a movie before 2000 and also in a movie after 2015(use join operation)

```
select act_name,mov_title,mov_year from actor a join movie_cast mc on
a.act_id=mc.act_id join movies m on m.mov_id

=mc.mov_id where m.mov_year not between 2005 and 2015;
```

#Query4

-- find the titles of movies and number of stars for each movie that has at least one rating and find the highest number of

-- stars that movie recieved. sort the result by movie title

```
select mov_title,max(rev_stars)
from movies
inner join rating using(mov_id)
group by mov_id
having max(rev_stars)>0
order by mov_title;
```

```
select mov_title,max(rev_stars) from movies m,rating r
where m.mov_id=r.mov_id group by r.mov_id having max(rev_stars)>0
order by mov_title;
```

#Query5

-- update rating of all movies directed by 'Steven Spielberg' to 5kl

```
update rating set rev_stars=5
where mov_id in(select mov_id from movies where dir_id in(select dir_id
from director where dir_name='Steven Spielberg'));
select *from rating;
```

Query1

mov_title
AKASH

Query2

mov_title
BAHUBALI-1

Query3

act_name	mov_title	mov_year
ANUSHKA	BAHUBALI-2	2017

Query4

mov_title	max(rev_stars)
AKASH	5
BAHUBALI-1	2
BAHUBALI-2	4
WAR HORSE	5

Query5

mov_id	rev_stars
1001	4
1002	2
1003	5
1004	5
NULL	NULL

rating 41

Program 10

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. 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.

```
CREATE DATABASE COLLEGEDB;

USE COLLEGEDB;


CREATE TABLE STUDENT (

USN VARCHAR (10),

SNAME VARCHAR (25),

ADDRESS VARCHAR (25),

PHONE LONG,

GENDER CHAR (1),

PRIMARY KEY (USN));


select * from student;


CREATE TABLE SEMSEC (
```

```
SSID VARCHAR (5),
SEM INT,
SEC CHAR (1),
PRIMARY KEY (SSID));
select * from semsec;

CREATE TABLE CLASS (
USN VARCHAR (10),
SSID VARCHAR (5),
PRIMARY KEY (USN, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
select * from class;

CREATE TABLE SUBJECT (
SUBCODE VARCHAR (8),
TITLE VARCHAR (20),
SEM INT,
CREDITS INT,
PRIMARY KEY (SUBCODE));
select * from subject;

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),

FOREIGN KEY (SUBCODE) REFERENCES SUBJECT (SUBCODE),

FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));

select * from iamarks;


INSERT INTO STUDENT VALUES ('1RN13CS020','AKSHAY','BELAGAVI', 8877881122,'M');

INSERT INTO STUDENT VALUES ('1RN13CS062','SANDHYA','BENGALURU',
7722829912,'F');

INSERT INTO STUDENT VALUES ('1RN13CS091','TEESHA','BENGALURU', 7712312312,'F');

INSERT INTO STUDENT VALUES ('1RN13CS066','SUPRIYA','MANGALURU',
8877881122,'F');

INSERT INTO STUDENT VALUES ('1RN14CS010','ABHAY','BENGALURU', 9900211201,'M');

INSERT INTO STUDENT VALUES ('1RN14CS032','BHASKAR','BENGALURU',
9923211099,'M');

INSERT INTO STUDENT VALUES ('1RN14CS025','ASMI','BENGALURU', 7894737377,'F');

INSERT INTO STUDENT VALUES ('1RN15CS011','AJAY','TUMKUR', 9845091341,'M');



INSERT INTO STUDENT VALUES ('1RN15CS029','CHITRA','DAVANGERE', 7696772121,'F');

INSERT INTO STUDENT VALUES ('1RN15CS045','JEEVA','BELLARY', 9944850121,'M');

INSERT INTO STUDENT VALUES ('1RN15CS091','SANTOSH','MANGALURU',
8812332201,'M');

INSERT INTO STUDENT VALUES ('1RN16CS045','ISMAIL','KALBURGI', 9900232201,'M');

INSERT INTO STUDENT VALUES ('1RN16CS088','SAMEERA','SHIMOGA', 9905542212,'F');

INSERT INTO STUDENT VALUES ('1RN16CS122','VINAYAKA','CHIKAMAGALUR',
8800880011,'M');
```

```
INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO SEMSEC VALUES ('CSE8B', 8,'B');
INSERT INTO SEMSEC VALUES ('CSE8C', 8,'C');
INSERT INTO SEMSEC VALUES ('CSE7A', 7,'A');
INSERT INTO SEMSEC VALUES ('CSE7B', 7,'B');
INSERT INTO SEMSEC VALUES ('CSE7C', 7,'C');
INSERT INTO SEMSEC VALUES ('CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES ('CSE6B', 6,'B');
INSERT INTO SEMSEC VALUES ('CSE6C', 6,'C');
INSERT INTO SEMSEC VALUES ('CSE5A', 5,'A');
INSERT INTO SEMSEC VALUES ('CSE5B', 5,'B');
INSERT INTO SEMSEC VALUES ('CSE5C', 5,'C');
INSERT INTO SEMSEC VALUES ('CSE4A', 4,'A');
INSERT INTO SEMSEC VALUES ('CSE4B', 4,'B');
INSERT INTO SEMSEC VALUES ('CSE4C', 4,'C');
INSERT INTO SEMSEC VALUES ('CSE3A', 3,'A');
INSERT INTO SEMSEC VALUES ('CSE3B', 3,'B');
INSERT INTO SEMSEC VALUES ('CSE3C', 3,'C');
INSERT INTO SEMSEC VALUES ('CSE2A', 2,'A');
INSERT INTO SEMSEC VALUES ('CSE2B', 2,'B');
INSERT INTO SEMSEC VALUES ('CSE2C', 2,'C');
INSERT INTO SEMSEC VALUES ('CSE1A', 1,'A');
INSERT INTO SEMSEC VALUES ('CSE1B', 1,'B');
INSERT INTO SEMSEC VALUES ('CSE1C', 1,'C');

INSERT INTO CLASS VALUES ('1RN13CS020','CSE8A');
```

```
INSERT INTO CLASS VALUES ('1RN13CS062','CSE8A');
INSERT INTO CLASS VALUES ('1RN13CS066','CSE8B');
INSERT INTO CLASS VALUES ('1RN13CS091','CSE8C');
INSERT INTO CLASS VALUES ('1RN14CS010','CSE7A');
INSERT INTO CLASS VALUES ('1RN14CS025','CSE7A');
INSERT INTO CLASS VALUES ('1RN14CS032','CSE7A');
INSERT INTO CLASS VALUES ('1RN15CS011','CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS029','CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS045','CSE4B');
INSERT INTO CLASS VALUES ('1RN15CS091','CSE4C');
INSERT INTO CLASS VALUES ('1RN16CS045','CSE3A');
INSERT INTO CLASS VALUES ('1RN16CS088','CSE3B');
INSERT INTO CLASS VALUES ('1RN16CS122','CSE3C');


INSERT INTO SUBJECT VALUES ('10CS81','ACA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82','SSM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83','NM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84','CC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS85','PW', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS71','OOAD', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS72','ECS', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS73','PTW', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS74','DWDM', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS75','JAVA', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS76','SAN', 7, 4);
INSERT INTO SUBJECT VALUES ('15CS51', 'ME', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS52','CN', 5, 4);
```

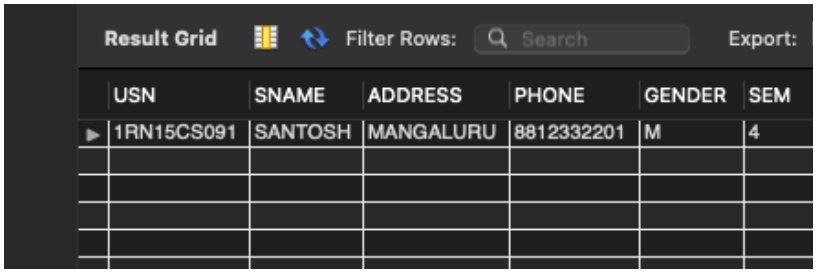
```
INSERT INTO SUBJECT VALUES ('15CS53','DBMS', 5, 4);

INSERT INTO SUBJECT VALUES ('15CS54','ATC', 5, 4);

INSERT INTO SUBJECT VALUES ('15CS55','JAVA', 5, 3);

INSERT INTO SUBJECT VALUES ('15CS56','AI', 5, 3);

INSERT INTO SUBJECT VALUES ('15CS41','M4', 4, 4);

INSERT INTO SUBJECT VALUES ('15CS42','SE', 4, 4);

INSERT INTO SUBJECT VALUES ('15CS43','DAA', 4, 4);

INSERT INTO SUBJECT VALUES ('15CS44','MPMC', 4, 4);

INSERT INTO SUBJECT VALUES ('15CS45','OOC', 4, 3);

INSERT INTO SUBJECT VALUES ('15CS46','DC', 4, 3);

INSERT INTO SUBJECT VALUES ('15CS31','M3', 3, 4);

INSERT INTO SUBJECT VALUES ('15CS32','ADE', 3, 4);

INSERT INTO SUBJECT VALUES ('15CS33','DSA', 3, 4);

INSERT INTO SUBJECT VALUES ('15CS34','CO', 3, 4);

INSERT INTO SUBJECT VALUES ('15CS35','USP', 3, 3);

INSERT INTO SUBJECT VALUES ('15CS36','DMS', 3, 3);


INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS81','CSE8C', 15, 16, 18);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS82','CSE8C', 12, 19, 14);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS83','CSE8C', 19, 15, 20);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS84','CSE8C', 20, 16, 19);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS85','CSE8C', 15, 15, 12);


/*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 SS.SEM, SS.SEC, 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 = '1RN13CS091';

SELECT * FROM STU_TEST1_MARKS_VIEW;

/*5. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = ‘Outstanding’
If FinalIA = 12 to 16 then CAT = ‘Average’
```

If FinalIA< 12 then CAT = ‘Weak’

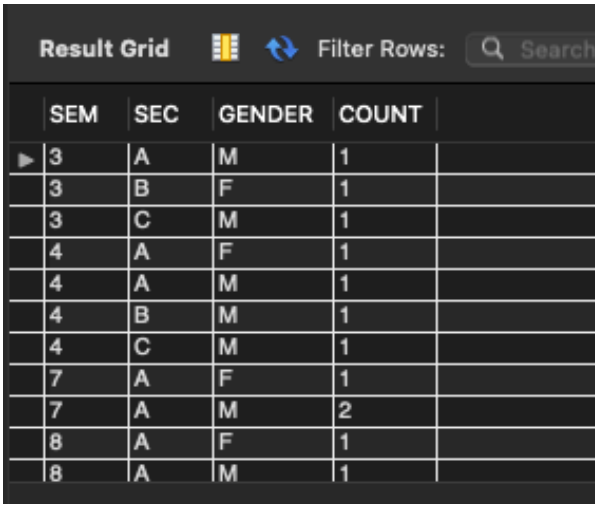
Give these details only for 8th semester A, B, and C section students. */

```
SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER,
(CASE
WHEN IA.FINALIA BETWEEN 17 AND 20 THEN 'OUTSTANDING'
WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
ELSE 'WEAK'
END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB
WHERE S.USN = IA.USN AND
SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE AND
SUB.SEM = 8;
```



USN	SNAME	ADDRESS	PHONE	GENDER	SEM
1RN15CS091	SANTOSH	MANGALURU	8812332201	M	4

query1



SEM	SEC	GENDER	COUNT
3	A	M	1
3	B	F	1
3	C	M	1
4	A	F	1
4	A	M	1
4	B	M	1
4	C	M	1
7	A	F	1
7	A	M	2
8	A	F	1
8	A	M	1

Query2

Result Grid Filter R

TEST1	SUBCODE
15	10CS81
12	10CS82
19	10CS83
20	10CS84
15	10CS85

Query3

Result Grid Filter Rows: Search Export:

USN	SNAME	ADDRESS	PHONE	GENDER	CAT
1RN13CS091	TEESHA	BENGALURU	7712312312	F	WEAK
1RN13CS091	TEESHA	BENGALURU	7712312312	F	WEAK
1RN13CS091	TEESHA	BENGALURU	7712312312	F	WEAK
1RN13CS091	TEESHA	BENGALURU	7712312312	F	WEAK
1RN13CS091	TEESHA	BENGALURU	7712312312	F	WEAK

Query4