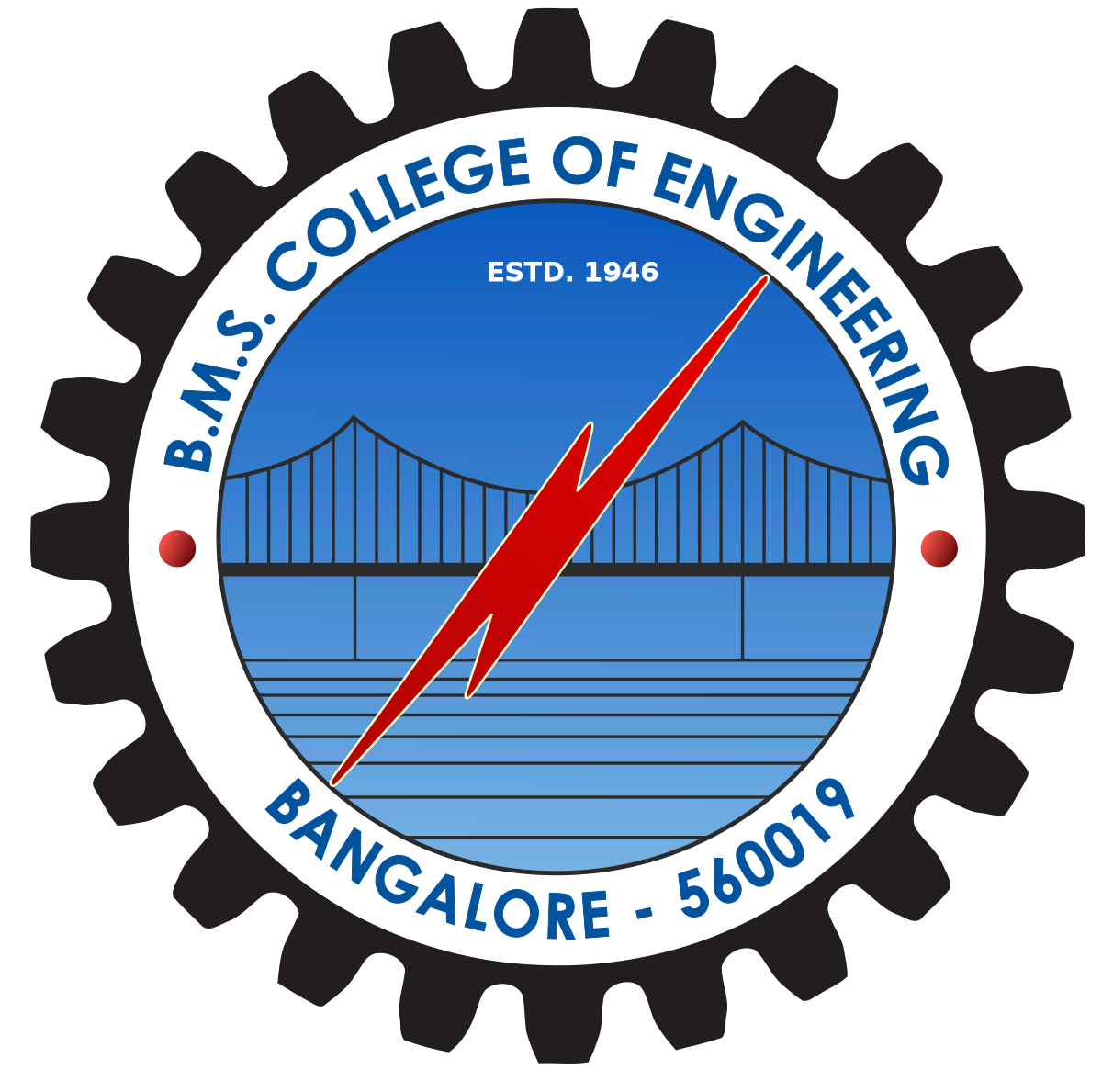
**B.M.S. COLLEGE OF ENGINEERING**

**(AUTONOMOUS COLLEGE UNDER VTU)**

**BENGALURU-19**



**LAB TEST 1 REPORT**

**NAME : ROHIT SATHEESH NAIR**

**USN : 1BM19CS206**

**COURSE NAME : DATABASE MANAGEMENT**

**SYSTEMS**

**COURSE TITLE : 19CS4PCDBM**

**SEM : 4**

**SECTION :D**

**PROGRAM 1: INSURANCE DATABASE**

Consider the Insurance database given below. The primary keys are underlined and the data types are specified.

PERSON (driver-id #: String, name: String, address: String)

CAR (Regno: String, model: String, year: int)

ACCIDENT (report-number: int, date: date, location: String)

OWNS (driver-id #: String, Regno: String)

PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)

**i. Create the above tables by properly specifying the primary keys and the foreign keys**

create database Lab1;

use Lab1;

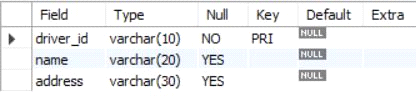
create table Lab1.person (driver\_id varchar(10),

name varchar(20),

address varchar(30),

primary key(driver\_id));

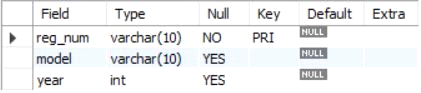
desc person;



create table car(reg\_num varchar(10),model

varchar(10),year int,primary key(reg\_num));

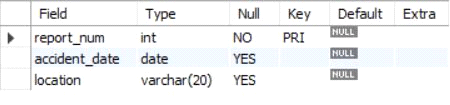
desc car;



create table accident(report\_num int,accident\_date

date,location varchar(20),primary key(report\_num));

desc accident;

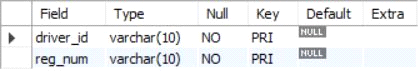


create table owns(driver\_id varchar(10),reg\_num

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;



create table participated(driver\_id

varchar(10), reg\_num varchar(10),report\_num int, damage\_amount int,

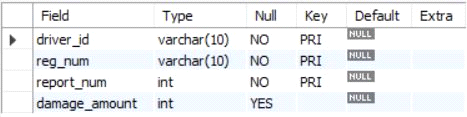
primary key(driver\_id,reg\_num,report\_num),

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;



**ii. Enter at least five tuples for each relation**

insert into person values('A01','Richard','Srinivas Nagar');

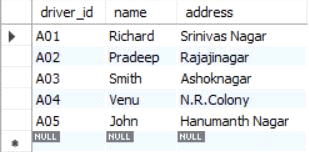
insert into person values('A02','Pradeep','Rajajinagar');

insert into person values('A03','Smith','Ashoknagar');

insert into person values('A04','Venu','N.R.Colony');

insert into person values('A05','John','Hanumanth Nagar');

select \* from person;



insert into car values('KA052250','Indica', 1990);

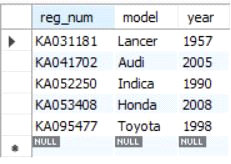
insert into car values('KA031181','Lancer', 1957);

insert into car values('KA095477','Toyota',1998);

insert into car values('KA053408','Honda',2008);

insert into car values('KA041702','Audi',2005);

select \* from car;



insert into accident values(11,'2003-01-01','Mysore Road');

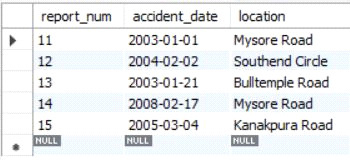
insert into accident values(12,'2004-02-02','Southend Circle');

insert into accident values(13,'2003-01-21','Bulltemple Road');

insert into accident values(14,'2008-02-17','Mysore Road');

insert into accident values(15,'2005-03-04','Kanakpura Road');

select \* from accident;



insert into owns values('A01','KA052250');

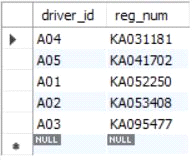
insert into owns values('A02','KA053408');

insert into owns values('A03','KA095477');

insert into owns values('A04','KA031181');

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

select \* from owns;



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

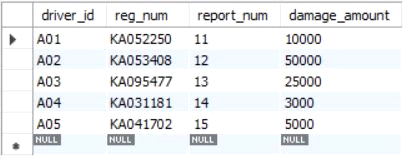
insert into participated values('A02','KA053408',12,50000);

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

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

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

select \* from participated;

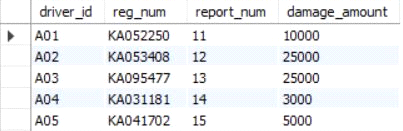


**iii. Demonstrate how you**

**a. Update the damage amount for the car with a specific Regno in the accident with report number 12 to 25000.**

update participated set damage\_amount=25000 where report\_num=12;

select \* from participated;



**b. Add a new accident to the database.**

insert into person values('A06','Jospeh','Shanti Nagar');

insert into car values('KA012370','Honda', 2008);

insert into accident values(16,'2008-01-01','MG Road');

insert into owns values('A06','KA012370');

insert into participated values('A06','KA012370',16,15000);

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

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



**v. Find the number of accidents in which cars belonging to a specific model were involved**

select count(\*) as no\_of\_acc from participated where reg\_num in(select reg\_num from car where model='Honda');



**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

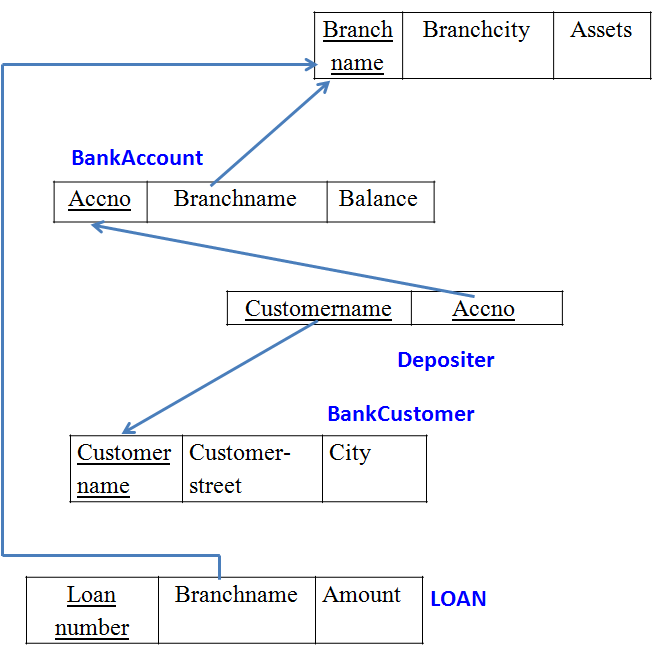
specific city (Ex. Delhi).

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**



**i. Create the above tables by properly specifying the primary keys and the**

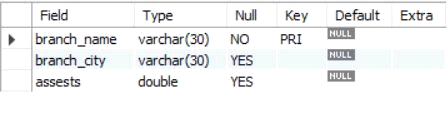
**foreign keys.**

create database Lab2;

use Lab2;

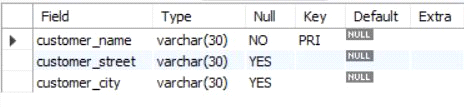
create table Branch(branch\_name varchar(30),branch\_city varchar(30),assests real, primary key(branch\_name));

desc Branch;



create table BankCustomer(customer\_name varchar(30),customer\_street varchar(30),customer\_city varchar(30), primary key(customer\_name));

desc BankCustomer;



create table BankAccount(

accno int,

branch\_name varchar(20),

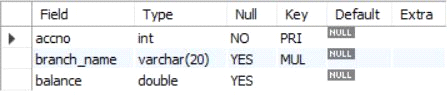
balance real,

primary key(accno),

foreign key(branch\_name) references Branch(branch\_name)

);

desc BankAccount;



create table Depositer(

customer\_name varchar(20),

accno int,

primary key(customer\_name,accno),

foreign key(customer\_name) references BankCustomer(customer\_name),

foreign key(accno) references BankAccount(accno)

);

desc Depositer;



create table Loan(

loan\_number int,

branch\_name varchar(20),

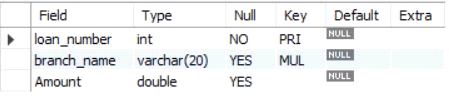
Amount real,

primary key(loan\_number),

foreign key(branch\_name) references Branch(branch\_name)

);

desc Loan;



**ii. Enter at least five tuples for each relation.**

insert into Branch values('SBI\_Chamrajpet','Bangalore',50000);

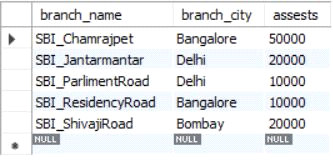
insert into Branch values('SBI\_ResidencyRoad','Bangalore',10000);

insert into Branch values('SBI\_ShivajiRoad','Bombay',20000);

insert into Branch values('SBI\_ParlimentRoad','Delhi',10000);

insert into Branch values('SBI\_Jantarmantar','Delhi',20000);

select \*from Branch;



insert into Loan values(2,'SBI\_ResidencyRoad',2000);

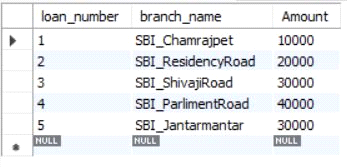
insert into Loan values(1,'SBI\_Chamrajpet',1000);

insert into Loan values(3,'SBI\_ShivajiRoad',3000);

insert into Loan values(4,'SBI\_ParlimentRoad',4000);

insert into Loan values(5,'SBI\_Jantarmantar',3000);

select \*from Loan;



insert into BankAccount values(1,'SBI\_Chamrajpet',2000);

insert into BankAccount values(2,'SBI\_ResidencyRoad',5000);

insert into BankAccount values(3,'SBI\_ShivajiRoad',6000);

insert into BankAccount values(4,'SBI\_ParlimentRoad',9000);

insert into BankAccount values(5,'SBI\_Jantarmantar',8000);

insert into BankAccount values(6, 'SBI\_ShivajiRoad', 4000);

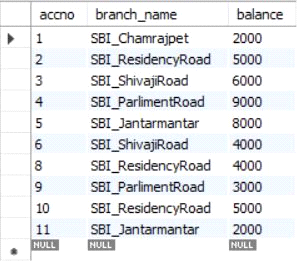
insert into BankAccount values(8, 'SBI\_ResidencyRoad', 4000);

insert into BankAccount values(9, 'SBI\_ParlimentRoad', 3000);

insert into BankAccount values(10, 'SBI\_ResidencyRoad', 5000);

insert into BankAccount values(11, 'SBI\_Jantarmantar', 2000);

select \*from BankAccount;



insert into BankCustomer values ('Avinash', 'Bull\_Temple\_Road', 'Bangalore');

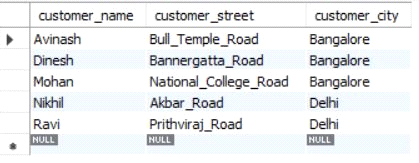
insert into BankCustomer values ('Dinesh', 'Bannergatta\_Road', 'Bangalore');

insert into BankCustomer values ('Mohan', 'National\_College\_Road', 'Bangalore');

insert into BankCustomer values ('Nikhil', 'Akbar\_Road', 'Delhi');

insert into BankCustomer values ('Ravi', 'Prithviraj\_Road', 'Delhi');

select \*from BankCustomer;



insert into Depositer values('Avinash', 1);

insert into Depositer values('Dinesh', 2);

insert into Depositer values('Nikhil', 4);

insert into Depositer values('Ravi', 5);

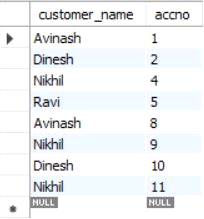
insert into Depositer values('Avinash', 8);

insert into Depositer values('Nikhil', 9);

insert into Depositer values('Dinesh', 10);

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

select \*from Depositer;



**iii. Find all the customers who have at least two accounts at the *Main* branch (ex. SBI\_ResidencyRoad).**

select c.customer\_name

from BankCustomer c

where exists(

select d.customer\_name

from Depositer d, BankAccount ba

where

d.accno=ba.accno and

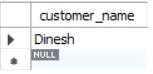
c.customer\_name=d.customer\_name and

ba.branch\_name='SBI\_ResidencyRoad'

group by d.customer\_name

having count(d.customer\_name)>=2

);

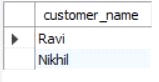


**iv. Find all the customers who have an account at *all* the branches located in a**

**specific city (Ex. Delhi).**

select distinct d.customer\_name from Depositer d where exists( select \* from BankAccount ba

where ba.accno=d.accno and exists (select \* from Branch b where b.branch\_name = ba.branch\_name and b.branch\_city='Delhi'));

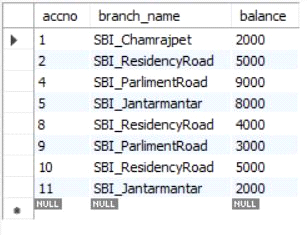


**v. Demonstrate how you delete all account tuples at every branch located in**

**a specific city (Ex. Bombay).**

delete from BankAccount where branch\_name in (select branch\_name from branch where branch\_city = 'Bombay');

select \*from BankAccount;



**PROGRAM 3: SUPPLIER DATABASE**

Consider the following schema:

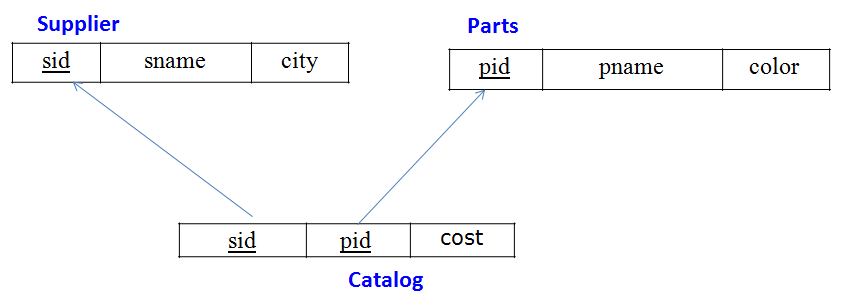
SUPPLIERS(sid: integer, sname: string, address: string)

PARTS(pid: integer, pname: string, color: string)

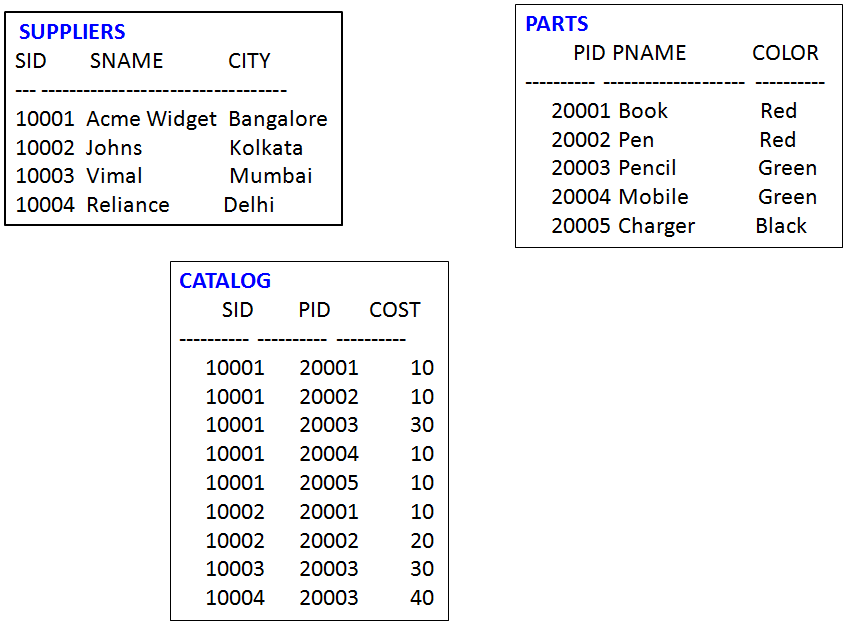
CATALOG(sid: integer, pid: integer, cost: real)

The Catalog relation lists the prices charged for parts by Suppliers.

**Schema Diagram**



**Table Data**

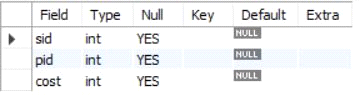


create database Lab3;

use Lab3;

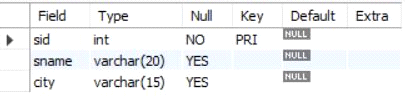
create table catalog(sid int,pid int,cost int);

desc catalog;



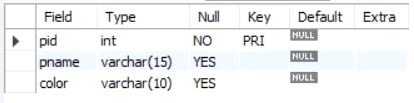
create table supplier(sid int,sname varchar(20),city varchar(15),primary key (sid));

desc supplier;



create table parts(pid int,pname varchar(15),color varchar(10),primary key (pid));

desc parts;



insert into supplier values(10001, 'Acme Widget','Bengaluru');

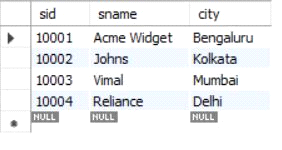
insert into supplier values(10002,'Johns','Kolkata');

insert into supplier values(10003, 'Vimal','Mumbai');

insert into supplier values(10004, 'Reliance','Delhi');

insert into supplier values(10005, 'Mahindra','Mumbai');

select \* from supplier;



insert into parts values(20001, 'Book','Red');

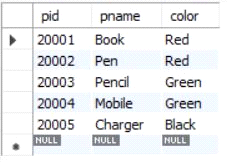
insert into parts values(20002, 'Pen','Red');

insert into parts values(20003, 'Pencil','Green');

insert into parts values(20004, 'Mobile','Green');

insert into parts values(20005, 'Charger','Black');

select \* from parts;



insert into catalog values(10001, '20001','10');

insert into catalog values(10001, '20002','10');

insert into catalog values(10001, '20003','30');

insert into catalog values(10001, '20004','10');

insert into catalog values(10001, '20005','10');

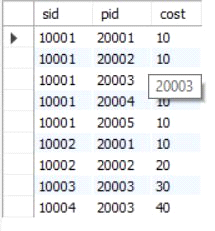
insert into catalog values(10002, '20001','10');

insert into catalog values(10002, '20002','20');

insert into catalog values(10003, '20003','30');

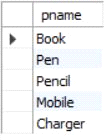
insert into catalog values(10004, '20003','40');

select \* from catalog;



**1.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;



**2.Find the snames of suppliers who supply every part.**

select s.sname from supplier 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));

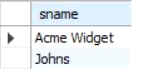


**3.Find the snames of suppliers who supply every red part.**

select s.sname from supplier 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)));



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

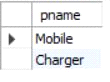
select p.pname from parts p , catalog c, supplier s

where p.pid = c.pid and c.sid = s.sid and s.sname = 'Acme Widget'

and not exists

(select \* from catalog c1, supplier s1 where

p.pid = c1.pid and c1.sid = s1.sid and s1.sname <> 'Acme Widget');



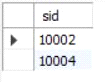
**5.Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).**

select distinct c.sid from catalog c

where c.cost > ( select avg (c1.cost)

from catalog c1

where c1.pid = c.pid );



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

select p.pid, s.sname

from parts p, supplier s, catalog c

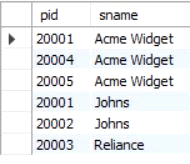
where c.pid = p.pid

and c.sid = s.sid

and c.cost = (select MAX(c1.cost)

from catalog c1

where c1.pid = p.pid);



**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, meetsat: time, room: string, fid: integer)

ENROLLED(snum: integer, cname:string)

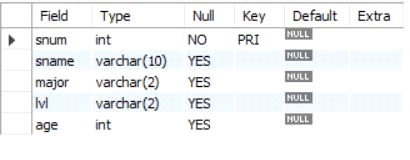
FACULTY(fid: integer, fname:string, deptid: integer)

create database Lab4;

use Lab4;

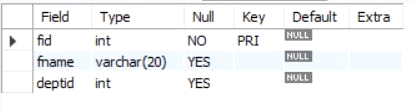
create table student(snum int, sname varchar(10), major varchar(2), lvl varchar(2), age int,primary key (snum));

desc student;



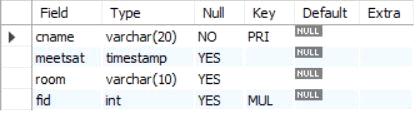
create table faculty(fid int, fname varchar(20), deptid int,primary key(fid));

desc faculty;



create table class(cname varchar(20), meetsat timestamp, room varchar(10), fid int,primary key (cname),foreign key(fid) references faculty(fid));

desc class;

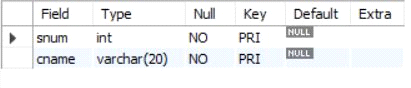


create table enrolled(snum int, cname varchar(20),primary key(snum,cname),

foreign key(snum) references student(snum),

foreign key(cname) references class(cname));

desc enrolled;



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);

select \* from student;



insert into faculty values(11, 'Harish', 1000);

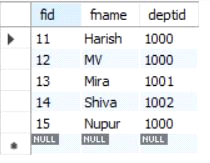
insert into faculty values(12, 'MV', 1000);

insert into faculty values(13 , 'Mira', 1001);

insert into faculty values(14, 'Shiva', 1002);

insert into faculty values(15, 'Nupur', 1000);

select \* from faculty;



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', 12);

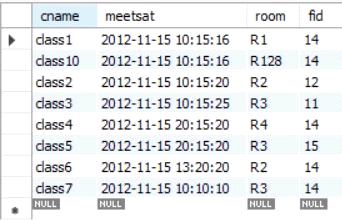
insert into class values('class4', '12/11/15 20:15:20', 'R4', 14);

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);

select \* from class;



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');

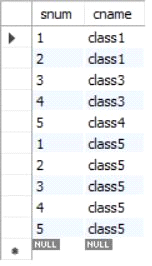
insert into enrolled values(2, 'class5');

insert into enrolled values(3, 'class5');

insert into enrolled values(4, 'class5');

insert into enrolled values(5, 'class5');

select \* from enrolled;



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

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';



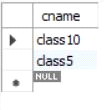
**ii.Find the names of all classes that either meet in room R128 or have five or more Students enrolled.**

SELECT C.cname

FROM class C WHERE C.room = 'R128'

OR C.cname IN (SELECT E.cname FROM enrolled E

GROUP BY E.cname HAVING COUNT(\*) >= 5);



**iii. 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

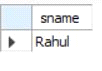
WHERE S.snum IN (SELECT E1.snum

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.meetsat = C2.meetsat);



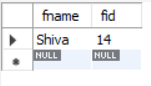
**iv.Find the names of faculty members who teach in every room in which some class is taught.**

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) );



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

SELECT DISTINCT F.fname

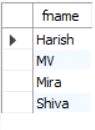
FROM faculty F

WHERE 5 > (SELECT COUNT(E.snum)

FROM class C, enrolled E

WHERE C.cname = E.cname

AND C.fid = F.fid);



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

SELECT DISTINCT S.sname

FROM student S

WHERE S.snum NOT IN (SELECT E.snum

FROM enrolled E );



**vii.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, S.lvl

HAVING S.lvl IN (SELECT S1.lvl FROM Student S1

WHERE S1.age = S.age

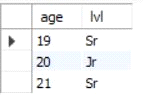
GROUP BY S1.lvl, S1.age

HAVING COUNT(\*) >= ALL (SELECT COUNT(\*)

FROM Student S2

WHERE s1.age = S2.age

GROUP BY S2.lvl, S2.age));



**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.

create database Lab5;

use Lab5;

create table flights(

flno int,

fromplace varchar(15),

toplace varchar(15),

distance int,

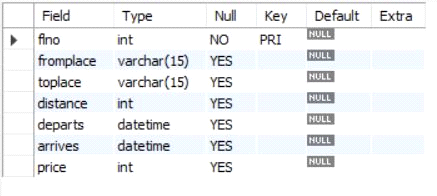
departs datetime,

arrives datetime,

price int,

primary key (flno));

desc flights;



create table aircraft(

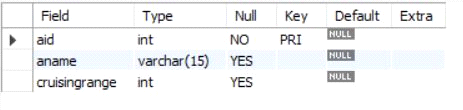
aid int,

aname varchar(15),

cruisingrange int,

primary key (aid));

desc aircraft;



create table employees (

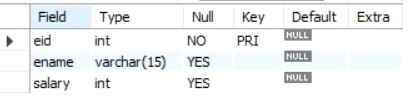
eid int,

ename varchar(15),

salary int,

primary key (eid));

desc employees;



create table certified (

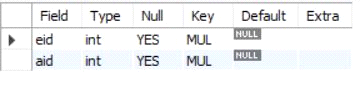
eid int,

aid int,

foreign key (eid) references employees(eid),

foreign key (aid) references aircraft(aid));

desc certified;



insert into flights values(101, 'Bangalore', 'Delhi', 2500, '2005-05-13 07:15:31', '2005-05-13 18:15:31', 5000);

insert into flights values(102, 'Bangalore', 'Lucknow', 3000, '2013-05-05 07:15:31', '2013-05-05 11:15:31', 6000);

insert into flights values(103, 'Lucknow', 'Delhi', 500, '2013-05-05 12:15:31', '2013-05-05 17:15:31', 3000);

insert into flights values(107, 'Bangalore', 'Frankfurt', 8000, '2013-05-05 07:15:31', '2013-05-05 22:15:31', 60000);

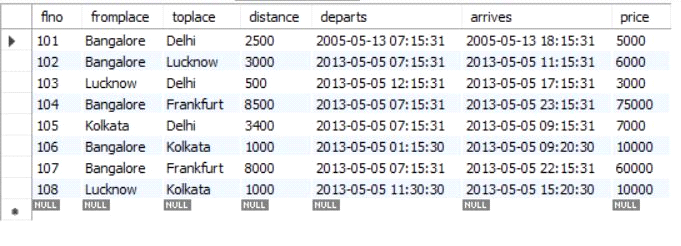
insert into flights values(104, 'Bangalore', 'Frankfurt', 8500, '2013-05-05 07:15:31', '2013-05-05 23:15:31', 75000);

insert into flights values(105, 'Kolkata', 'Delhi', 3400, '2013-05-05 07:15:31', '2013-05-05 09:15:31', 7000);

insert into flights values(106, 'Bangalore', 'Kolkata', 1000, '2013-05-05 01:15:30', '2013-05-05 09:20:30', 10000);

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

select \* from flights;



insert into aircraft values(101, '747', 3000);

insert into aircraft values(102, 'Boeing', 900);

insert into aircraft values(103, '647', 800);

insert into aircraft values(104, 'Dreamliner', 10000);

insert into aircraft values(105, 'Boeing', 3500);

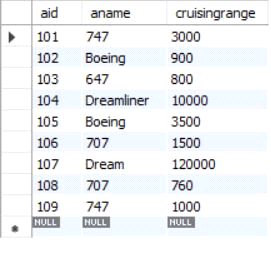
insert into aircraft values(106, '707', 1500);

insert into aircraft values(107, 'Dream', 120000);

insert into aircraft values(108, '707', 760);

insert into aircraft values(109, '747', 1000);

select \* from aircraft;



insert into employees values(701, 'A', 50000);

insert into employees values(702, 'B', 100000);

insert into employees values(703, 'C', 150000);

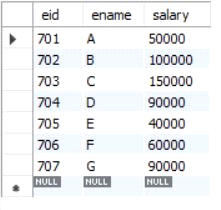
insert into employees values(704, 'D', 90000);

insert into employees values(705, 'E', 40000);

insert into employees values(706, 'F', 60000);

insert into employees values(707, 'G', 90000);

select \* from employees;



insert into certified values(701, 101);

insert into certified values(701, 102);

insert into certified values(701, 106);

insert into certified values(701, 105);

insert into certified values(702, 104);

insert into certified values(703, 104);

insert into certified values(704, 104);

insert into certified values(702, 107);

insert into certified values(703, 107);

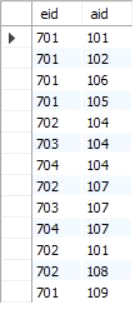
insert into certified values(704, 107);

insert into certified values(702, 101);

insert into certified values(702, 108);

insert into certified values(701, 109);

select \* from certified;



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

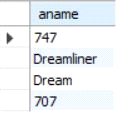
select distinct a.aname from aircraft a where a.aid in (

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

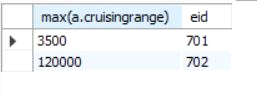
));



**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.**

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;



**iii.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 <(

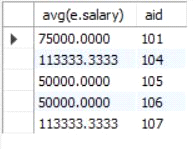
select min(price) from flights where fromplace='Bangalore' and toplace='Frankfurt');



**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.**

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;



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

select ename from employees where eid in(

select eid from certified where aid in(

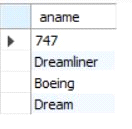
select aid from aircraft where aname = 'Boeing'));



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

select aname from aircraft where cruisingrange > any

(select distance from flights where fromplace='Bangalore' and toplace='Delhi');



**vii.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.**

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));

