ISHA SINGH
1BM19CS218
SECTION-4A

Question:

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 keys. 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 were involved in accidents in 2008. v)Find the number of accidents in which cars belonging to a specific model (example)were involved.

Person Table

driver_id	name	address
A01	Richard	Srinivas nagar
A02	Pradeep	Rajaji nagar
A03	Smith	Ashhok nagar
A04	Venu	N R Colony
A05	Jhon	Hanumanth nagar
NULL	NULL	NULL

Car Table

KA052250	Indica	1990
KA053408	Honda	2008
KA095477	Toyota	1998
NULL	NULL	NULL

Accident Table

report_num	accindent_date	location
11	2001-01-03	Mysore Road
12	2002-02-03	South and Circle
13	2021-01-03	Bull temple Road
14	2017-02-08	Mysore Road
15	2004-03-05	Kanakpura Road
16	2008-03-15	Domlur
NULL	NULL	NULL

Owns Table

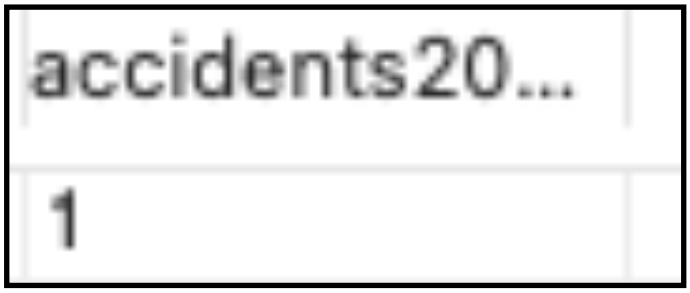
driver_id	reg_num
A03	KA031181
A05	KA041702
A01	KA052250
A02	KA053408
A04	KA095477
NULL	NULL

Participated Table

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

Find the total number of people who owned cars that involved in accidents in 2008.

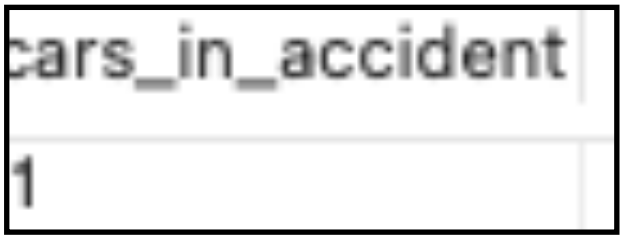
```
select count(distinct driver_id)
from Participated p,Accident a
where p.report_num=a.report_num AND a.accindent_date like '2008%';
```



accidents20...
1

Find the number of accidents in which cars belonging to a

```
select count(*)
from Participated p, Car c
where p.reg_num=c.reg_num and
c.model='Lancer';
```



cars_in_accident
1

Add a new accident to database- (16,2008-03-15,Domlur)

```
insert into
Accident(report_num,accindent_date,loc
ation)
values(16,'2008-03-15','Domlur');
select * from Accident;
```

report_num	accindent_date	location
11	2001-01-03	Mysore Road
12	2002-02-03	South and Circle
13	2021-01-03	Bull temple Road
14	2017-02-08	Mysore Road
15	2004-03-05	Kanakpura Road
16	2008-03-15	Domlur
NULL	NULL	NULL

update damage amount to 25000 for car with report_num 12 and

```
update Participated set
damage_amount=25000 where
report_num=12 and
reg_num="KA053408";
```

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

ISHA SINGH
1BM19CS218
SECTION-4A

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).

BRANCH TABLE

branch_name	branch_city	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000

LOAN TABLE

loan_number	branch_name	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_shivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000

BANK ACCOUNT TABLE

accno	branch_name	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000

CUSTOMER TABLE

customer_name	customer_street	customer_city
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi

DEPOSITOR TABLE

customer_name	accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11

Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).

```
SELECT customer_name

FROM depositor d,BankAccount a

 WHERE d.accno=a.accno

 AND a.branch_name='SBI_ResidencyRoad'

 GROUP BY d.customer_name

  HAVING COUNT(d.customer_name)>=2;
```

customer_name
Dinesh

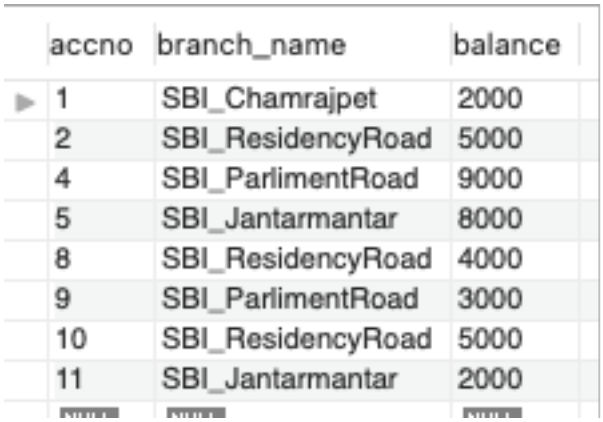
Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi)

```
select d.customer_name, b.branch_city,count(b.branch_name)

from Depositor d,BankAccount a, Branch b

where a.accno=d.accno and b.branch_name=a.branch_name and b.branch_city="Delhi"

group by d.customer_name

having count(distinct b.branch_name)=(select count(*)

from branch

where branch_city=“Delhi");
```

customer_name
Nikil

Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

DELETE FROM BankAccount WHERE branch_name IN(SELECT branch_name FROM BRANCH WHERE branch_city='Bombay');



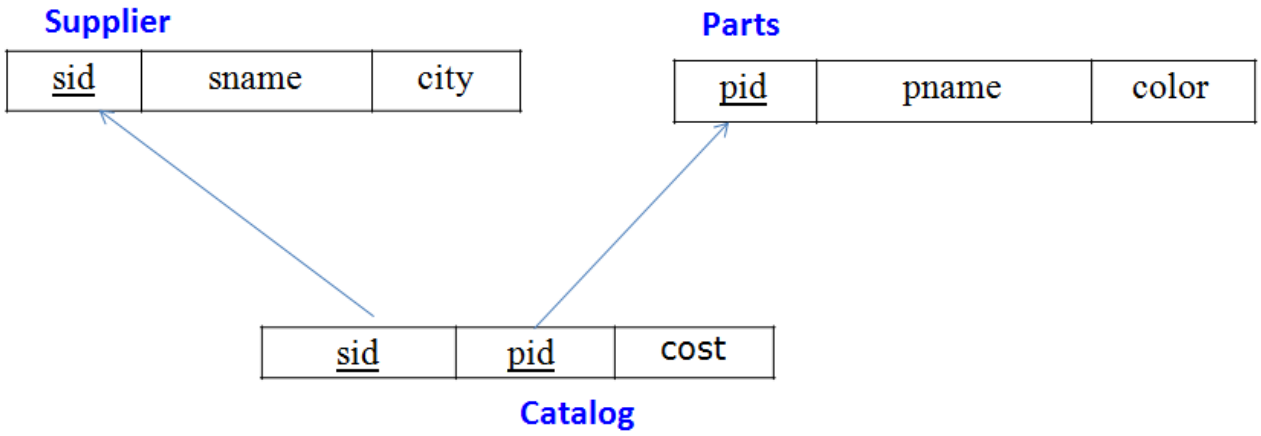
accno	branch_name	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000

ISHA SINGH
1BM19CS218
SECTION-4A

DBMS LAB-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.

Schema Diagram



SUPPLIER TABLE

sid	sname	city
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
10005	Mahindra	Mumbai
NULL	NULL	NULL

PARTS TABLE

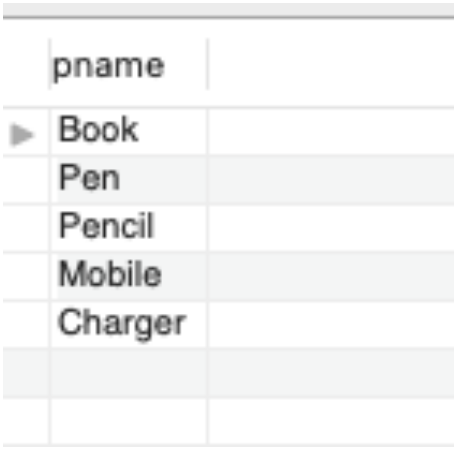
pid	pname	color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black
NULL	NULL	NULL

CATALOG TABLE

sid	pid	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

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

```
SELECT DISTINCT P.pname
FROM PARTS P,CATALOG C
WHERE P.pid=C.pid;
```



pname
Book
Pen
Pencil
Mobile
Charger

Find the snames of suppliers who supply every part

```
SELECT s.sname
from SUPPLIERS s,CATALOG c
where s.sid=c.sid
group by c.sid
having count(distinct c.pid)=(select count(*) from PARTS);
```

sname
Acme Widget

Find the snames of suppliers who supply every red part.

```
SELECT distinct s.sname
from SUPPLIERS s,CATALOG c
where s.sid=c.sid and c.pid in(select pid from PARTS where color='red');
```

sname
Acme Widget
Johns

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

```
select pname
from PARTS p, SUPPLIERS s1, CATALOG c1
where p.pid=c1.pid and s1.sid=c1.sid and s1.sname='Acme Widget' and
not exists
(select *
from CATAlOG c2,SUPPLIERS s2
where c1.pid=c2.pid and s2.sname<>'Acme Widget' and s2.sid=c2.sid);
```

pname
Mobile
Charger

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 sid
from CATALOG c1
where cost >
(select avg(cost)
from CATALOG c2
where c1.pid=c2.pid);
```

sid
10002
10004

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

```
select pid, sname
from CATALOG c1,SUPPLIERS S
where c1.sid=s.sid and cost =
(select max(cost)
from CATALOG c2
where c1.pid=c2.pid);
```

pid	sname
20001	Acme Widget
20004	Acme Widget
20005	Acme Widget
20001	Johns
20002	Johns
20003	Reliance

Isha Singh

1BM19CS218

SECTION-4A

Consider the following database for student enrollment for course :

STUDENT(snum: integer, sname:string, major: string, lvl: string, age: integer) CLASS(cname: string, meetsat: 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 "name"
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).

STUDENT TABLE

snum	sname	major	lvl	age
1	jhon	CS	Sr	19
2	Smith	CS	Jr	20
3	Jacob	CV	Sr	20
4	Tom	CS	Jr	20
5	Rahul	CS	Jr	20
6	Rita	CS	Sr	21
NULL	NULL	NULL	NULL	NULL

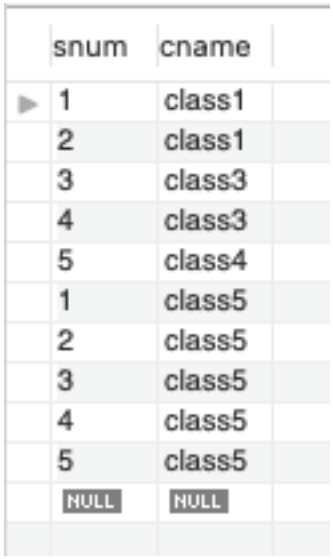
FACULTY TABLE

fid	fname	deptid
11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000
NULL	NULL	NULL

CLASS TABLE

cname	meets_at	room	fid
class1	2012-11-15 10:15:16	R1	14
class10	2012-11-15 10:15:16	R128	14
class2	2012-11-15 10:15:20	R2	12
class3	2012-11-15 10:15:25	R3	11
class4	2012-11-15 20:15:20	R4	14
class5	2012-11-15 20:15:20	R3	15
class6	2012-11-15 13:20:20	R2	14
class7	2012-11-15 10:10:10	R3	14
NULL	NULL	NULL	NULL

ENROLLED TABLE



snum	cname
1	class1
2	class1
3	class3
4	class3
5	class4
1	class5
2	class5
3	class5
4	class5
5	class5
NULL	NULL

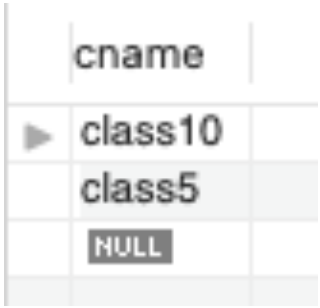
Find the names of all Juniors (level = JR) who are enrolled in a class taught by “name” select s.sname from Student s,Enrolled e,Class c

```
select sname
from student
where lvl='Jr' and snum in
(select snum
from enrolled e,class c,faculty f
where e.cname=c.cname and c.fid=f.fid and f.fname=‘Harish');
```

sname
Tom

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

```
select distinct c.cname

from class c

where c.room='R128' or cname in

(select cname

from enrolled

group by cname

having count(*)>=5);
```



cname
class10
class5
NULL

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

```
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.meets_at = C2.meets_at);
```

sname
Rahul

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

```
select f.fname
from faculty f,class c
 where c.fid=f.fid
 group by c.fid
 having count(c.fid)=
 (select count(distinct room)
 from class);
```

fname
Shiva

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

```
select fname
from faculty
where fid not in
(select f.fid
from faculty f,class c,enrolled e
where e.cname=c.cname and c.fid=f.fid
group by e.cname
having count(*)>=5);
```

fname
Harish
MV
Mira
Shiva

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

```
select sname
from student s
where s.snum not in
(select e.snum
from enrolled e);
```

sname
Rita

For each age value that appears in Students, find the level value that appears most often.

```
select s.age,s.lvl
from student s
group by s.age
having s.lvl in
(select s1.lvl
from student s1
where s1.age=s.age
group by s1.age
having count(*)>=all(select s2.lvl from student s2
where s2.age=s1.age
group by s2.age));
```

age	lvl
19	Sr
20	Jr
21	Sr

ISHA SINGH

1BM19CS218

SECTION-4A

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.

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

2. 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.

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

4. 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.

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

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

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

FLIGHTS TABLE

FLNO	FFROM	TTO	DISTANCE	DEPARTS	ARRIVES	PRICE
101	Bangalore	Delhi	2500	07:15:31	12:15:31	5000
102	Bangalore	Lucknow	3000	07:15:31	11:15:31	6000
103	Lucknow	Delhi	500	12:15:31	17:15:31	3000
104	Bangalore	Frankfurt	8500	07:15:31	23:15:31	75000
105	Kolkata	Delhi	3400	07:15:31	09:15:31	7000
106	Delhi	Kolkata	3400	12:15:35	14:20:00	7000
107	Bangalore	Frankfurt	8000	07:15:31	22:15:31	60000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

AIRCRAFT TABLE

AID	ANAME	CRUISINGRANGE
101	747	3000
102	Boeing	900
103	647	800
104	Dreamliner	10000
105	Boeing	3500
106	707	1500
107	Dream	12000
NULL	NULL	NULL

EMPLOYEES TABLE

EID	ENAME	SALARY
701	A	50000
702	B	100000
703	C	150000
704	D	90000
705	E	40000
706	F	60000
707	G	90000
NULL	NULL	NULL

CERITFIED TABLE

EID	AID
701	101
702	101
701	102
705	103
702	104
703	104
704	104
701	105
703	105
704	105
701	106
702	107
703	107
704	107
NULL	NULL

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

```
SELECT DISTINCT A.ANAME

FROM AIRCRAFT A,CERTIFIED C,EMPLOYEES E

WHERE A.AID=C.AID AND C.EID=E.EID AND SALARY>80000;
```

ANAME
747
Dreamliner
Dream
Boeing

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.

```
SELECT C.EID,MAX(A.CRUISINGRANGE)
FROM CERTIFIED C,AIRCRAFT A
WHERE C.AID=A.AID
GROUP BY C.EID
HAVING COUNT(*)>3;
```

EID	MAX(A.CRUISINGRANGE)
701	3500

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

```
SELECT ENAME
FROM EMPLOYEES
WHERE SALARY <(SELECT MIN(PRICE)
FROM FLIGHTS
WHERE FFROM="Bangalore" AND TTO=“Frankfurt");
```

ENAME
A
E

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.

```
SELECT A.ANAME,AVG(E.SALARY)
FROM CERTIFIED C,EMPLOYEES E,AIRCRAFT A
WHERE A.CRUISINGRANGE>1000 AND C.AID=A.AID AND C.EID=E.EID
GROUP BY A.ANAME;
```

ANAME	AVG(E.SALARY)
747	75000.0000
Dreamliner	113333.3333
Boeing	96666.6667
707	50000.0000
Dream	113333.3333

Find the names of pilots certified for some Boeing aircraft.

```
SELECT DISTINCT E.ENAME

FROM EMPLOYEES E,AIRCRAFT A,CERTIFIED C

WHERE E.EID=C.EID AND A.AID=C.AID AND A.ANAME=“Boeing";
```

ENAME
A
C
D

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

```
 SELECT A.AID
FROM AIRCRAFT A
WHERE A.CRUISINGRANGE>(SELECT MIN(DISTANCE) FROM FLIGHTS F WHERE
F.FFROM="Bangalore" AND F.TTO="Delhi");
```

AID
101
104
105
107
NULL

A customer wants to travel from bangalore to kolkata with no more than two changes of flight -- List the choice of departure times customer wants to arrive by 6 p.m.

```
select f.FFROM, f.TTO, f.ARRIVES from FlIGHTS f where (f.FFROM = "Bangalore" and f.TTO
                                        = (select FFROM from Flights where TTO = "Kolkata"))
                                          or f.TTO = “Kolkata";
```

FFROM	TTO	ARRIVES
Bangalore	Delhi	12:15:31
Delhi	Kolkata	14:20:00

ISHA SINGH

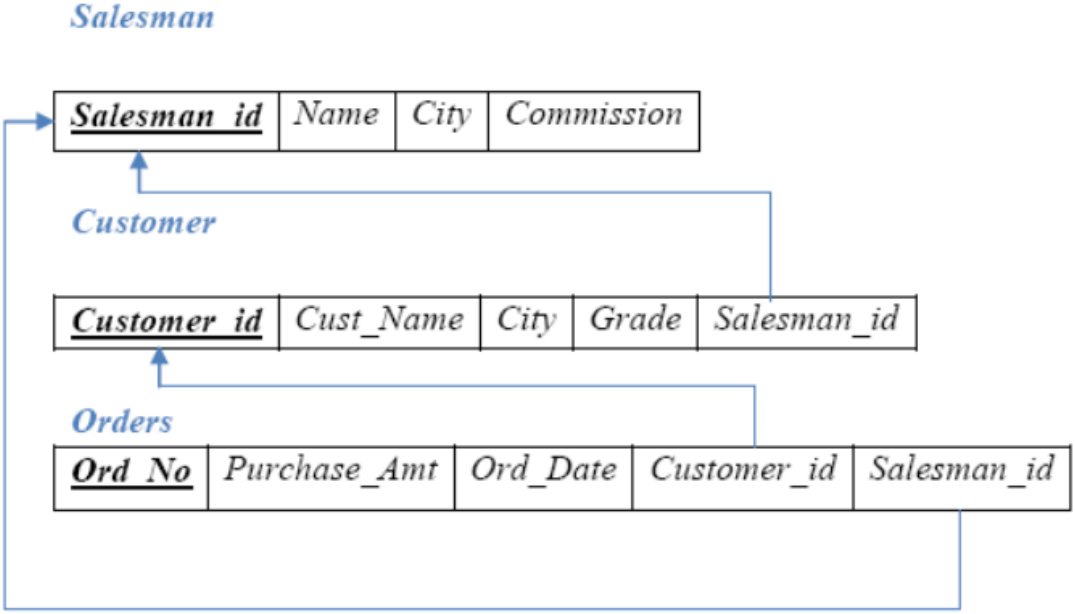
1BM19CS218

SECTION-4A

LAB-6

ORDER DATABASE

Schema Diagram




```
create database order_lab6;
use order_lab6;

create table salesman(
salesman_id int,
name varchar(30),
city varchar(20),
commission varchar(10),
primary key(salesman_id)
);
```

```
create table customer(
customer_id int,
cust_name varchar(30),
city varchar(20),
grade int,
salesman_id int,
primary key(customer_id),
foreign key (salesman_id) references salesman(salesman_id) on delete cascade
);

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

insert into salesman
values(1000,"John","Bangalore","25%"),(2000,"Ravi","Bangalore","20%"),
(3000,"Kumar","Mysore","15%"),(4000,"Smith","Delhi","30%"),(5000,"Harsha","Hydrabad","15%");

insert into customer
values(10,"Preethi","Bangalore",100,1000),(11,"Vivek","Mangalore",300,1000),
(12,"Bhaskar","Chennai",400,2000),(13,"Chethan","Bangalore",200,2000),
(14,"Mamatha","Bangalore",400,3000);

insert into orders
```

values(50,5000,"2017-05-04",10,1000),(51,450,"2017-01-20",10,2000),
(52,1000,"2017-02-24",13,2000),(53,3500,"2017-04-13",14,3000),(54,550,"2017-03-09",12,2000);

select * from salesman;

salesman_id	name	city	commission
1000	John	Bangalore	25%
2000	Ravi	Bangalore	20%
3000	Kumar	Mysore	15%
4000	Smith	Delhi	30%
5000	Harsha	Hydrabad	15%
NULL	NULL	NULL	NULL

select * from customer;

customer_id	cust_name	city	grade	salesman_id
10	Preethi	Bangalore	100	1000
11	Vivek	Mangalore	300	1000
12	Bhaskar	Chennai	400	2000
13	Chethan	Bangalore	200	2000
14	Mamatha	Bangalore	400	3000
NULL	NULL	NULL	NULL	NULL

select * from orders;

ord_no	purchase_amt	ord_date	customer_id	salesman_id
50	5000	2017-05-04	10	1000
51	450	2017-01-20	10	2000
52	1000	2017-02-24	13	2000
53	3500	2017-04-13	14	3000
54	550	2017-03-09	12	2000
NULL	NULL	NULL	NULL	NULL

QUERIES:

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

```
select count(distinct c.customer_id),grade
from customer c
where c.grade>(select avg(grade)
                    from customer c
          where city="Bangalore")
group by grade;
```

	count(distinct c.customer_id)	grade
▶	1	300
	2	400

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

```
select s.salesman_id,s.name
from salesman s,customer c
where c.salesman_id=s.salesman_id
group by c.salesman_id
having count(*)>1;
```

	salesman_id	name
▶	1000	John
	2000	Ravi

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

```
select s.salesman_id, name, cust_name, commission
FROM salesman s, customer c
WHERE s.CITY = c.CITY
UNION
SELECT SALESMAN_ID, NAME, "no match", COMMISSION
FROM SALESMAN
WHERE NOT CITY = ANY
(SELECT CITY
```

FROM CUSTOMER)

ORDER BY 2 DESC;

salesman_id	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%
1000	John	Preethi	25%
1000	John	Chethan	25%
1000	John	Mamatha	25%
5000	Harsha	no match	15%

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

```
create view salesman_highest
as
select o1.salesman_id,ord_date,name
from orders o1,salesman s
where o1.salesman_id=s.salesman_id and o1.salesman_id in(select salesman_id
                from orders o2
                where o1.ord_date=o2.ord_date and purchase_amt =(select max(purchase_amt)
                                                from orders o3
                                                where o3.ord_date=o2.ord_date));
select * from salesman_highest;
```

salesman_id	ord_date	name
1000	2017-05-04	John
2000	2017-01-20	Ravi
2000	2017-02-24	Ravi
3000	2017-04-13	Kumar
2000	2017-03-09	Ravi

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

DELETE FROM SALESMAN

WHERE SALESMAN_ID=1000;

select * from salesman;

salesman_id	name	city	commission
2000	Ravi	Bangalore	20%
3000	Kumar	Mysore	15%
4000	Smith	Delhi	30%
5000	Harsha	Hydrabad	15%
NULL	NULL	NULL	NULL

select * from customer;

customer_id	cust_name	city	grade	salesman_id
12	Bhaskar	Chennai	400	2000
13	Chethan	Bangalore	200	2000
14	Mamatha	Bangalore	400	3000
NULL	NULL	NULL	NULL	NULL

select * from orders;

ord_no	purchase_amt	ord_date	customer_id	salesman_id
52	1000	2017-02-24	13	2000
53	3500	2017-04-13	14	3000
54	550	2017-03-09	12	2000
NULL	NULL	NULL	NULL	NULL

ISHA SINGH
1BM19CS218
CSE-4A

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 ofpublisher, authors, number of copies in each branch, etc.
2. Get the particulars of borrowers who have borrowed more than 3books, but from Jan 2017 to Jun 2017
3. Delete a book in BOOK table. Update the contents of other tablesto 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 arecurrently available in the Library.

```
create database book;
use book;
create table publisher(
name varchar(20),
phone_no varchar(15),
address varchar(20),
primary key(name)
);

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

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

create table library_branch(
branch_id int,
branch_name varchar(50),
address varchar(50),
primary key(branch_id)
);

create table book_copies(
no_of_copies int,
book_id int,
branch_id 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
);

create table card(
card_no int,
primary key(card_no)
);

create table book_lending(
date_out date,
due_date date,
book_id int,
branch_id int,
card_no int,
primary key(book_id,branch_id,card_no),
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
);

insert into publisher
```

```
values("Mcgraw_Hill",9989076587,"Bangalore"),
("Pearson",9889076565,"New_Delhi"),
("Random_house",7455679345,"Hydrabad"),
("Hachette_Liver",8970862340,"Chennai"),
("Grupo_Planeta",7756120238,"Bangalore");

INSERT INTO book VALUES (1,"DBMS","JAN-2017",
"Mcgraw_Hill");
INSERT INTO book VALUES (2,"ADBMS","JUN-2016",
"Mcgraw_Hill");
INSERT INTO book VALUES (3,"CN","SEP-2016", "Pearson");
INSERT INTO book VALUES
(4,"CG","SEP-2015","Grupo_Planeta");
INSERT INTO book VALUES (5,"OS","MAY-2016", "Pearson");

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 ANGE", 4);
INSERT INTO book_authors VALUES ("GALVIN", 5);

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","Upupi");

INSERT INTO book_copies VALUES (10, 1, 10),
(5, 1, 11),
(2, 2, 12),
(5, 2, 13),
(7, 3, 14),
 (1, 5, 10),
(3, 4, 11);
truncate table 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);
```

```
INSERT INTO book_lending VALUES ("2017-01-01","2017-06-01",
1, 10, 101);
INSERT INTO book_lending VALUES ("2017-01-11","2017-03-11",
3, 14, 101);
INSERT INTO book_lending VALUES ("2017-02-21","2017-04-21",
2, 13, 101);
INSERT INTO book_lending VALUES ("2017-03-15","2017-07-15",
4, 11, 101);
INSERT INTO book_lending VALUES ("2017-04-12","2017-05-12",
1, 11, 104);

select * from book;
select * from book_authors;
select * from book_copies;
select * from book_lending;
select * from card;
select * from library_branch;
select * from publisher
```

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

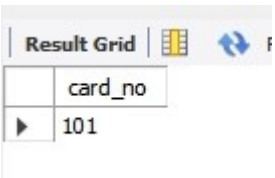
```
select
b.book_id,b.title,b.publisher_name,a.author_name,l.branch_id,c.no_of
_copies
from book b,book_authors a,book_copies c,library_branch
where b.book_id=a.book_id and b.book_id=c.book_id and
l.branch_id=c.branch_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

book_id	title	publisher_name	author_name	no_of_copies	branch_id
1	DBMS	MCGRAW-HILL	NAVATHE	10	10
1	DBMS	MCGRAW-HILL	NAVATHE	5	11
2	ADBMS	MCGRAW-HILL	NAVATHE	2	12
2	ADBMS	MCGRAW-HILL	NAVATHE	5	13
3	CN	PEARSON	TANENBAUM	7	14
4	CG	GRUPO PLANETA	EDWARD ANGEL	3	11
5	OS	PEARSON	GALVIN	1	10

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 b
where  date_out between "2017-01-01" and "2017-07-01"
group by card_no
having count(*)>3;
```



Result Grid

card_no
101

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

```
DELETE FROM
BOOKWHERE
BOOK_ID=3;

select * from book;
```

	BOOK_ID	TITLE	PUB_YEAR	PUBLISHER_NAME
▶	1	DBMS	JAN-2017	MCGRAW-HILL
	2	ADBMS	JUN-2016	MCGRAW-HILL
	4	CG	SEP-2015	GRUPO PLANETA
	5	OS	MAY-2016	PEARSON
*	NULL	NULL	NULL	NULL

select * from book_authors;

	AUTHOR_NAME	BOOK_ID
▶	NAVATHE	1
	NAVATHE	2
	EDWARD ANGEL	4
	GALVIN	5
*	NULL	NULL

select * from book_lending;

	DATE_OUT	DUE_DATE	BOOK_ID	BRANCH_ID	CARD_NO
▶	2017-01-01	2017-06-01	1	10	101
	2017-04-12	2017-05-12	1	11	104
	2017-02-21	2017-04-21	2	13	101
	2017-01-17	2017-03-17	3	14	101
	2017-03-15	2017-07-15	4	11	101
*	NULL	NULL	NULL	NULL	NULL

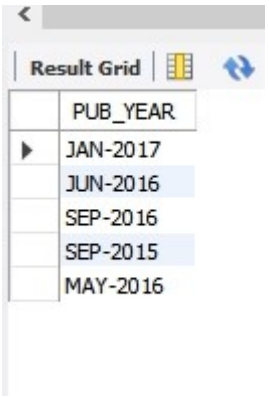
BOOK_LENDING 9 ×

select * from book_copies;

	NO_OF_COPIES	BOOK_ID	BRANCH_ID
▶	10	1	10
	5	1	11
	2	2	12
	5	2	13
	3	4	11
	1	5	10
*	NULL	NULL	NULL

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

```
create view publication as
select pub_year
from book;
select * from publication;
```

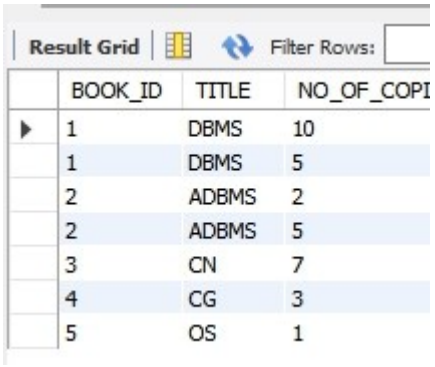


Result Grid

	PUB_YEAR
▶	JAN-2017
	JUN-2016
	SEP-2016
	SEP-2015
	MAY-2016

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

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



Result Grid | Filter Rows:

	BOOK_ID	TITLE	NO_OF_COPI
▶	1	DBMS	10
	1	DBMS	5
	2	ADBMS	2
	2	ADBMS	5
	3	CN	7
	4	CG	3
	5	OS	1

ISHA SINGH
1BM19CS218
SECTION-CSE-4A

PROGRAM 8. STUDENT ENROLLMENT DATABASE

Consider the following database of student enrollment in courses and books adopted for each course.

STUDENT (regno: String, name: String, major: String, bdate: date)
COURSE (course #: int, cname: String, dept: String)
ENROLL (regno: String, cname: String, 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.

```
CREATE DATABASE COLLEGE;

USE COLLEGE;


CREATE TABLE student(
    regno VARCHAR(15),
    sname VARCHAR(20),
    major VARCHAR(20),
    bdate DATE,
    PRIMARY KEY (regno) );


CREATE TABLE course(
    courseno INT,
    cname VARCHAR(20),
```

```
    dept VARCHAR(20),
    PRIMARY KEY (courseno) );
    select * from course;
CREATE TABLE enroll(
    regno VARCHAR(15),
    courseno INT,
    sem INT(3),
    marks INT(4),
    PRIMARY KEY (regno,courseno),
    FOREIGN KEY (regno) REFERENCES student (regno),
    FOREIGN KEY (courseno) REFERENCES course (courseno) );

CREATE TABLE text(
    book_isbn INT(5),
    book_title VARCHAR(20),
    publisher VARCHAR(20),
    author VARCHAR(20),
    PRIMARY KEY (book_isbn) );

CREATE TABLE book_adoption(
    courseno INT,
    sem INT(3),
    book_isbn INT(5),
    PRIMARY KEY (courseno,book_isbn),
    FOREIGN KEY (courseno) REFERENCES course (courseno),
    FOREIGN KEY (book_isbn) REFERENCES text(book_isbn) );


INSERT INTO student (regno,sname,major,bdate) VALUES
```

```
    ('1pe11cs002','b','sr','19930924'),
    ('1pe11cs003','c','sr','19931127'),
    ('1pe11cs004','d','sr','19930413'),
    ('1pe11cs005','e','jr','19940824');
 INSERT INTO student (regno,sname,major,bdate) VALUES
    ('1pe11cs001','a','jr','19930922');
select * from student;

INSERT INTO course VALUES (111,'OS','CSE'),
    (112,'EC','CSE'),
    (113,'SS','ISE'),
    (114,'DBMS','CSE'),
    (115,'SIGNALS','ECE');

INSERT INTO text VALUES
    (10,'DATABASE SYSTEMS','PEARSON','SCHIELD'),
    (900,'OPERATING SYS','PEARSON','LELAND'),
    (901,'CIRCUITS','HALL INDIA','BOB'),
    (902,'SYSTEM SOFTWARE','PETERSON','JACOB'),
    (903,'SCHEDULING','PEARSON','PATIL'),
    (904,'DATABASE SYSTEMS','PEARSON','JACOB'),
    (905,'DATABASE MANAGER','PEARSON','BOB'),
    (906,'SIGNALS','HALL INDIA','SUMIT');


INSERT INTO enroll (regno,courseno,sem,marks) VALUES ('1pe11cs001',115,3,100),
    ('1pe11cs002',114,5,100),
    ('1pe11cs003',113,5,100),
    ('1pe11cs004',111,5,100),
```

('1pe11cs005',112,3,100);

INSERT INTO book_adoption (courseno,sem,book_isbn) VALUES

(111,5,900),

(111,5,903),

(111,5,904),

(112,3,901),

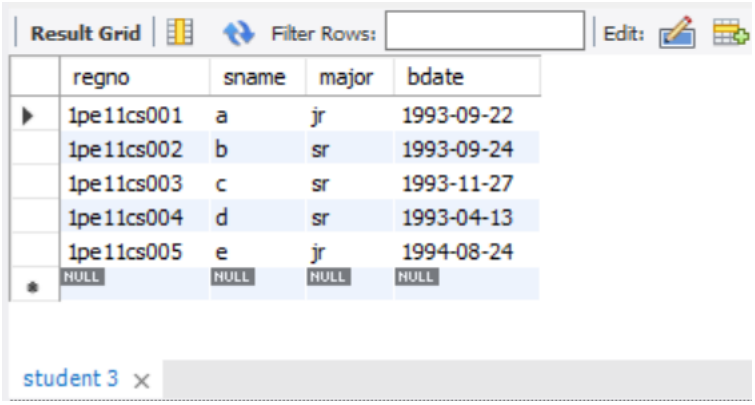
(113,3,10),

(114,5,905),

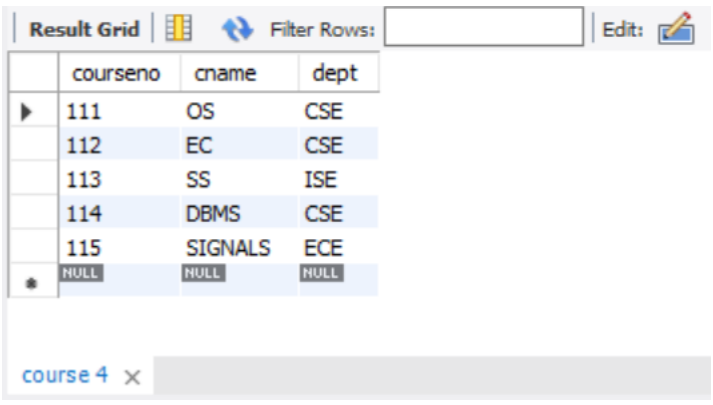
(113,5,902),

(115,3,906);

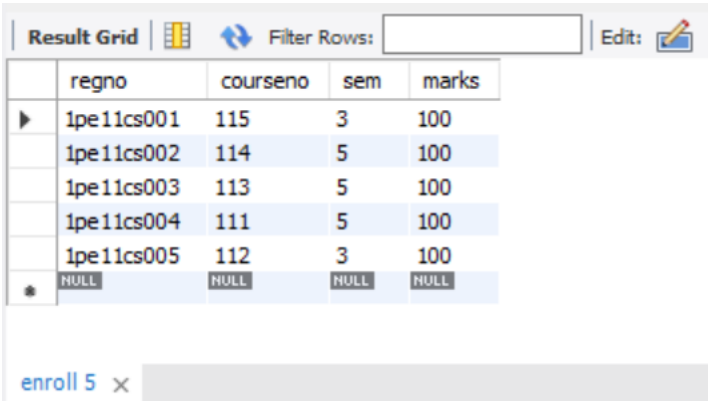
select * from student;



select * from course;



select * from enroll;

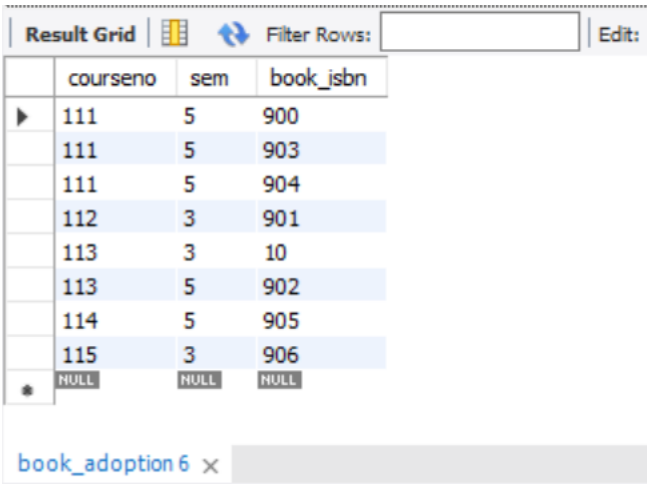


Result Grid | Filter Rows: | Edit:

regno	courseno	sem	marks
1pe11cs001	115	3	100
1pe11cs002	114	5	100
1pe11cs003	113	5	100
1pe11cs004	111	5	100
1pe11cs005	112	3	100
NULL	NULL	NULL	NULL

enroll 5 ×

select * from book_adoption;

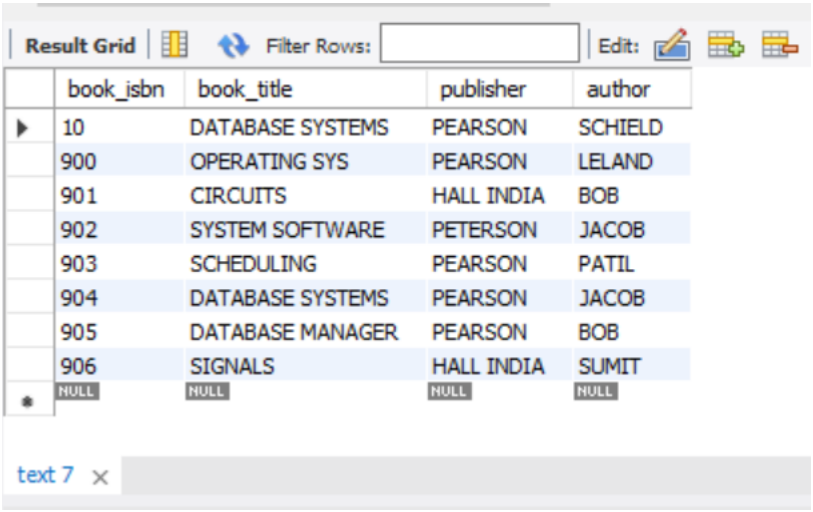


Result Grid | Filter Rows: | Edit:

courseno	sem	book_isbn
111	5	900
111	5	903
111	5	904
112	3	901
113	3	10
113	5	902
114	5	905
115	3	906
NULL	NULL	NULL

book_adoption 6 ×

select * from text;



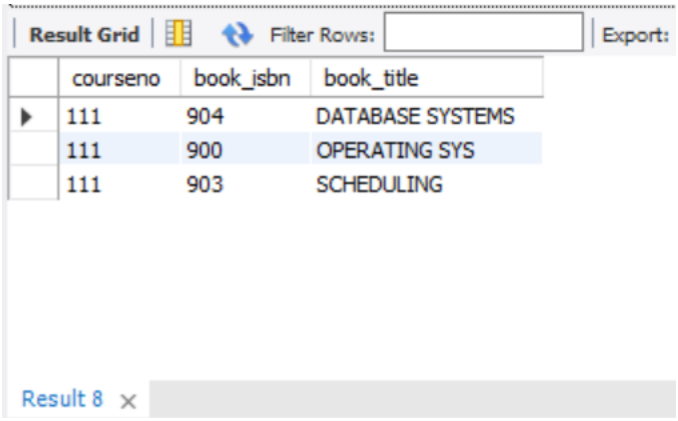
Result Grid | Filter Rows: | Edit:

book_isbn	book_title	publisher	author
10	DATABASE SYSTEMS	PEARSON	SCHIELD
900	OPERATING SYS	PEARSON	LELAND
901	CIRCUITS	HALL INDIA	BOB
902	SYSTEM SOFTWARE	PETERSON	JACOB
903	SCHEDULING	PEARSON	PATIL
904	DATABASE SYSTEMS	PEARSON	JACOB
905	DATABASE MANAGER	PEARSON	BOB
906	SIGNALS	HALL INDIA	SUMIT
NULL	NULL	NULL	NULL

text 7 ×

4. 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.

```
SELECT c.courseno,t.book_isbn,t.book_title
FROM course c,book_adoption ba,text 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;
```

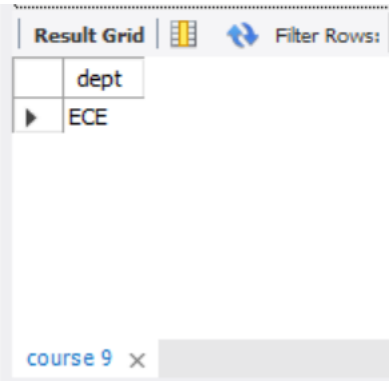


5.List any department that has all its adopted books published by a specific publisher.

```
select c.dept
from course c, book_adoption ba
where c.courseno=ba.courseno
group by c.dept
having count(ba.book_isbn)=(select count(ba2.book_isbn)
```

from text t,book_adoption ba2,course c2

where t.book_isbn=ba2.book_isbn and c2.courseno=ba2.courseno and t.publisher='HALL INDIA' and c2.dept=c.dept);



ISHA SINGH
1BM19CS218
SECTION-CSE-4A

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
i. List the titles of all movies directed by 'Hitchcock'.
ii. Find the movie names where one or more actors acted in two or more movies.
iii. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
iv. 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.
v. 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));


CREATE TABLE DIRECTOR (

DIR_ID INT,

DIR_NAME VARCHAR (20),
```

```
DIR_PHONE LONG,
PRIMARY KEY (DIR_ID));

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));

CREATE TABLE MOVIE_CAST (
ACT_ID INT,
MOV_ID INT,
AROLE VARCHAR(10),
PRIMARY KEY (ACT_ID, MOV_ID),
FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID) ON DELETE CASCADE,
FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON DELETE
CASCADE);

CREATE TABLE RATING (
MOV_ID INT,
REV_STARS VARCHAR (25),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));

INSERT INTO ACTOR VALUES (301,'ANUSHKA','F');
INSERT INTO ACTOR VALUES (302,'PRABHAS','M');
```

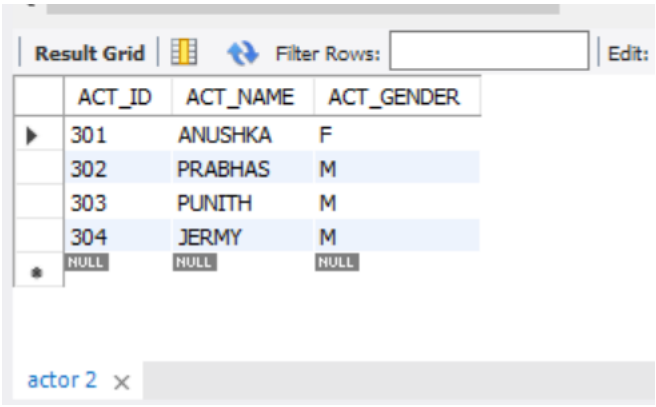
```
INSERT INTO ACTOR VALUES (303,'PUNITH','M');
INSERT INTO ACTOR VALUES (304,'JERMY','M');

INSERT INTO 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);

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);

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');

INSERT INTO RATING VALUES (1001, 4);
INSERT INTO RATING VALUES (1002, 2);
INSERT INTO RATING VALUES (1003, 5);
INSERT INTO RATING VALUES (1004, 4);

select * from rating;
```

Result Grid | Filter Rows:

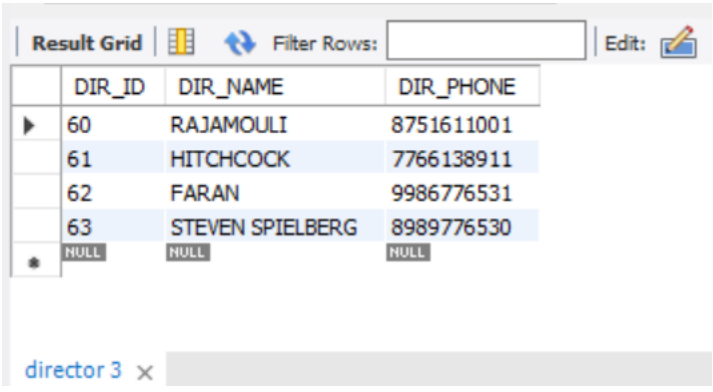
MOV_ID	REV_STARS
1001	4
1002	2
1003	5
1004	4
NULL	NULL

rating 1

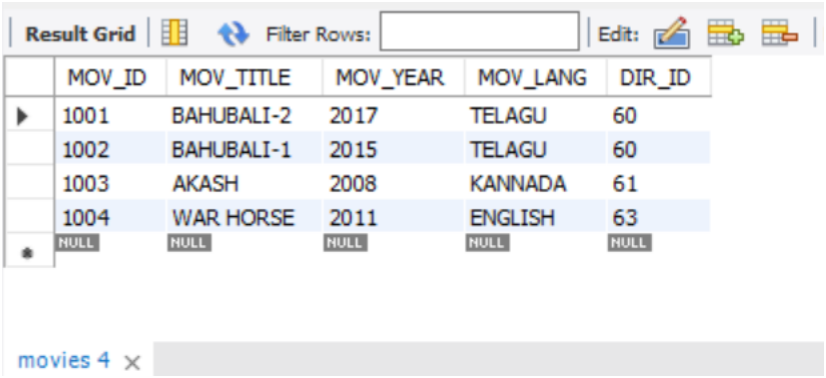
select * from actor;



select * from director;



select * from movies;

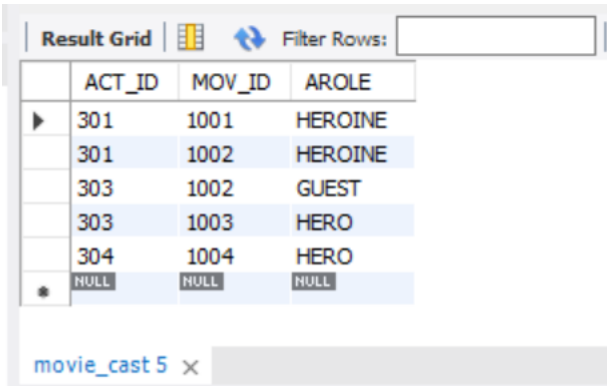


Result Grid | Filter Rows: | Edit:

MOV_ID	MOV_TITLE	MOV_YEAR	MOV_LANG	DIR_ID
1001	BAHUBALI-2	2017	TELAGU	60
1002	BAHUBALI-1	2015	TELAGU	60
1003	AKASH	2008	KANNADA	61
1004	WAR HORSE	2011	ENGLISH	63
NULL	NULL	NULL	NULL	NULL

movies 4

select * from movie_cast;



Result Grid | Filter Rows:

ACT_ID	MOV_ID	AROLE
301	1001	HEROINE
301	1002	HEROINE
303	1002	GUEST
303	1003	HERO
304	1004	HERO
NULL	NULL	NULL

movie_cast 5

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 dir_name="Hitchcock";

Result Grid | Filter Rows:

MOV_TITLE
AKASH

MOVIES 6

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

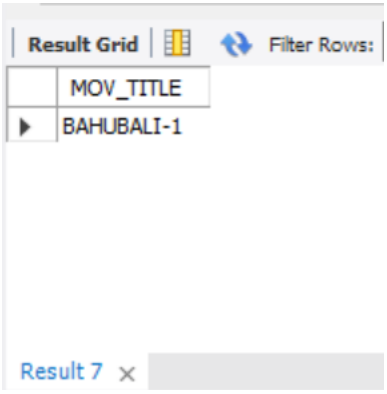
select m.mov_title

from movies m,movie_cast c

where m.mov_id=c.mov_id and act_id in(select act_id from movie_cast group by act_id having count(act_id)>1)

group by m.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

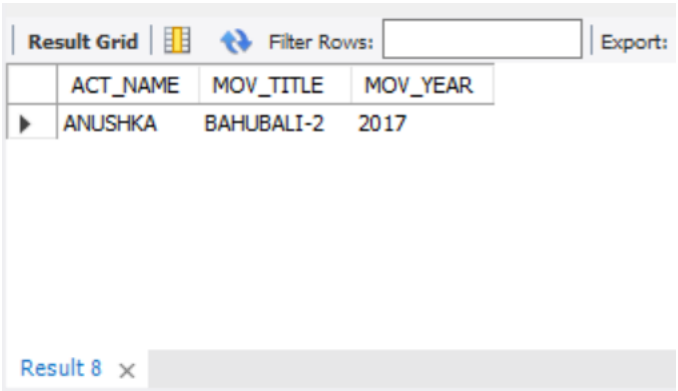
from actor

where act_id =(

select c.act_id

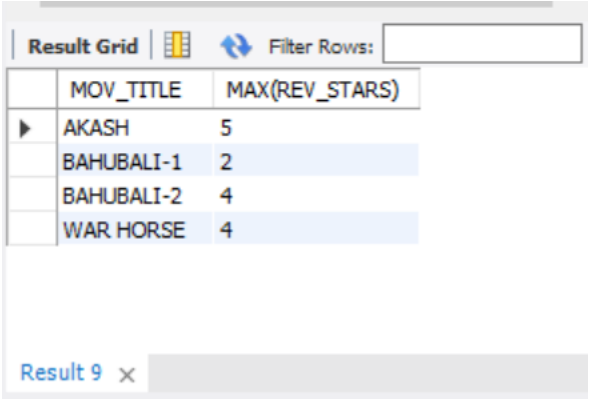
from movie_cast c,movies m

where c.mov_id=m.mov_id and m.mov_year not between 2000 and 2015);



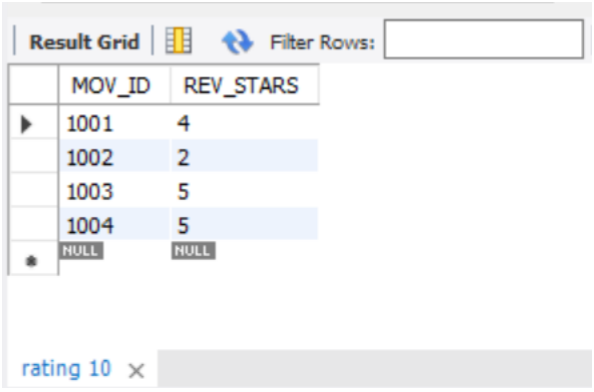
4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received.

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



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

```
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;
```


Result Grid
Filter Rows:
MOV_ID
REV_STARS
1001
4
1002
2
1003
5
1004
5
NULL
NULL
rating 10

ISHA SINGH
1BM19CS218
SECTION-CSE-4A

PROGRAM 10:COLLEGE DATABASE

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
i. List all the student details studying in fourth semester 'C' section.
ii. Compute the total number of male and female students in each semester and in each section.
iii. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.
iv. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.
v. 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.

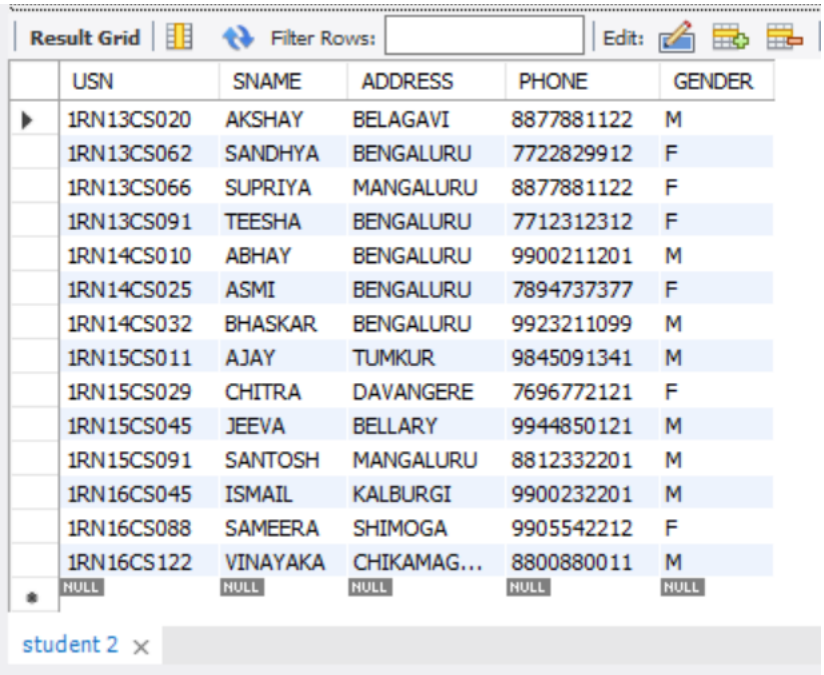
```
CREATE DATABASE COLLEGEDB;
USE COLLEGEDB;

CREATE TABLE STUDENT (
USN VARCHAR (10),
SNAME VARCHAR (25),
ADDRESS VARCHAR (25),
PHONE LONG,
GENDER CHAR (1),
PRIMARY KEY (USN));

CREATE TABLE SEMSEC (
SSID VARCHAR (5),
SEM INT,
SEC CHAR (1),
PRIMARY KEY (SSID));
```

```
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));

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

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));

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);
```

select * from student;



Result Grid | Filter Rows: | Edit:

USN	SNAME	ADDRESS	PHONE	GENDER
1RN13CS020	AKSHAY	BELAGAVI	8877881122	M
1RN13CS062	SANDHYA	BENGALURU	7722829912	F
1RN13CS066	SUPRIYA	MANGALURU	8877881122	F
1RN13CS091	TEESHA	BENGALURU	7712312312	F
1RN14CS010	ABHAY	BENGALURU	9900211201	M
1RN14CS025	ASMI	BENGALURU	7894737377	F
1RN14CS032	BHASKAR	BENGALURU	9923211099	M
1RN15CS011	AJAY	TUMKUR	9845091341	M
1RN15CS029	CHITRA	DAVANGERE	7696772121	F
1RN15CS045	JEEVA	BELLARY	9944850121	M
1RN15CS091	SANTOSH	MANGALURU	8812332201	M
1RN16CS045	ISMAIL	KALBURGI	9900232201	M
1RN16CS088	SAMEERA	SHIMOGA	9905542212	F
1RN16CS122	VINAYAKA	CHIKAMAG...	8800880011	M
NULL	NULL	NULL	NULL	NULL

student 2

select * from semsec;

SSID	SEM	SEC
CSE1A	1	A
CSE1B	1	B
CSE1C	1	C
CSE2A	2	A
CSE2B	2	B
CSE2C	2	C
CSE3A	3	A
CSE3B	3	B
CSE3C	3	C
CSE4A	4	A
CSE4B	4	B
CSE4C	4	C
CSE5A	5	A
CSE5B	5	B
CSE5C	5	C
CSE6A	6	A
CSE6B	6	B

semsec 3 ×

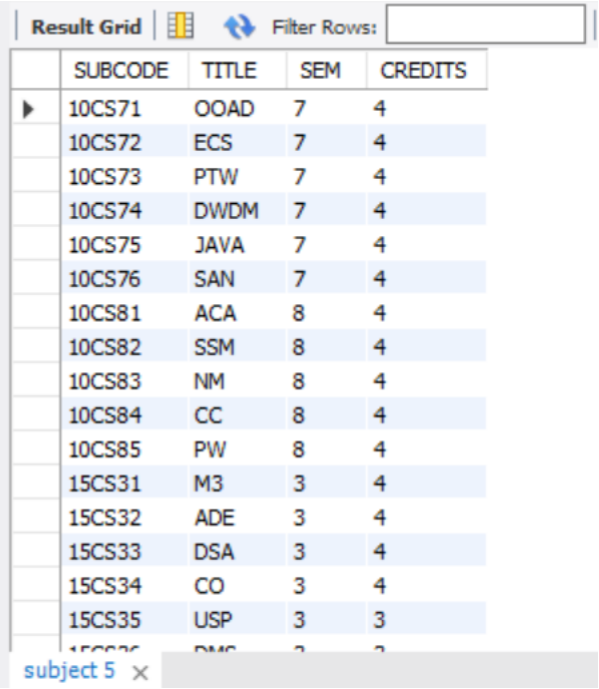
select * from class;

Result Grid | Filter Rows:

USN	SSID
1RN16CS045	CSE3A
1RN16CS088	CSE3B
1RN16CS122	CSE3C
1RN15CS011	CSE4A
1RN15CS029	CSE4A
1RN15CS045	CSE4B
1RN15CS091	CSE4C
1RN14CS010	CSE7A
1RN14CS025	CSE7A
1RN14CS032	CSE7A
1RN13CS020	CSE8A
1RN13CS062	CSE8A
1RN13CS066	CSE8B
1RN13CS091	CSE8C
NULL	NULL

class 4 ×

select * from subject;

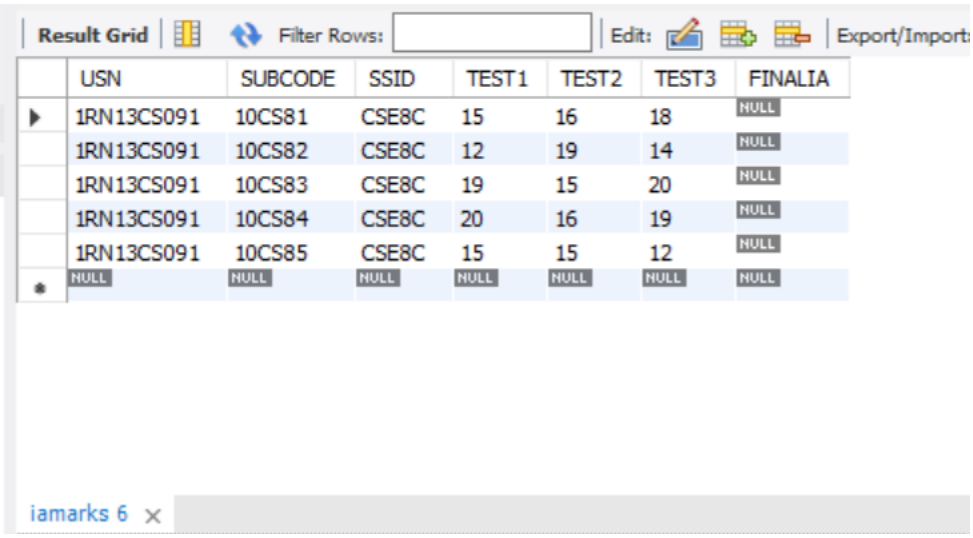


Result Grid | Filter Rows:

SUBCODE	TITLE	SEM	CREDITS
10CS71	OOAD	7	4
10CS72	ECS	7	4
10CS73	PTW	7	4
10CS74	DWDM	7	4
10CS75	JAVA	7	4
10CS76	SAN	7	4
10CS81	ACA	8	4
10CS82	SSM	8	4
10CS83	NM	8	4
10CS84	CC	8	4
10CS85	PW	8	4
15CS31	M3	3	4
15CS32	ADE	3	4
15CS33	DSA	3	4
15CS34	CO	3	4
15CS35	USP	3	3

subject 5

select * from iamarks;



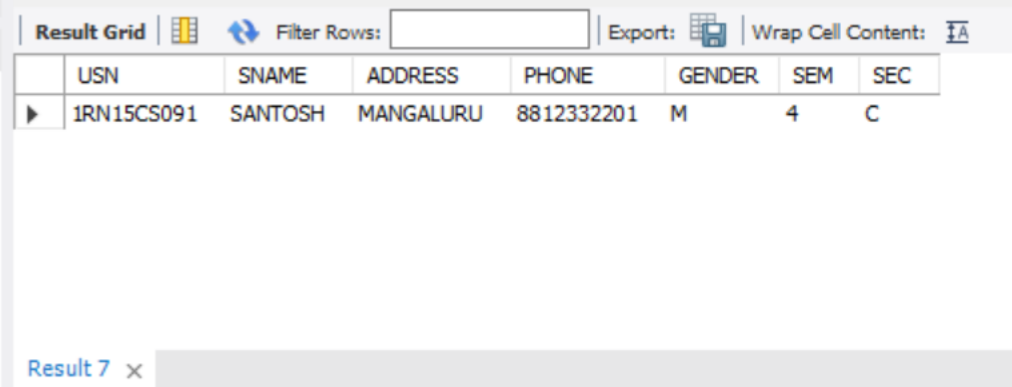
Result Grid | Filter Rows: | Edit: | Export/Import:

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
1RN13CS091	10CS81	CSE8C	15	16	18	NULL
1RN13CS091	10CS82	CSE8C	12	19	14	NULL
1RN13CS091	10CS83	CSE8C	19	15	20	NULL
1RN13CS091	10CS84	CSE8C	20	16	19	NULL
1RN13CS091	10CS85	CSE8C	15	15	12	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

iamarks 6

1. List all the student details studying in fourth semester ‘C’ section.

```
select s.*,sm.sem,sm.sec
from student s,semsec sm,class c
where sm.ssid=c.ssid and s.usn=c.usn and sm.sem=4 and sm.sec="C";
```

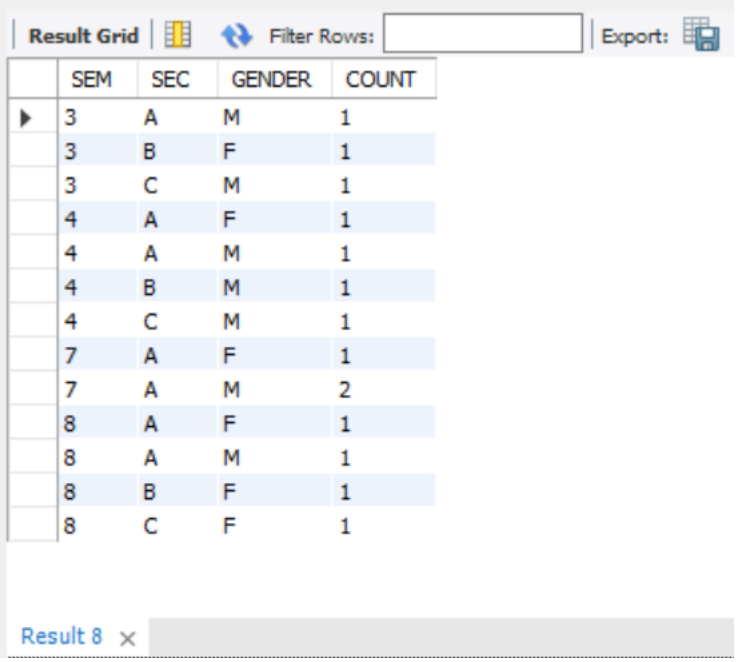

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 7

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

```
select sm.sem,sm.sec,s.gender,count(s.gender)
from student s,semsec sm,class c
where sm.ssid=c.ssid and s.usn=c.usn and s.gender="M"
group by sm.sem,sm.sec
UNION
select sm.sem,sm.sec,s.gender,count(s.gender)
from student s,semsec sm,class c
where sm.ssid=c.ssid and s.usn=c.usn and s.gender="F"
group by sm.sem,sm.sec;
```



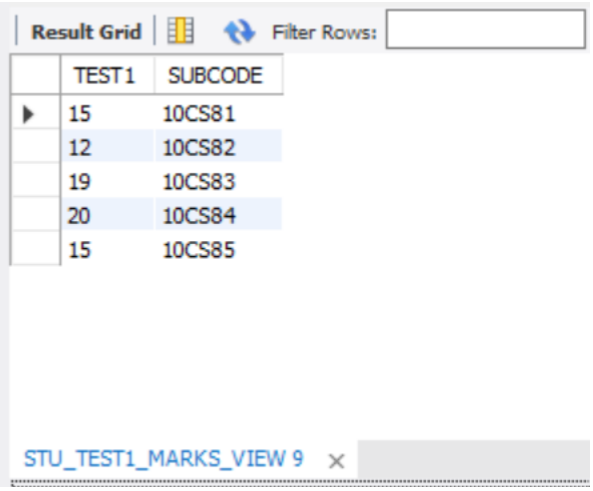
Result Grid | Filter Rows: | Export:

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
8	B	F	1
8	C	F	1

Result 8 ×

/*3. Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects. */
create view Marks (subcode,test1_marks) as
select s.subcode,m.test1
from iamarks m,subject s
where m.subcode=s.subcode and m.usn="1RN13CS091";

select * from Marks;



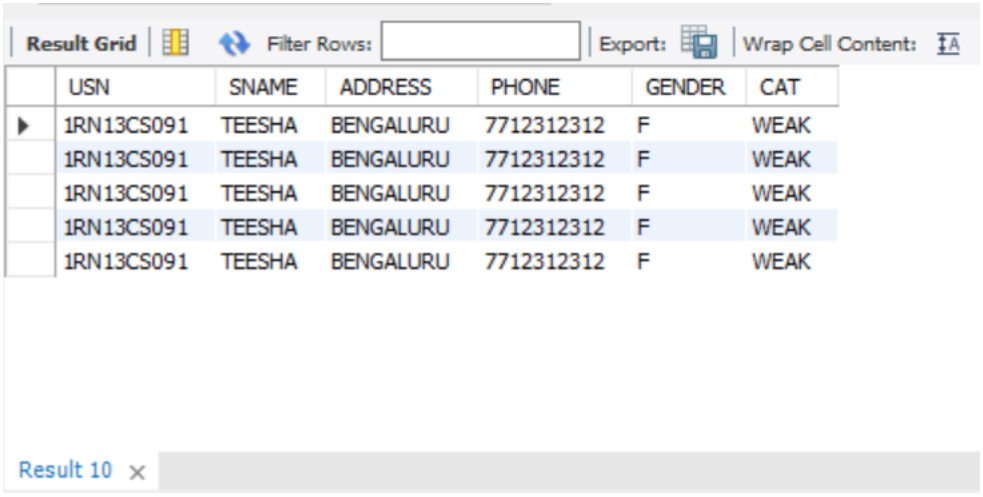
Result Grid | Filter Rows:

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

STU_TEST1_MARKS_VIEW 9 ×

/*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	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