VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

DATA BASE MANAGEMENT SYSTEM LAB REPORT PHASE 1

(19CS4PCDBM)

Submitted by

RAVI SAJJANAR (1BM19CS127)

Under the Guidance of

Prof. Sheetal V A
Assistant Professor, BMSCE

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Mar-2021 to Jun-2022

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab Assignment work entitled "**Database Management System** " carried out by **RAVI SAJJANAR (1BM19CS127)** who is the bonafide students of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visveswaraiah Technological University, Belgaum during the year 2021-2022. The Lab report has been approved as it satisfies the academic requirements in respect of **Database Management System (19CS4PCDBM) LAB** work prescribed for the said degree.

Signature of the Guide
Prof. Prof. Sheetal VA
Assistant Professor
BMSCE, Bengaluru

Signature of the HOD
Dr. Umadevi V
Associate Prof.& Head, Dept. of CSE
BMSCE, Bengaluru

External Viva

Name of the Examiner | Signature with date

1._____ _____

2. _____ _____

INDEX

PROGRAM 1: INSURANCE DATABASE

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

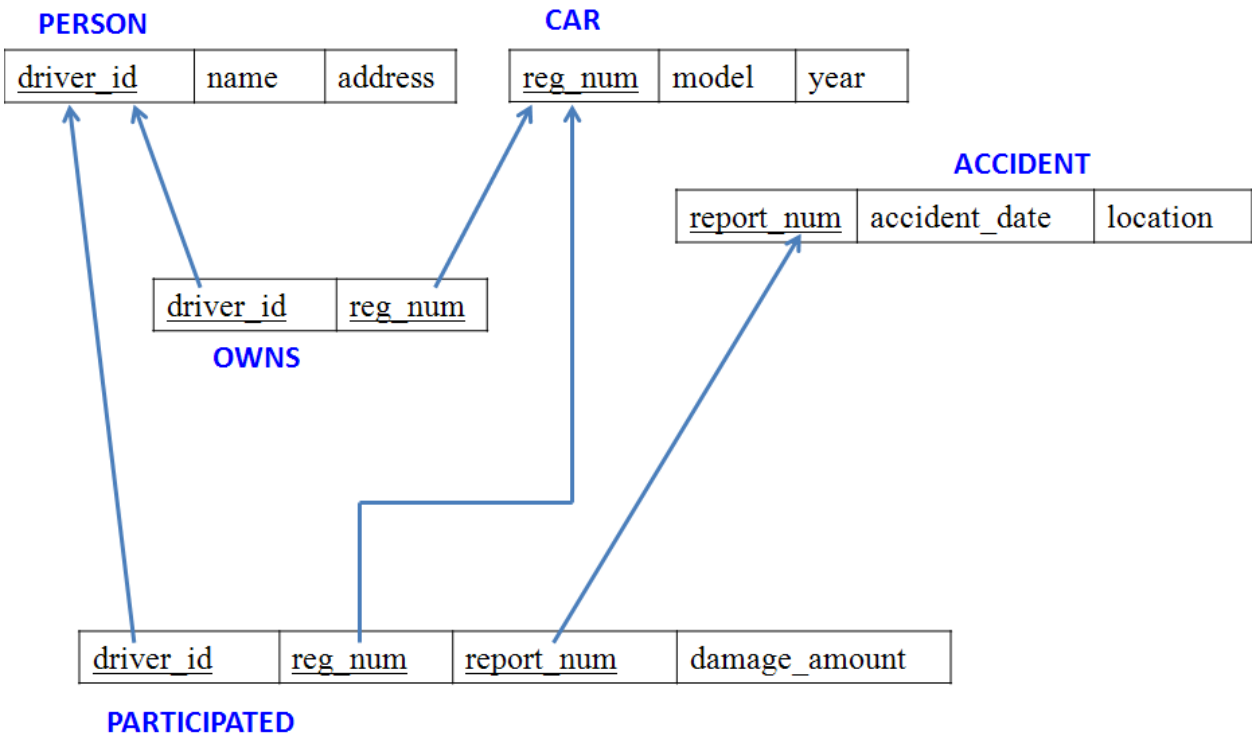
PERSON (driver_id: String, name: String, address: String)
CAR (reg_num: String, model: String, year: int)
ACCIDENT (report_num: int, accident_date: date, location: String)
OWNS (driver_id: String, reg_num: String)
PARTICIPATED (driver_id: String, reg_num: String, report_num: int, damage_amount: int)

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

Schema Diagram:




```
-- Creating the database and the tables

create database insurance;
use insurance;

create table person(
        driver_id varchar(10),
        name varchar(20),
        address varchar(30),
        primary key(driver_id)
);

desc person;
```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
address	varchar(30)	YES		NULL	

```
create table car(
        reg_num varchar(10),
        model varchar(10),
        year int,
        primary key(reg_num)
);

desc car;
```

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

Field	Type	Null	Key	Default	Extra
reg_num	varchar(10)	NO	PRI	NULL	
model	varchar(10)	YES		NULL	
year	int	YES		NULL	

```
create table accident(
        report_num int,
        accident_date date,
        location varchar(20),
        primary key(report_num)
);
```

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

Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(20)	YES		NULL	

```
create table 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)
);

desc owns;
```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	

```
create table 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)
);

desc participated;
```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

```
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');

commit;

select * from person;
```

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

driver_id	name	address
A01	Richard	Srinivas Nagar
A02	Pradeep	Rajajinagar
A03	Smith	Ashoknagar
A04	Venu	N.R.Colony
A05	John	Hanumanth Nagar
NULL	NULL	NULL

```
insert into car values('KA031181','Lancer',1957);
insert into car values('KA041702','Audi',2005);
insert into car values('KA043408','Honda',2008);
insert into car values('KA052250','Indica',1990);
insert into car values('KA095477','Toyota',1998);

commit;
select * from car;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Co

reg_num	model	year
KA031181	Lancer	1957
KA041702	Audi	2005
KA043408	Honda	2008
KA052250	Indica	1990
KA095477	Toyota	1998
NULL	NULL	NULL

```
insert into accident values(11,'2001-01-03','Mysore Road');
insert into accident values(12,'2021-01-03','Southend Circle');
insert into accident values(13,'2020-03-03',' Bulltemple Road');
insert into accident values(14,' 2017-02-08',' Mysore Road');
insert into accident values(15,'2004-03-05','Kanakpura Road');
commit;

select * from accident;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

report_num	accident_date	location
11	2001-01-03	Mysore Road
12	2021-01-03	Southend Circle
13	2020-03-03	Bulltemple Road
14	2017-02-08	Mysore Road
15	2004-03-05	Kanakpura Road
NULL	NULL	NULL

```
insert into owns values ('A01','KA052250');
insert into owns values ('A02','KA043408');
insert into owns values ('A03','KA031181');
insert into owns values ('A04','KA095477');
insert into owns values ('A05','KA041702');
commit;
select * from owns;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

driver_id	reg_num
A03	KA031181
A05	KA041702
A02	KA043408
A01	KA052250
A04	KA095477
NULL	NULL

```
insert into participated values ('A01','KA052250',11, 25000);
insert into participated values ('A02','KA043408',12, 50000);
insert into participated values ('A03','KA031181',13, 25000);
insert into participated values ('A04','KA095477',14, 3000);
insert into participated values ('A05','KA041702',15, 5000);
commit;

select * from participated;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	25000
A02	KA043408	12	50000
A03	KA031181	13	25000
A04	KA095477	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

```
update participated
set damage_amount = 2500
where reg_num='KA031111';

select * from participated;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	25000
A02	KA043408	12	50000
A03	KA031181	13	25000
A04	KA095477	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

```
insert into accident values(101,'2020-12-01','Xavier Road');
insert into participated values('A01','KA031111',101, 1001);
commit;
select * from accident;
```

report_num	accident_date	location
11	2001-01-03	Mysore Road
12	2021-01-03	Southend Circle
13	2020-03-03	Bulltemple Road
14	2017-02-08	Mysore Road
15	2004-03-05	Kanakpura Road
101	2020-12-01	Xavier Road
NULL	NULL	NULL

```
select * from participated;
```

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	25000
A02	KA043408	12	50000
A03	KA031181	13	25000
A04	KA095477	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

```
insert into car values('KA01010', 'Accord', 2002);
insert into owns values('A02', 'KA01010');
insert into accident values(200, '2008-12-01', 'Pinto Road');
insert into participated values('A02', 'KA01010', 200, 500);
commit;

select * from car;
```

reg_num	model	year
KA01010	Accord	2002
KA031181	Lancer	1957
KA041702	Audi	2005
KA043408	Honda	2008
KA052250	Indica	1990
KA095477	Toyota	1998
NULL	NULL	NULL

select * from owns;

driver_id	reg_num
A02	KA01010
A03	KA031181
A05	KA041702
A02	KA043408
A01	KA052250
A04	KA095477
NULL	NULL

select * from accident;

report_num	accident_date	location
11	2001-01-03	Mysore Road
12	2021-01-03	Southend Circle
13	2020-03-03	Bulltemple Road
14	2017-02-08	Mysore Road
15	2004-03-05	Kanakpura Road
101	2020-12-01	Xavier Road
200	2008-12-01	Pinto Road
NULL	NULL	NULL

select * from participated;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	25000
A02	KA01010	200	500
A02	KA043408	12	50000
A03	KA031181	13	25000
A04	KA095477	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

select count(*) from accident where year(accident_date)=2008;

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

count(*)
1

select count(*) from participated where reg_num in (select reg_num from car where model="Accord");

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

count(*)
1

PROGRAM 2: **BANKING ENTERPRISE DATABASE**

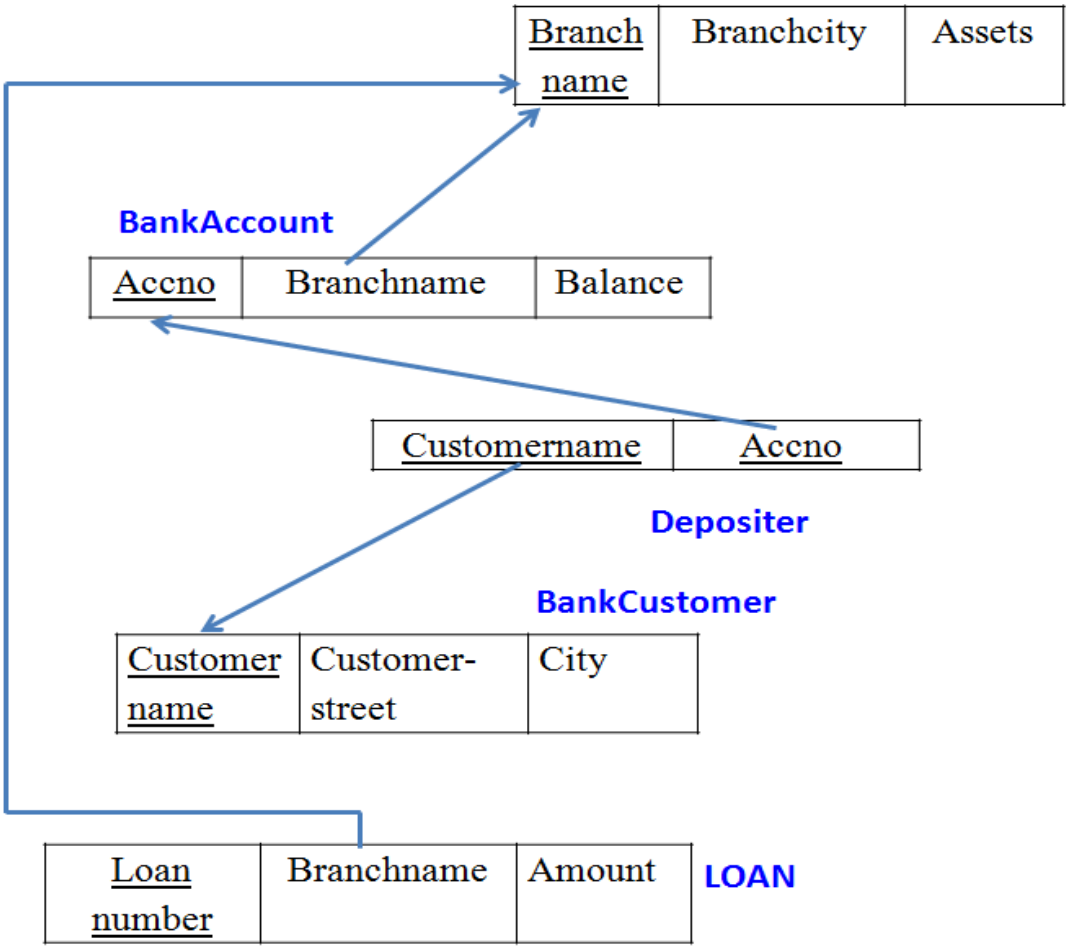
Consider the following database for a banking enterprise.

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).

INTRODUCTION: This database is developed for supporting banking facilities. Details of the branch along with the accounts and loans handled by them are recorded. Also details of the depositors of the corresponding branches are maintained.

Schema Diagram



-- Creating the Database and the tables

```
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;
```

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

```
select * from bank_account;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

```
select * from bank_customer;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_name	customer_street	customer_city
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	National_College_Road	Bangalore
Nikhil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

select * from depositer;

Result Grid | Filter Rows: | Edi

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

select * from loan;

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

loan_number	branch_name	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParliamentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

-- 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);
```

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

customer_name
Dinesh

-- 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');
```

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

customer_name
Nikhil

-- Query 5

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

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

PROGRAM 3: SUPPLIER DATABASE

Consider the following schema:

SUPPLIERS(<u>sid: integer</u>, sname: string, address: string)

PARTS(<u>pid: integer</u>, pname: string, color: string)

CATALOG(<u>sid: integer, pid</u>: 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.

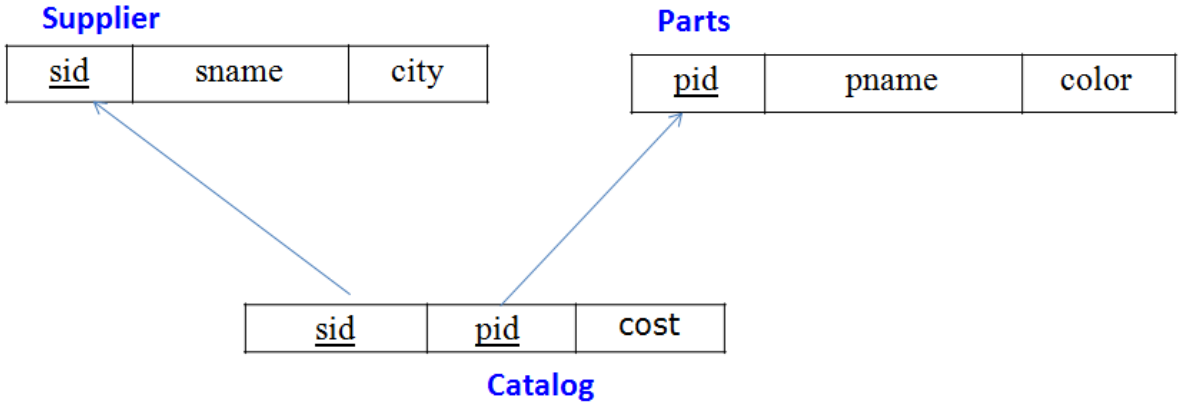
iii) Find the snames of suppliers who supply every red part.

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).

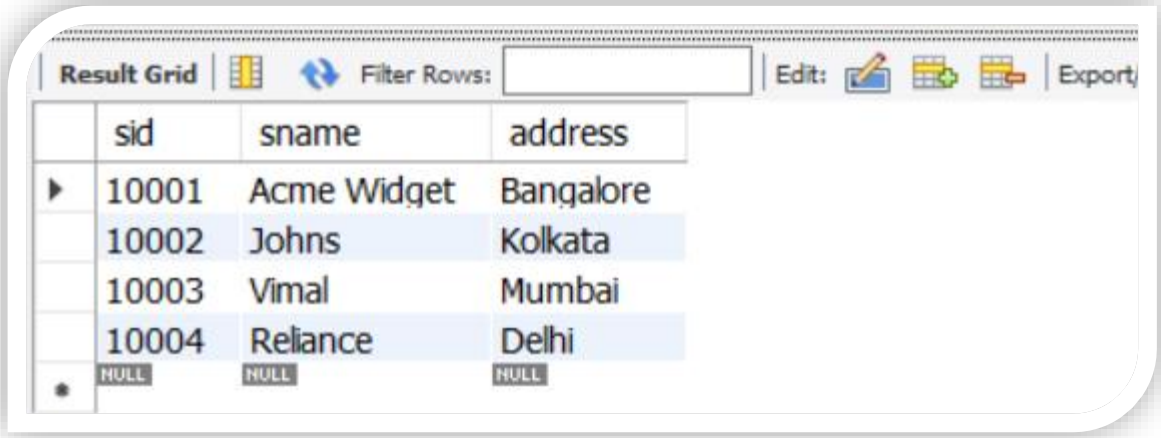
vi) For each part, find the sname of the supplier who charges the most for that part.

Schema Diagram



```
--  Creating databases and tables

Create  database supplier;
use supplier;
create table SUPPLIERS(sid integer,sname varchar(20),address varchar(40),primary key(sid));
INSERT INTO `supplier`.`suppliers` (`sid`, `sname`, `address`) VALUES ('10001', 'Acme Widget', 'Bangalore');
INSERT INTO `supplier`.`suppliers` (`sid`, `sname`, `address`) VALUES ('10002', 'Johns', 'Kolkata');
INSERT INTO `supplier`.`suppliers` (`sid`, `sname`, `address`) VALUES ('10003', 'Vimal', 'Mumbai');
INSERT INTO `supplier`.`suppliers` (`sid`, `sname`, `address`) VALUES ('10004', 'Reliance', 'Delhi');
commit;
select* from SUPPLIERS;
```



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

create table PARTS(pid integer,pname varchar(20),color varchar(30),primary key(pid));

INSERT INTO `supplier`.`parts` (`pid`, `pname`, `color`) VALUES ('20001', 'Book', 'Red');

INSERT INTO `supplier`.`parts` (`pid`, `pname`, `color`) VALUES ('20002', 'Pen', 'Red');

INSERT INTO `supplier`.`parts` (`pid`, `pname`, `color`) VALUES ('20003', 'Pencil', 'Green');

INSERT INTO `supplier`.`parts` (`pid`, `pname`, `color`) VALUES ('20004', 'Mobile', 'Green');

INSERT INTO `supplier`.`parts` (`pid`, `pname`, `color`) VALUES ('20005', 'Charger', 'Black');

commit;

select* from PART;

Result Grid | Filter Rows: | Edit:

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

create table CATALOG(sid integer,pid integer,foreign key(sid) references SUPPLIERS(sid),foreign key(pid) references PARTS(pid), cost integer,primary key(sid,pid));

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10001', '20001', '10');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10001', '20002', '10');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10001', '20003', '30');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10001', '20004', '10');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10001', '20005', '10');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10002', '20001', '10');

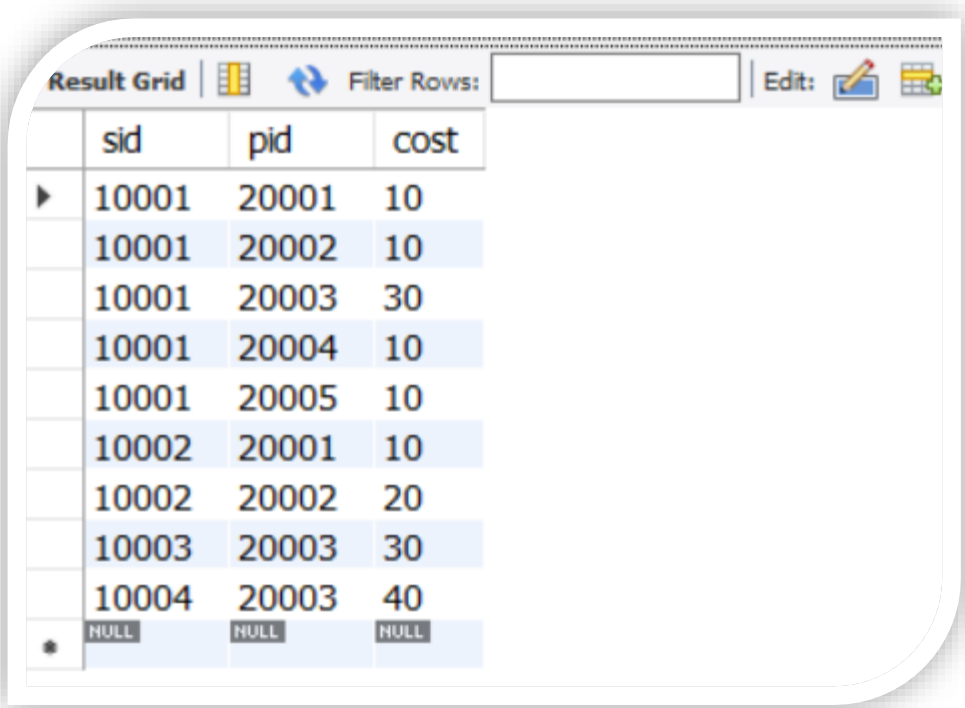
INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10002', '20002', '20');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10003', '20003', '30');

INSERT INTO `supplier`.`catalog` (`sid`, `pid`, `cost`) VALUES ('10004', '20003', '40');

commit;

select* from CATALOG;



Result Grid | Filter Rows: | Edit:

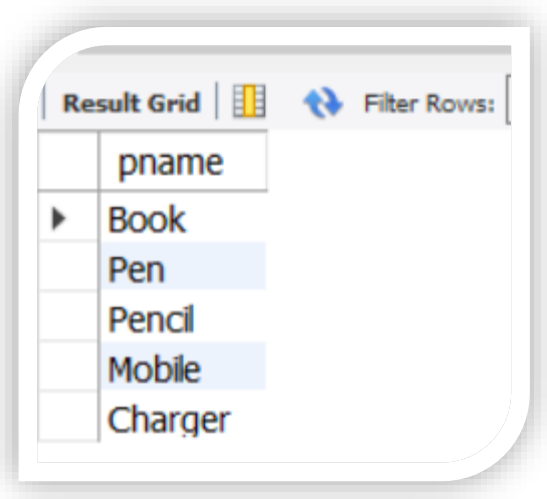
sid	pid	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

-- Query 1

SELECT DISTINCT P.pname

FROM Parts P, Catalog C

WHERE P.pid = C.pid;



Result Grid | Filter Rows:

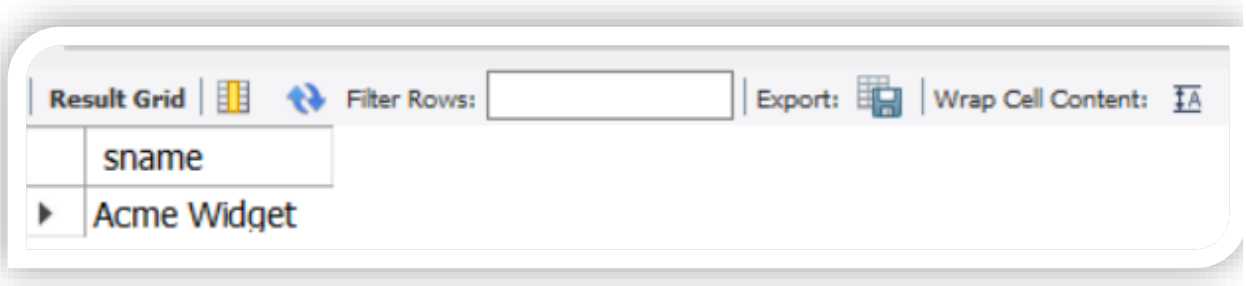
pname
Book
Pen
Pencil
Mobile
Charger

-- Query 2

select S.sname from SUPPLIERS S where not exists

(select P.pid from PARTS P where not exists

(select C.sid from CATALOG C where C.sid = S.sid and C.pid = P.pid));

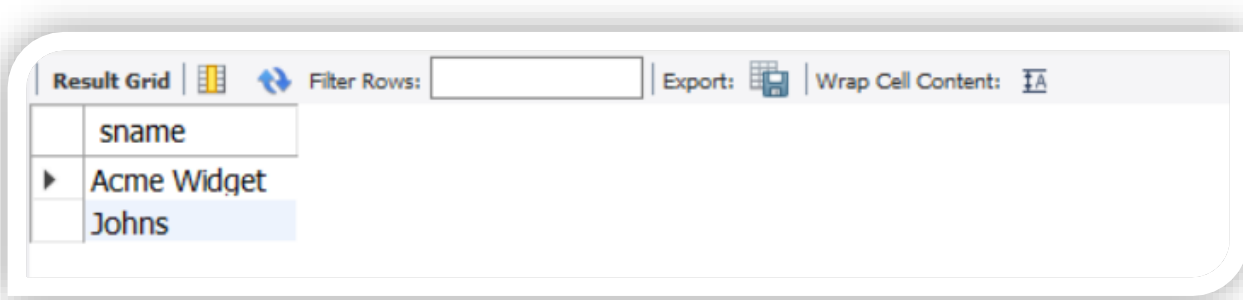


-- Query 3

select S.sname from SUPPLIERS S where not exists

(select P.pid from PARTS P where P.color = 'Red' and

(not exists (select C.sid from CATALOG C where C.sid = S.sid and C.pid = P.pid)));



-- Query 4

select P.pname from PARTS P, CATALOG C, SUPPLIERS S

where P.pid = C.pid and C.sid = S.sid and S.sname = 'Acme Widget'

and not exists (select * from CATALOG C1, SUPPLIERS S1

where P.pid = C1.pid and C1.sid = S1.sid and S1.sname <> 'Acme Widget');

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

pname
Mobile
Charger

-- Query 5

SELECT DISTINCT C.sid FROM Catalog C

WHERE C.cost > (SELECT AVG (C1.cost)

FROM Catalog C1

WHERE C1.pid = C.pid);

Result Grid | Filter Rows: | Export:

sid
10002
10004

-- Query 6

SELECT P.pid, S.sname

FROM Parts P, Suppliers S, Catalog C

WHERE C.pid = P.pid

```
AND C.sid = S.sid
AND C.cost = (SELECT MAX(C1.cost)
FROM Catalog C1
WHERE C1.pid = P.pid);
```

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

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

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).

-- Creating database and tables

```
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');
```

Student table

Result Grid | Filter Rows: | Edit:

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

Result Grid | Filter Rows: | Edit: | Expor

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

Class table

Result Grid | Filter Rows: | Edit:

cname	metts_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

-- 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

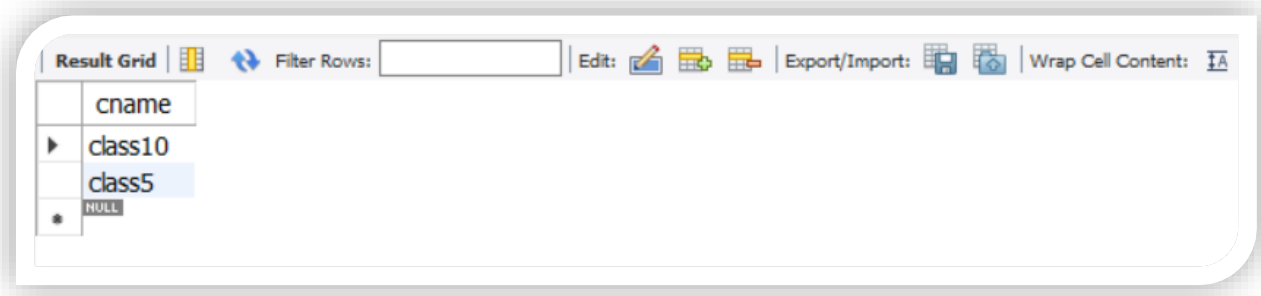
F.fname = 'Harish' AND S.lvl = 'Jr';

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

Sname
Tom

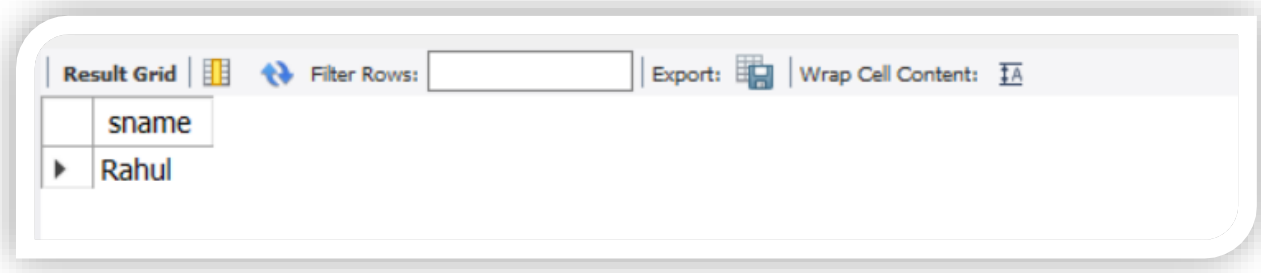
-- Query 2

```
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

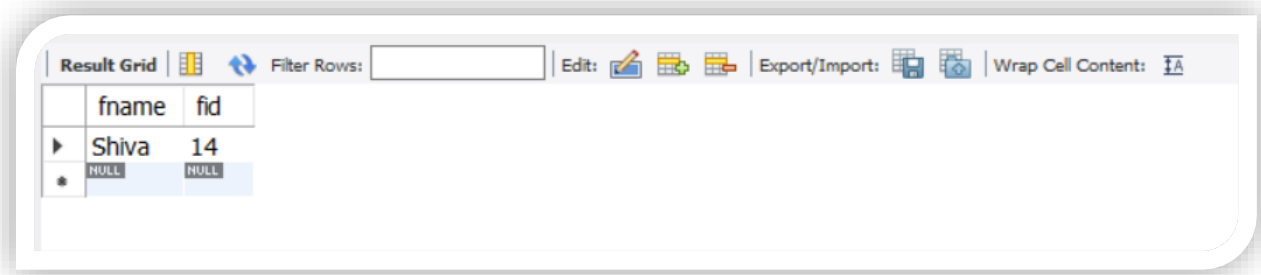
```
SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum IN (SELECT E1.snum
                FROM Enrolled E1, Enrolled E2, Class C1, Class C2
                WHERE E1.snum = E2.snum AND E1.cname <> E2.cname
                AND E1.cname = C1.cname
                AND E2.cname = C2.cname AND C1.metts_at =
C2.metts_at);
```



sname
Rahul

-- 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) );
```



fname	fid
Shiva	14
NULL	NULL

-- 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);
```

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

fname
Harish
MV
Mira
Shiva

-- Query 6

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

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

sname
Rita

-- 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;
```

age	lvl
19	Sr
20	Jr
21	Sr

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.

```
--  Creating database and tables
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;
```

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

Field	Type	Null	Key	Default	Extra
flno	int	NO	PRI	NULL	
fromplace	varchar(15)	YES		NULL	
toplace	varchar(15)	YES		NULL	
distance	int	YES		NULL	
departs	datetime	YES		NULL	
arrives	datetime	YES		NULL	
price	int	YES		NULL	

```
create table aircraft(
        aid int,
    aname varchar(15),
    cruisingrange int,
    primary key (aid)
);
desc aircraft;
```

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

Field	Type	Null	Key	Default	Extra
aid	int	NO	PRI	NULL	
aname	varchar(15)	YES		NULL	
cruisingrange	int	YES		NULL	

```
create table employees (
        eid int,
    ename varchar(15),
    salary int,
    primary key (eid)
);
desc employees;
```

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

Field	Type	Null	Key	Default	Extra
eid	int	NO	PRI	NULL	
ename	varchar(15)	YES		NULL	
salary	int	YES		NULL	

```
create table certified (
        eid int,
    aid int,
    foreign key (eid) references employees(eid),
    foreign key (aid) references aircraft(aid)
);
desc certified;
```

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

Field	Type	Null	Key	Default	Extra
eid	int	YES	MUL	NULL	
aid	int	YES	MUL	NULL	

```
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;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

flno	fromplace	toplace	distance	departs	arrives	price
101	Bangalore	Delhi	2500	2005-05-13 07:15:31	2005-05-13 18:15:31	5000
102	Bangalore	Lucknow	3000	2013-05-05 07:15:31	2013-05-05 11:15:31	6000
103	Lucknow	Delhi	500	2013-05-05 12:15:31	2013-05-05 17:15:31	3000
104	Bangalore	Frankfurt	8500	2013-05-05 07:15:31	2013-05-05 23:15:31	75000
105	Kolkata	Delhi	3400	2013-05-05 07:15:31	2013-05-05 09:15:31	7000
106	Bangalore	Kolkata	1000	2013-05-05 01:15:30	2013-05-05 09:20:30	10000
107	Bangalore	Frankfurt	8000	2013-05-05 07:15:31	2013-05-05 22:15:31	60000
108	Lucknow	Kolkata	1000	2013-05-05 11:30:30	2013-05-05 15:20:30	10000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
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;
```

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

aid	aname	cruisingrange
101	747	3000
102	Boeing	900
103	647	800
104	Dreamliner	10000
105	Boeing	3500
106	707	1500
107	Dream	120000
108	707	760
109	747	1000
NULL	NULL	NULL

```
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;
```

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

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

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);

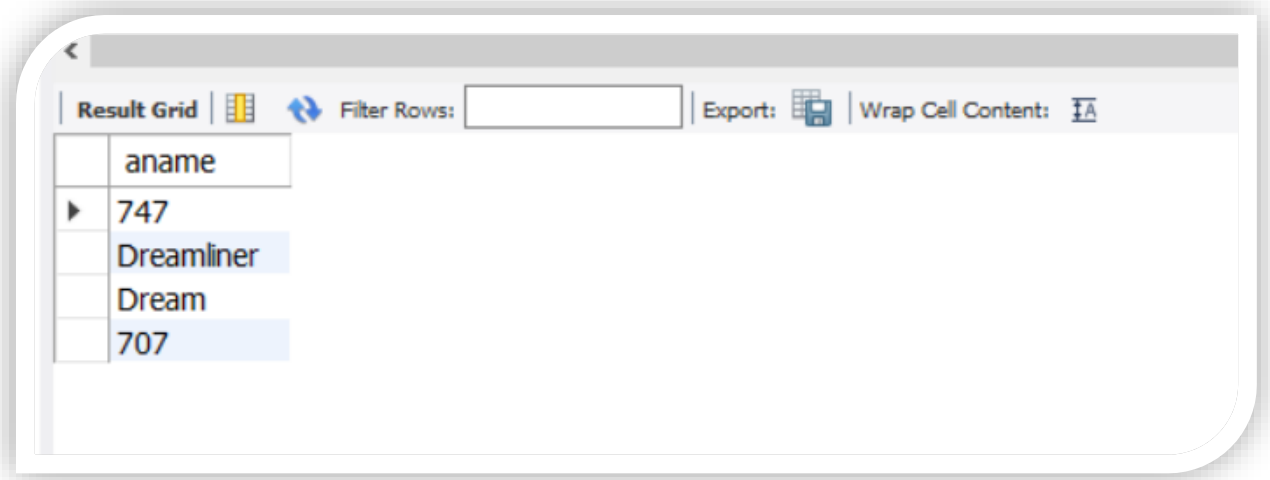
```
insert into certified values(702, 108);
insert into certified values(701, 109);
commit;
select * from certified;
```

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

eid	aid
701	101
701	102
701	106
701	105
702	104
703	104
704	104
702	107
703	107
704	107
702	101
702	108
701	109

-- 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;
```

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

max(a.cruisingrange)	eid
3500	701
120000	702

-- Query 3

select ename from employees where salary <(

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

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

ename
A
E

-- 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;

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

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

-- Query 5

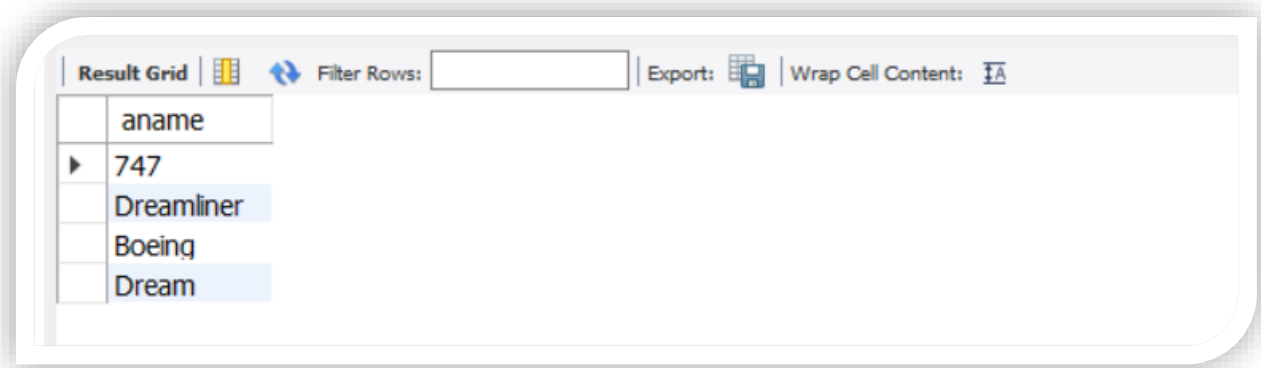
select ename from employees where eid in(
select eid from certified where aid in(
select aid from aircraft where aname = 'Boeing'));

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

ename
A

-- Query 6

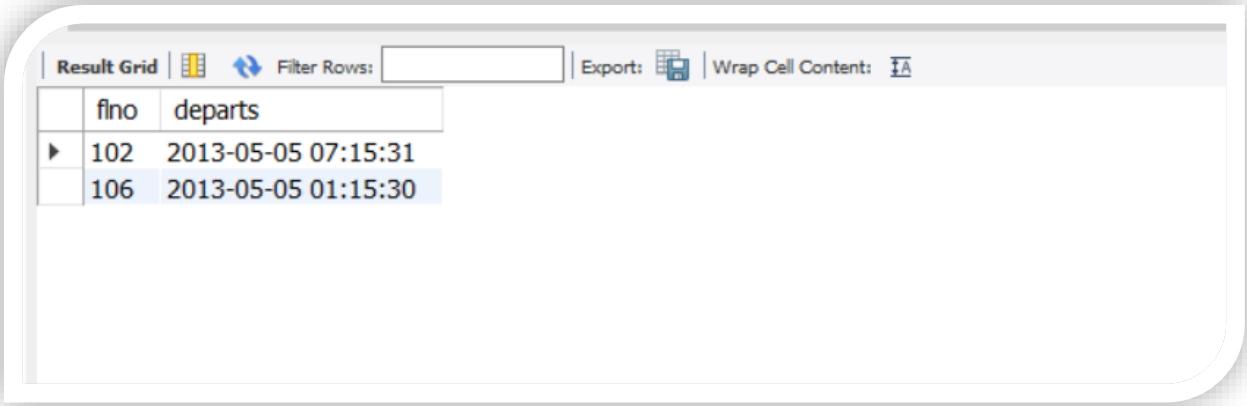
select aname from aircraft where cruisingrange > any (select distance from flights where fromplace='Bangalore' and toplace='Delhi');



-- 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 | Filter Rows: | Export: | Wrap Cell Content:

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